GELİŞMİŞ SİNYAL İŞLEME İÇİN VERİMLİ HETEROJEN PARALEL HESAPLAMA

EFFICIENT HETEROGENEOUS PARALLEL COMPUTING FOR ADVANCED SIGNAL PROCESSING

ALPARSLAN FİŞNE

DR. ÖĞR. ÜYESİ ADNAN ÖZSOY

Tez Danışmanı

Hacettepe Üniversitesi

Lisansüstü Egitim-Öğretim ve Sınav Yönetmeliğinin

Bilgisayar Mühendisliği Anabilim Dalı için Öngördüğü

DOKTORA TEZİ olarak hazırlanmıştır.

2022

ÖZET

GELİŞMİŞ SİNYAL İŞLEME İÇİN VERİMLİ HETEROJEN PARALEL HESAPLAMA

Alparslan FİŞNE

Doktora, Bilgisayar Mühendisliği

Danışman: Dr. Öğr. Üyesi Adnan ÖZSOY

Şubat 2022, 129 sayfa

Bu tezde, geleneksel sinyal işleme yöntemlerine kıyasla gelişmiş sinyal işleme algoritmalarının artan hesaplama yükünü azaltmayı amaçlayan güç verimli, gerçek zamanlı, paralel uygulamalar tasarlanmıştır. Sensör sinyal işleme mimarisinde yer alan sayısal alt çevrimi, makine öğrenmesi tabanlı hedef tespiti ve sıkıştırılmış algılama tabanlı geliş açısı kestirimi hesaplamaları için darboğaz işlem noktaları, bellek gereksinimleri ve hesaplama karmaşıklıkları analiz edilmektedir. Paralel hesaplama adımları, kapsamlı analizlerin bir sonucu olarak tasarlanmıştır. Heterojen paralel hesaplama ile gelişmiş sinyal işleme uygulamaları için performans iyileştirmeleri sağlanmıştır. Makine öğrenmesi tabanlı hedef tespitinde, katman optimizasyonları ile hesaplama karmaşıklığı yaklaşık 8.5x azaltılmış ve GPU hesaplama, bu uygulama için CPU paralel sürümüne kıyasla 7x hızlanma sağlamıştır. Sayısal alt çevrimi uygulamasında, son derece düşük güç tüketimi olan 6.5 Watt ile gerçek zamanlı alt çevrim gerçekleştirilmiştir. Sıkıştırılmış algılama tabanlı geliş açısı kestirimi için, gerçek zamanlı hesaplama gereksinimlerini karşılamak adına CPU ve GPU tabanlı iyileştirmeler yapılmıştır. Paralel hesaplamalar ile gerçek zamanlı uygulamalar 15 Watt güç tüketimi ile entegre edilmiştir. Böylece düşük güç tüketimi gereksinimleri olan mobil sensör mimarileri için verimli

bir çözüm sunulmuştur. Hipotezlere ve performans testlerinin sonuçlarına göre, bu tez çalışması, uzun batarya ömrüne ihtiyaç duyan uç hesaplama birimlerinin gerçek zamanlı sensör sinyal işleme gereksinimlerini karşılamaya katkıda bulunan bir tasarım konsepti sunmuştur.

Anahtar Kelimeler: paralel hesaplama, enerji verimliliği, makine öğrenmesi, sıkıştırılmış algılama, sensör sinyal işleme, CPU, GPGPU.

ABSTRACT

EFFICIENT HETEROGENEOUS PARALLEL COMPUTING FOR ADVANCED SIGNAL PROCESSING

Alparslan FİŞNE

Doctor of Philosophy, Department of Computer Engineering

Supervisor: Asst. Prof. Dr. Adnan ÖZSOY

February 2022, 129 pages

In this thesis, power-efficient, real-time, parallel applications are designed which aims to reduce increased computational overhead of advanced signal processing algorithms compared to conventional signal processing methods. The bottleneck processing points, memory requirements and computational complexity are analyzed for computations of digital down conversion, machine learning-based target detection and compressed sensing-based direction of arrival estimation in the sensor signal processing architecture. The parallel computation steps are designed as a result of comprehensive analysis. Performance improvements have been achieved for advanced signal processing applications with heterogeneous parallel computing. In machine learning-based target detection, the computational complexity has been reduced by approximately 8.5x with the layer optimizations, and GPU computing provided 7x acceleration compared to CPU parallel version for this application. In the digital down conversion application, real-time down conversion was implemented with 6.5 Watts, leading to extremely low power consumption. For the compressed sensing-based direction of arrival estimation, CPU and GPU-based improvements have been performed to succeed real-time computation requirements. With parallel computations, real-time applications were integrated with 15 Watts power consumption. Thus, an efficient solution has been presented for

mobile sensor architectures with low power consumption requirements. According to the hypotheses and the results of the performance tests, this thesis has presented a design concept that contributes to meet the real-time sensor signal processing requirements of edge computing units that need long battery life.

Keywords: parallel computing, energy efficiency, machine learning, compressed sensing, sensor signal processing, CPU, GPGPU.

TEŞEKKÜR

Her şeyden önce danışmanım Dr. Öğr. Üyesi Adnan ÖZSOY'a değerli tavsiye ve rehberliklerinden ötürü ayrıca bu tezin her aşamasında bilgi, deneyim, motivasyon ve teşvikleriyle beni desteklediklerı için teşekkür ederim.

Ayrıca tez kurulu üyelerime Prof. Dr. Hasan Şakir BİLGE, Doç. Dr. Kayhan Mustafa İMRE, Prof. Dr. Özcan ÖZTÜRK ve Prof. Dr. Süleyman TOSUN'a bu tezi gözden geçirdikleri ve içgörülü yorumlar yaptıkları için teşekkür ederim.

Tez çalışmalarım sırasında maddi ve manevi desteklerinden ötürü işyerim ASELSAN A.Ş.'ye ve 2211-C Öncelikli Alanlara Yönelik Yurt İçi Doktora Burs Programı ile çalışmalarıma motivasyon katan TÜBİTAK-BİDEB'e teşekkür ederim.

Tez konusunun seçimindeki rehberlik ve çalışmalarımdaki destekleri için işyerimdeki ekip arkadaşlarıma teşekkür ederim.

Son olarak, eğitim hayatım boyunca bana destekleri ile güç veren anneme, babama ve kardeşlerime minnettarım. Tez çalışmalarım sırasında bana varlıklarıyla güç veren kıymetli eşim Neslihan ve biricik kızıma teşekkür ederim.

İÇİNDEKİLER

Sayfa

ŞEKİLLER DİZİNİ

Sayfa

ÇİZELGELER DİZİNİ

Sayfa

KISALTMALAR

ADAS	:	**A**dvanced **D**river-**A**ssistance **S**ystem
ADC	:	**A**nalog **D**igital **C**onversion
ADMM	:	**A**lternating **D**irection **M**ethod of **M**ultipliers
BLAS	:	**B**asic **L**inear **A**lgebra **S**ubprograms
BO	:	**B**üyük **O**lan
BPDN	:	**B**asis **P**ursuit **D**e**N**oising
BRAM	:	**B**lock **R**andom **A**ccess **M**emory
CDMA	:	**C**ode **D**ivision **M**ultiple **A**ccess
CIC	:	**C**ascaded **I**ntegrator-**C**omb
CNN	:	**C**onvolutional **N**eural **N**etwork
CORDIC	:	**CO**ordinate **R**otation **DI**gital **C**omputer
CPI	:	**C**oherent **P**rocessing **I**nterval
CPU	:	**C**entral **P**rocessing **U**nit
CRB	:	**C**ramer **R**ao **B**ound
CS	:	**C**ompressed **S**ensing
C-SALSA	:	**C**onstrained-**S**plit **A**ugmented **L**agrangian **S**hrinkage **A**lgorithm
CUDA	:	**C**ompute **U**nified **D**evice **A**rchitecture
DARPA	:	**D**efense **A**dvanced **R**esearch **P**rojects **A**gency
DB	:	**D**ar **B**ant
DDC	:	**D**igital **D**own **C**onversion
DFT	:	**D**iscrete **F**ourier **T**ransform
DNN	:	**D**eep **N**eural **N**etwork
DoA	:	**D**irection of **A**rrival

DSP	:	**D**igital **S**ignal **P**rocessor
DTF	:	**D**arbe **T**ekrarlama **F**rekansı
FF	:	**F**lip **F**lop
FFT	:	**F**ast **F**ourier **T**ransform
FIR	:	**F**inite **I**mpulse **R**esponse
FPGA	:	**F**ield **P**rogrammable **G**ate **A**rray
GB	:	**G**eniş **B**ant
GMTI	:	**G**round **M**oving **T**arget **I**ndicator
GNU	:	**G**NU's **N**ot **U**nix
GPU	:	**G**raphics **P**rocessing **U**nit
GPGPU	:	**G**eneral **P**urpose **G**raphics **P**rocessing **U**nit
GSM	:	**G**lobal **S**ystem for **M**obile **C**ommunications
HO	:	**H**ücre **O**rtalamalı
IF	:	**I**ntermediate **F**requency
IFFT	:	**I**nverse **F**ast **F**ourier **T**ransform
IQ	:	**I**nphase **Q**uadrature
İHA	:	**İ**nsansız **H**ava **A**racı
KO	:	**K**üçük **O**lan
LASSO	:	**L**east **A**bsolute **S**hrinkange and **S**election **O**perator
LPF	:	**L**ow **P**ass **F**ilter
LUT	:	**L**ook **U**p **T**able
MAC	:	**M**ultiply **AC**cumulate
MKL	:	**M**ath **K**ernel **L**ibrary
MNIST	:	**M**odified **N**ational **I**nstitute of **S**tandards and **T**echnology
MPI	:	**M**agnetic **P**article **I**maging
MSTAR	:	**M**oving and **S**tationary **T**arget **A**cquisition and **R**ecognition
MUSIC	:	**MU**ltiple **SI**gnal **C**lassification
NCO	:	**N**umerically **C**ontrolled **O**scillator
NUC	:	**N**ext **U**nit of **C**omputing

OFDM	:	**O**rthogonal **F**requency **D**ivision **M**ultiplexing
OpenMP	:	**O**pen **M**ulti-**P**rocessing
OpenCL	:	**O**pen **C**omputing **L**anguage
OS	:	**O**verlap **S**ave
PCA	:	**P**rincipal **C**omponent **A**nalysis
ReLU	:	**R**ectified **L**inear **U**nit
RF	:	**R**adyo **F**rekansı
RNN	:	**R**ecurrent **N**eural **N**etwork
SAR	:	**S**entetik **A**çıklıklı **R**adar
SDR	:	**S**oftware **D**efined **R**adio
SIMD	:	**S**ingle **I**nstruction **M**ultiple **D**ata
SIMT	:	**S**ingle **I**nstruction **M**ultiple **T**hreads
SNR	:	**S**ignal-to-**N**oise **R**atio
STFT	:	**S**hort **T**ime **F**ourier **T**ransform
SVM	:	**S**upport **V**ector **M**achine
SVD	:	**S**ingular **V**alue **D**ecomposition
SWaP-C	:	**S**ize **W**eight and **P**ower-**C**ost
SYAO	:	**S**abit **Y**anlış **A**larm **O**ranı
ULA	:	**U**niform **L**inear **A**rray
UMTS	:	**U**niversial **M**obile **T**elecommunications **S**ystem
WMI	:	**W**oodbury **M**atrix **I**dentity
YOLO	:	**Y**ou **O**nly **L**ook **O**nce
4G	:	**F**ourth **G**eneration
5G	:	**F**ifth **G**eneration
6G	:	**S**ixth **G**eneration

1. GİRİŞ

Bu bölümde tez çalışmasının gelişmiş sinyal işleme uygulamalarındaki hangi problem ve problemlere çözümler sunduğu, katkıları ve organizasyonu anlatılmıştır.

1.1. Motivasyon

Algılayıcı (sensör) sistemleri çalıştıkları ortamdan aldıkları veriyi sayısallaştırdıktan sonra sinyal işleme bloklarına sokarak veri üzerinden anlamlı bir özellik ve sonuç çıkarmaya çalışırlar. Optik tabanlı çalışan bir kamera görüntü üretip görüntüye ait çıkarımlar yaparken, akustik tabanlı çalışan bir mikrofon veya sonar sistemi sesin geliş yönünü ve ayrıştırmasını sağlar. Radyo frekans (RF) sistemlerde yer alan alıcı antenler sayesinde elektromanyetik sinyaller süzülüp geniş/dar bant sinyal analizi, hedef tespiti ve geliş açısı kestirimi gibi sinyal işleme adımları gerçekleştirilir. Bu tez çalışmasında RF tabanlı radar-elektronik harp sistemlerinde yer alabilecek sensör mimarisinde gerçekleşen sinyal işleme adımları incelenmektedir.

Tez çalışmasında geleneksel sinyal işleme blokları yerine daha yüksek algoritma doğruluk başarımı sunan sıkıştırılmış algılama, makine öğrenmesi gibi gelişmiş sinyal işleme yöntemleri kullanılarak gerçek zamanlı çalışabilecek sensör uygulamalarının tasarlanması amaçlanmaktadır. Gelişmiş sinyal işleme yöntemleri ile doğruluk başarımı artmasına rağmen hesaplama yükü de artmaktadır. Özellikle otonom sistemlerde kullanılan sensör birimlerinde istenen güç ve maliyet etkin hesaplama mimarisi yaklaşımı yüksek hesaplama gücü ihtiyacı nedeniyle olumsuz etkilenmektedir. Bu problemin üstesinden gelebilmek için gelişmiş sinyal işleme uygulamalarına yönelik paralel programlamaya uygun mobil sistemler için önerilen GPGPU (general purpose graphics processing unit) tabanlı paralel hesaplama ve optimizasyon yöntemleri çalışılmaktadır.

Performans ihtiyacını karşılayabilmek amacıyla gelişmiş sinyal yöntemleri için yapılan hesaplama başarım artışları ve optimizasyonlar tez çalışmasında anlatılmaktadır. Bu kapsamda 3 temel başlık altında incelemeler sunulacaktır:

- Evrişim filtreleme tabanlı yöntem yerine tez çalışmasında önerilen kısa zamanlı Fourier dönüşümü (STFT) tabanlı dar bant alt çevrimi (DDC) uygulamaları,
- İstatiksel sabit yanlış alarm oranını temel alan hareketli hedef tespiti yerine kullanılan makine öğrenmesi tabanlı hareketli hedef tespiti,
- Kovaryans matris girdi hesabına dayanan geliş açısı kestirimi (DoA) algoritması yerine kullanılan sıkıştırılmış algılama (CS) tabanlı DoA algoritması incelenmektedir.

Gerçek zamanlı sensör çalışma kriteri içerisinde bu uygulamaların çalışması hedeflenmektedir. GPGPU'ların verimli kullanılabilmesi için her bir algoritmanın hesaplanması paralel hesaplamaya uygun hale getirilmektedir. Gerçek zamanlı çalışabilen mobil GPGPU'ların yer aldığı güç etkin sensör sistemlerine bu uygulamaların kazandırımı tez çalışmasının ana motivasyonunu oluşturmaktadır.

1.2. Ana Katkılar

Tezin odak noktası, gelişmiş sinyal işleme uygulamalarının artan hesaplama yükünü azaltabilmek için düşük güç tüketen mobil GPGPU'lar ile paralel hesaplama yapılarak gerçek zamanlı senaryolarda uygulamaların çalışması olmuştur. Hesaplama karmaşıklığı artan bir algoritma akışı güçlü hesaplayıcı mimarilerine sahip çok çekirdekli CPU ve GPU mimarileri ile gerçek zamanlı yapılabilmektedir. Fakat bu mimarilerin mobil kullanıma elverişli sensör sistemlerinde kullanılması sensörün yer aldığı platformun daha çok güç tüketmesine, ağırlığının artmasına ve maliyet artışına neden olur.
GPGPU'lara uygun paralel hesaplama sayesinde hem düşük güç tüketen hem de gerçek zamanlı çalışabilen uygulamaları tasarlayarak literatüre katkılar bu tez çalışması ile sunulmaktadır. Çalışmada her bir gelişmiş sinyal işleme uygulaması için literatüre sunulan katkılar maddeler halinde belirtilmektedir:

- FPGA (Field Programmable Gate Array)-DSP (Digital Signal Processor) mimarilerinde kullanılan evrişim filtreleme tabanlı DDC yerine FFT tabanlı filtre bankaları ile

gömülü mobil GPGPU'lara uygun çarpım tabanlı gerçek zamanlı çalışan ve paralel hesaplama ile iyileştirilmiş 6.5 Watt güç tüketen DDC uygulamasının tasarımı,

- Yüksek gürültü altında geleneksel geliş açısı kestirimi yöntemlerine göre güvenilir açı kestirimi sunan CS-DoA işlemini paralel hesaplamaya uygun hale getiren bir yöntem ile gerçek zamanlı uygulama için CPU-GPU hesaplama ve bellek optimizasyonları,
- Makine öğrenmesi tabanlı hedef tespiti işleminde GPU hesaplama optimizasyonları, sinir ağı katman hesaplama optimizasyonu ve literatürdeki düşük güç tüketimli ilk CNN (convolutional neural network) tabanlı radar hedef tespit uygulaması,
- 15 Watt gibi düşük güç tüketimi ile gerçek zamanlı algoritma hesaplamaları yapılarak sensör sistemlerinin pratik kabiliyetlerinin arttırılmasıdır.

Bu tez çalışması ile birlikte hesaplama karmaşıklığı yüksek makine öğrenmesi tabanlı hedef tespiti ve sıkıştırışmış algılama tabanlı geliş açısı kestirimi uygulamaları, 15 Watt gibi düşük güç tüketen bir mimaride gerçek zamanlı çalışabildiği gösterilmektedir. Ayrıca, tez çalışması sayesinde çok kanallı sayısal alt çevrimi işlemleri yüksek hesaplama gücü içeren FPGA-DSP mimarilerine gerek kalmadan güç verimli paralel hesaplama ve paralel programlamaya uygun hale getiren yaklaşım ile yapılmaktadır. Makine öğrenmesi tabanlı radar hedef tespitinde CNN modelinde katman optimizasyonları gerçekleştirilerek hesaplama yükü azaltılmış olup uygulama CPU ve GPU birimleri ile gerçek zamanlı hesaplanabilir hale getirilmektedir.

1.3. Beklenen Kazanımlar

Tezin sunduğu sonuçlar ile birlikte özellikle düşük güç tüketimi ile çalışması gereken sensör sistemleri veya uç (edge) hesaplama birimlerinin gelişmiş sinyal işleme uygulamalarını kullanımıyla kabiliyetleri artacaktır. Her üç uygulamada da yer alan matris-vektör işlemleri, lineer cebir problemleri, evrişim işlemleri ve dönüşüm yöntemleri sadece radar-elektronik harp senaryolarında değil 5G, nesnelerin interneti, bulut/uç hesaplama ve insansız hava/kara-/deniz sistemlerinde sensör sinyal işleme blokları içerisinde yer alabilmektedir. Bu kapsamda

elde edilen hesaplama başarım artışı, edinilen tecrübeler ve optimizasyon prosedürlerinin düşük güç tüketimi içeren çeşitli alanlara aktarılması önemli bir kazanım olacaktır. Tez çalışmasının bilişsel radar ve elektronik harp teknolojilerinin yanı sıra akustik algılayıcılardaki çözümlere de ilham kaynağı olabileceği düşünülmektedir.

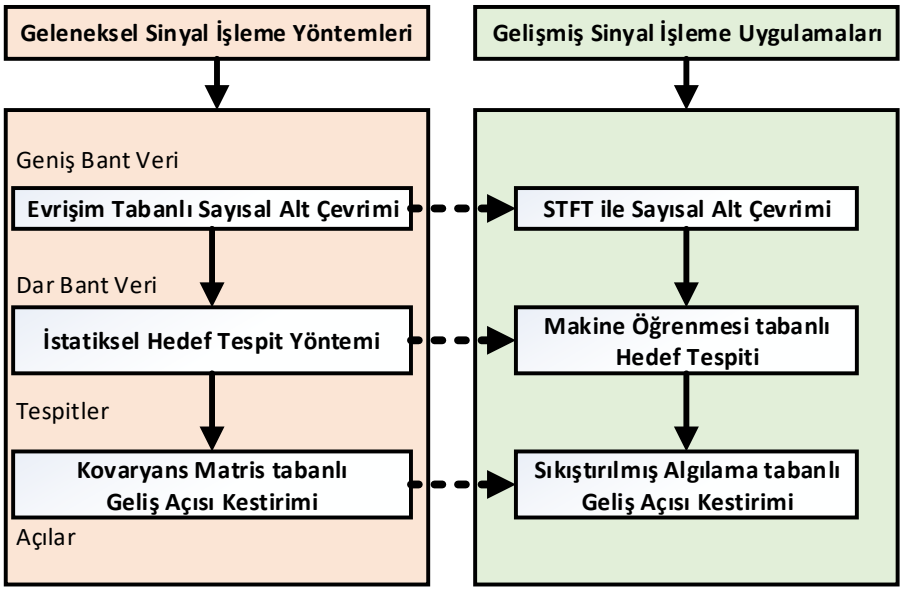
1.4. Organizasyon

Tez çalışmasında yer alan kısaltmaların açıklamaları KISALTMALAR listesinde verilmektedir. Çalışmanın organizasyonu maddeler halinde aşağıdaki sırasıyla verilmektedir:

- İkinci bölüm, incelenen gelişmiş sinyal işleme uygulamaları ve geleneksel yöntemler hakkındaki detaylı ve karşılaştırmalı teorik bilgileri içermektedir.
- Üçüncü bölümde, gelişmiş sinyal işleme uygulamalarının hesaplanmasına yönelik akademik çalışmaların detayları ve literatürdeki ilgili çalışmalar anlatılmaktadır.
- Dördüncü bölümde, sayısal alt çevrimi için darboğaz analizi, önerilen güç verimli paralel hesaplama, heterojen mimari tasarımı ve test sonuçları sunulmaktadır.
- Beşinci bölümde, makine öğrenmesi tabanlı hedef tespiti için darboğaz analizi, önerilen güç verimli paralel hesaplama, heterojen mimari tasarımı ve test sonuçları sunulmaktadır.
- Altıncı bölümde, sıkıştırılmış algılama tabanlı geliş açısı kestirimi için darboğaz analizi, CPU ve GPU paralel hesaplama ve test sonuçları sunulmaktadır.
- Yedinci bölümde, gerçek zamanlı entegrasyon akışı, test sonuçları hakkındaki değerlendirmeler ve analizler anlatılmaktadır.
- Son bölümde, tez çalışmasının sunduğu pratik kazanım çıktıları, karşılaşılan zorluklar ve gelecekteki yapılabilecek çalışmalar sunulmaktadır.

2. GELİŞMİŞ SİNYAL İŞLEME UYGULAMALARI

Algılayıcı mimarileri içeren sistemlerde ortamdan algılanan işaretler sayısallaştırılıp veri haline getirildikten sonra çeşitli işleme adımlarına sokulur. Bu algılayıcı mimarileri elektromanyetik (RF), optik ve akustik sinyalleri algılayacak şekilde çeşitlenebilmektedir. RF tabanlı bir algılayıcı sisteminde yer alan anten mimarisi sayesinde elektromanyetik sinyaller algılandıktan sonra analog sayısal çeviriciler sayesinde örneklenerek veri işleme bloklarına sokulmaktadır. Benzer şekilde kameralar optik sinyalleri, mikrofon ve sonar mimarileri ise akustik sinyalleri örnekledikten sonra veri işlemeye sokmaktadır. Tez çalışmasında bir RF tabanlı mimaride yer alabilecek gelişmiş sinyal işleme blokları incelenmekte olup Şekil 2.1'de görüldüğü üzere geleneksel yöntemlerin yerine bu çalışmada teklif edilen gelişmiş sinyal işleme blokları çalışılmaktadır.



Şekil 2.1: Geleneksel Sinyal İşleme Bloklarına karşılık Gelişmiş Sinyal İşleme Uygulamaları

Şekil 2.1'deki akış doğrultusunda olduğu gibi üç sinyal işleme bloğu birbirini takip eden aşamalarda sensör mimarisinde çalışmaktadır. Radar veya elektronik harp sensör mimarisinde bu üç işlem örnek bir akış doğrultusunda sırasıyla işlenmektedir:

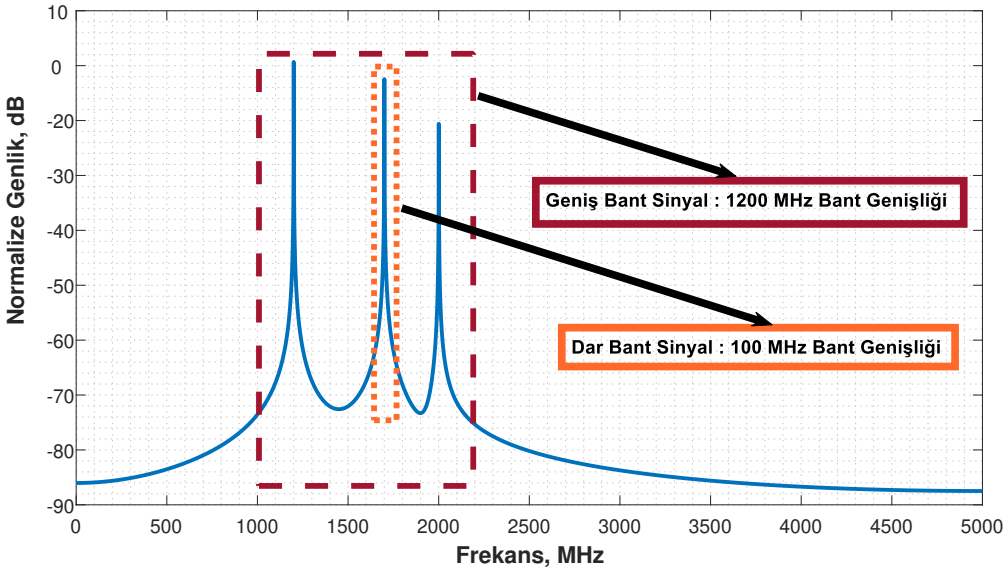
- Geniş bant sayısal veri üzerinde ilgilenilen frekans bölgesi için sayısal alt çevrimi işlemi gerçekleştirilerek örnekleme hızı indirgenmiş veri elde edilir.
- Sayısal alt çevrimi çıktısı dar bant verisi için zaman ve frekans ekseninde oluşturulan matris üzerinde tespit yapılarak hareketli nesnelerin algılaması gerçekleştirilir.
- Tespit edilen her bir hareketli nesne için geliş açısı kestirimi (yön bulma) işlemleri yapılarak nesnenin takibi ve analizleri için açı bilgisi bulunur.

Tez çalışmasında istatiksel yaklaşımlı hedef tespit yönteminin yerine makine öğrenmesi tabanlı hedef tespiti yöntemi incelenmektedir. Ayrıca kovaryans matris hesabına dayanan faz tabanlı ve taramalı geliş açısı kestirimi yerine sıkıştırılmış algılama tabanlı geliş açısı kestirimi (CS-DoA) yöntemi incelenmektedir. Bu iki temel işlem bloğunun üstünde yer alan ve geniş bant sinyali dar bant sinyale çeviren sayısal alt çevrimi (DDC) işlemi de tez çalışmasında incelenen bir uygulamadır. Filtreleme tabanlı DDC yerine paralel programlamaya uygun Kısa Zamanlı Fourier Dönüşümü (Short Time Fourier Transform, STFT) ile DDC işleminin kullanımı anlatılmaktadır.

2.1. Sayısal Alt Çevrimi

Sayısal Alt Çevrimi işlemi (DDC) sinyal işleme alanında filtreleme amacıyla kullanılır. Geniş bant (GB) bir sinyal detaylı analiz, tespit, geliş açısı kestirimi işlemleri için dar bant bir sinyale dönüştürülmektedir. DDC işlemi bu dönüşümü sağlayarak dar bant (DB) IQ (Inphase-Quadrature) verinin elde edilmesini sağlar. DDC işlemi, geniş bant spektrumda tespit edilen ya da istenen bir merkez frekansa ait işareti belirli bir bant genişliğinde filtrelemektedir. Şekil 2.2'de DDC işlemini görsel olarak anlatan bir sinyal grafiği yer almaktadır. 1200 MHz bant genişliğinde olan geniş bant sinyali 100 MHz bant genişliğine sahip olacak bir dar bant sinyale dönüştürmek için DDC işlemi yapılır.
DDC işlemi sonucunda dar bant sinyal verisi üzerinde radar ve elektronik harp uygulamaları gerçekleştirilir. DDC işlemi elektronik harp uygulamalarında sinyal teşhisi, sınıflandırma ve

karar işlemlerinin yapılması için gerekli dar bant sinyalin oluşmasını sağlar. Özellikle yazılım tanımlı telsiz (SDR) uygulamalarında çok kanallı gerçek zamanlı DDC yapılabilmesi kaliteli bir haberleşme uygulamasının gerçekleşmesine katkı sağlar. Radar almacından analog olarak elde edilen sinyalin taşıyıcı frekansı ara frekans (IF) değerine kaydırılır. IF frekans etrafında yapılan örnekleme ve DDC işlem adımları sonucu taban-bantta varsayılan bant genişliğine sahip sayısal IQ verisi oluşturulur. Filtrelenmiş bu taban-bant sayısal sinyal üzerinde sinyal işleme blokları gerçekleştirilir. Tez çalışmasında belirtilen hedef tespiti ve geliş açısı kestirimi işlemleri filtrelenen bu verinin işlem akışında gerçekleştirilir.



Şekil 2.2: Geniş Bant - Dar Bant Filtreleme için Spektrum Gösterimi

DDC işleminin gerçek zamanlı ve yüksek başarımlı yapılması sensör sinyal işlemenin gerçek zamanlı çalışma prensibine olumlu etki etmektedir. Son yıllarda bilişsel radar ve elektronik harp uygulamalarında sayısal sinyal işleme bloklarının yüksek başarımlı ve çok girdi-çok çıktı kanallı bir yapıda işlenmesi beklenmektedir. Tez çalışmasının günümüzde revaçta olan uç hesaplama mimarilerinde etkin algoritma hesaplaması tasarımlarına ön fikir sunması öngörülmektedir. Geleneksel yöntemlere karşın FFT tabanlı bir DDC yöntemi ile GPGPU'larda paralel hesaplama için daha uygun bir algoritma tasarlayarak özellikle çok kanallı dar bant sinyal işlemeye avantaj kazandırmak tez çalışmasında amaçlanmıştır. Bu bölümde geleneksel algoritma ile FFT tabanlı DDC işleminin ayrıntıları ve karşılaştırmaları aktarılır.

2.1.1. Filtreleme tabanlı DDC

DDC işleminde literatürde sıkça karşılaşılan yöntem, kombine entegre taranmış (CIC) filtre kullanımı ile alt örnekleme (downsampling) işlemleri yapılarak karmaşık dar bant sinyal elde edilmesidir. CIC filtre kullanımı için FPGA mimarisinde özelleşmiş kütüphaneler ve sayısal devreler vardır. CIC filtre sayısal devreler için bir blok akış şeması içerdiğinden, FPGA gibi çok sayıda koşut çekirdeklerin olduğu bir mimari için daha uygun olmaktadır. Ayrıca CIC filtre kullanımın GPU üzerinde uygulaması yapılarak GPU tabanlı geniş bant SDR uygulaması için kullanım örnekleri verilen çalışmalar mevcuttur [3, 4]. Fakat bu çalışmalarda CIC filtre işleminin GPU birimleri için verimli olmadığından bahsedilmektedir. Filtreleme işlemi grafik işlemci mimarilerinde yapılırken veri bağımlılığı içeren bir işlem olduğu için paralel hesaplama yapılması zorlaşmaktadır. Ardışık düzen bir işlemci yapısında ancak filtreleme işlemi avantajlı hale gelmektedir. Bu işlemci mimarileri genellikle çok sayıda DSP işlemci kaynağı içeren FPGA-DSP yapıları ile mümkün olmaktadır.

Filtre uzunluğu arttıkça kaynak kullanım sayısı artmakta bu durumda çok kanallı bir sensör mimarisinde darboğazlar ile karşılaşılmaktadır. Ayrıca çoğu DDC uygulamasında CIC filtre işleminden sonra filtreleme sırasında oluşabilecek gürültü bileşenleri elemek için alçak geçirgen filtre (low pass filter, LPF) işlemi uygulanır.

2.1.2. STFT tabanlı DDC

Filtreleme işleminde veri bağımlığının bulunması, filtreleme modüllerin doğrusal evrişim (konvolüsyon) blokları ile yapılması ve CIC gibi filtreleme işlemlerinin FPGA-DSP mimarilerine uygun olması, DDC işleminin CPU ve GPU paralel hesaplama başarımlarını kısıtlayan bir duruma neden olmaktadır. DDC işlemi için en büyük hesaplama darboğazı geniş banttan dar bant sinyale çevrim sırasında yapılan filtreleme işlemidir. Bu işlem için kullanılan doğrusal evrişim yöntemi $O(n^2)$ hesaplama karmaşıklık değeri ile çalışır [5]. CIC ile alt filtreleme işlemlerinde daha küçük filtre uzunluğu ile bu hesaplama karmaşıklığı düşürülmeye çalışılmaktadır. Veri uzunluğu veya filtre uzunluğu arttıkça hesaplama ihtiyacı da artmaktadır.

Yüksek bant genişliğine sahip radar, elektronik harp ve optik sensörlerde birim zamandaki örnek sayısı oldukça fazla olmaktadır. Örneğin, 100 MHz bant genişliğine sahip bir geniş bant sinyal Nyquist Teoremi'ne göre 200 MHz örnekleme hızına sahip olmaktadır [6]. 16 bit örnekleme yapan bir sistemde saniyede 6.4 Gbit uzunluğunda IQ veri örneklenmektedir. 5G ve ötesi haberleşme, bilişsel radar gibi sistemlerde daha yüksek bant genişlikleri olabileceği dikkate alındığında örnek miktarı daha fazla olabilmektedir. Filtreleme yönteminde daha düşük hesaplama karmaşıklığına sahip algoritma yapıları seçilerek daha hızlı çalışabilen DDC uygulamasının tasarımı gerçekleşecektir. Filtreleme işlemi zaman uzayı yerine frekans uzayında yapılırsa doğrusal evrişim işlemi vektör çarpım işlemine dönüşür.

Çarpım işleminin gerçekleşmesi için öncelikle zaman uzayından frekans uzayına dönüşüm yapılması gerekir. Bu dönüşümü Hızlı Fourier Dönüşümü (FFT) sayesinde yapmak mümkündür [7]. FFT tabanlı DDC işlemi için teorik alt yapı incelenmeden önce filtreleme tabanlı bir yaklaşımda Eşitlik 1'de görüldüğü üzere bir evrişim işlemi yapılmaktadır. Girdi vektörünün (x) uzunluğu N, filtre vektörünün (h) uzunluğu M olan bir evrişim işleminde çıktı vektörünün (y) uzunluğu $N + M - 1$ olur. Eşitlik 1'de görülen indisler n ve m sırasıyla girdi ve filtre vektörlerinin indislerini temsil etmektedir. Fakat evrişim işleminde filtre gecikmesi olduğu için çıktı vektörü üzerinde başlangıç kısmından $M/2$ ve bitiş kısmından $M/2 - 1$ kadar veri atılır. Böylece, çıktı uzunluğu girdi uzunluğu ile aynı hale getirilir.

$$\begin{aligned} &y_n = x_n * h_m, y[n] = \sum_{m=0}^{M-1} x[m]h[n-m] \\ &n = 0 : N-1, m = 0 : M-1. \end{aligned} \tag{1}$$

DDC işleminde filtre katsayılarının filtre tasarımı sonrası sabit tutulacağı varsayılır. Böylece işleme başlamadan önce filtre katsayıları için bir kez FFT işlemi yapılarak FFT tabanlı DDC işleminde bir optimizasyon yapılır. FFT uzunluğu ile aynı uzunlukta olması için aradaki uzunluk farkı kadar filtre katsayılarının sonuna sıfır değeri eklenir. DDC işleminin girdi verisi zamanla değişeceği için algoritmaya giren her girdi bloğunun FFT işlemi gerçekleştirilir.

Eşitlik 2'de DDC sırasındaki filtreleme işleminin FFT dönüşümü ile yapılmasının matematiksel gösterimi sunulmaktadır. Eşitlik 2'te FFT uzunluğu N alınmıştır.

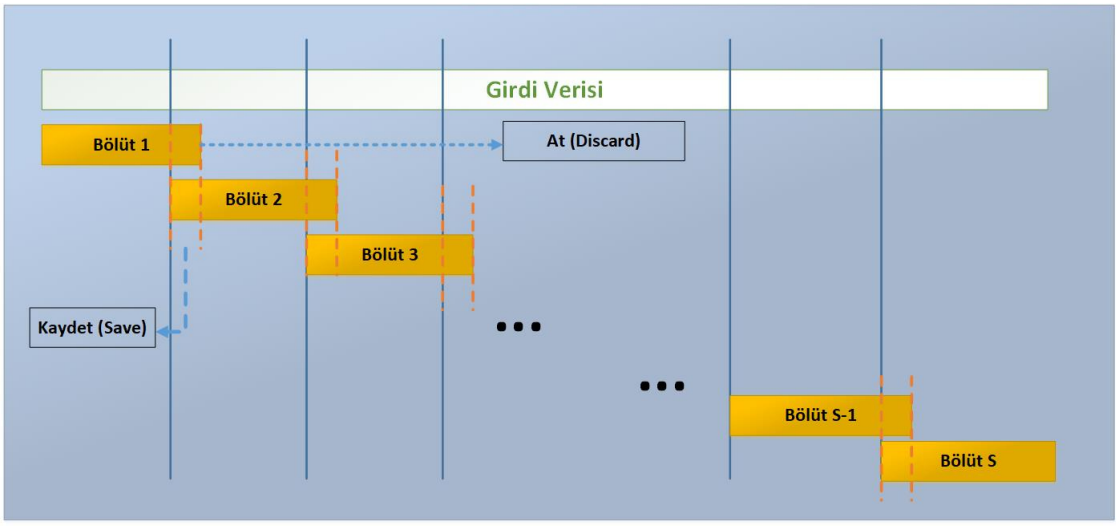
$$\begin{aligned} &X[k] = FFT\left\{x[n]\right\}, H[k] = FFT\left\{h[n]\right\}, \\ &Y[k] = X[k] \cdot H[k], \\ &y[n] = IFFT\left\{Y[k]\right\}. \end{aligned} \tag{2}$$

Eşitlik 2'de görüldüğü üzere frekans uzayında çarpım işleminden sonra zaman uzayına tekrar geçmek için ters FFT (IFFT) işlemi gerçekleştirilir. Eşitlik 2'de doğrusal evrişim işlemi yerine 1 adet FFT, 1 adet IFFT, vektör çarpımı ve 1 adet önceden alınmış FFT ile DDC filtreleme işlemi gerçekleştirilir. Gerçekleşen bu algoritma iyileştirmesi sayesinde hesaplama karmaşıklığı asimptotik olarak $O(nlogn)$'e düşer [5]. Çizelge 2.1'de girdi uzunluğuna göre iki algoritmanın hesaplama karmaşıklığının karşılaştırılması gösterilmektedir.

Girdi Uzunluğu (FFT Uzunluğu, n)	**Doğrusal**, $O(n^2)$	**FFT tabanlı**, $O(nlogn)$	**Hızlanma**
4	16	8	**2x**
8	64	24	**2.67x**
16	256	64	**4x**
32	1024	160	**6.4x**
64	4096	384	**10.67x**
128	16384	896	**18.29x**
256	65536	2048	**32x**
512	262144	4608	**56.89x**
1024	1048576	10240	**102.4x**

Çizelge 2.1: Doğrusal ile FFT tabanlı Evrişim Yönteminin Hesaplama Karşılaştırmaları

Büyük uzunluktaki FFT işlemleri hem CPU hem de GPU'da önbellek kapasitesi, yazmaç boyutları ve maksimum görev (thread) sayısı gibi kısıtlı donanımsal kaynaklar nedeniyle uzun bir işlem süresinde bitmektedir. FFT işlemlerini daha küçük segmentlere bölerek örtüşen kaydet/at (overlap save/discard) FFT ile kısa süreli Fourier dönüşümü (STFT) yöntemi kullanılarak daha verimli bir hesaplama mümkündür. Bu tez çalışmasında STFT ile büyük uzunluktaki FFT işlemi küçük segmentler ile yapılarak paralel hesaplamadaki başarım arttırılmaktadır. Şekil 2.3'te STFT ile ilgili bir görsel anlatım sunulmaktadır.



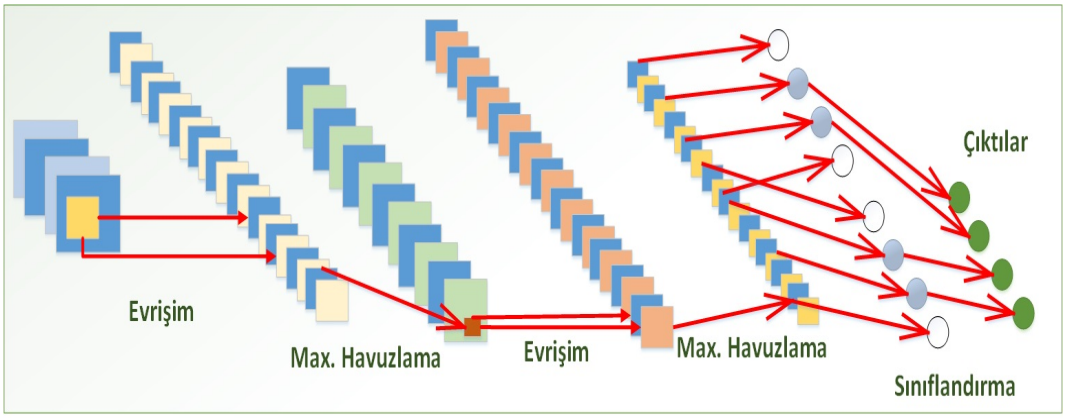
Şekil 2.3: Büyük Verinin Örtüşen Kaydet/At tabanlı STFT ile Bölütlenmesi

Büyük uzunluktaki veri bölüt (segment) parçalarına ayrıldıktan sonra girdi uzunluğuna göre daha küçük FFT boyu ile işlem gerçekleştirilir. Her bir bölüt uzunluğu alt filtreleme işlemi yapılarak bölütler arası aktarım örtüşen kısımların kaydedilmesi ve fazlalık kısımların atılması ile gerçekleştirilir. Filtreleme işlemi sonrasında alt örnekleme oranı kadar veri indirgenir. Böylece DDC işlemi tamamlanır. STFT tabanlı DDC işlemi GPGPU'lara uygun hale getirildiği için paralel hesaplama ile başarımının daha fazla artması öngörülmektedir. Çok kanallı DDC işlemine ihtiyaç duyan yüksek bant genişliğine sahip sensör sistemlerine uyarlanabilecek bir hesaplama yöntemi sunulmuştur.

2.2. Makine Öğrenmesi tabanlı Hedef Tespit İşlemi

Derin öğrenme, makine öğrenmesi, örüntü tanıma ve yapay zekâ çalışmaları tarihsel gelişimleri ile birlikte nesne teşhisi, tespiti ve sınıflandırma problemlerine önemli çözüm alternatifleri sunmuştur [8]. 21. yüzyılda işlemci teknolojilerinin gelişmesi ile birlikte kayda değer bir teknolojik başarım gerçekleşmiştir. Derin öğrenme ve makine öğrenmesi çalışmaları bilgisayarlı görü, doğal dil işleme, robotik, ses tanıma ve nesne tespiti problemlerinin çözümüne öznitelik öğrenimi ve otomatik sınıflandırma yaklaşımları ile uyarlamalı çıktılar sunar [9].

Tez çalışmasında, öncesinde geri yayılımlı (back propagation) öğrenmesi yapılmış ve katsayı-ofset seti hazırlanmış bir model ile ilgilenilmektedir [1]. Model, Evrişimsel Sinir Ağı (CNN) tabanlı bir model olup Şekil 2.4'te tipik CNN modeline ait sinir ağı akışları yer alır [10].



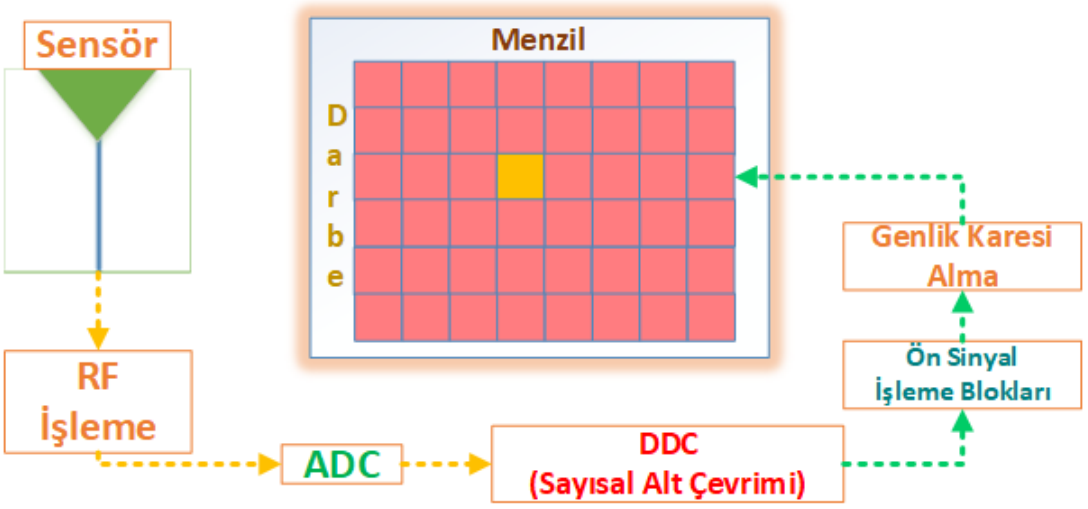
Şekil 2.4: Jenerik CNN Modeli

Tez çalışmasında, radar sensör sistemlerinde hareketli hedef tespiti için kullanılan Sabit Yanlış Alarm Oranı (SYAO) tekniğindeki yapıya benzer bir CNN modeli kullanılmaktadır [11]. Çalışmada, öznitelik öğrenen CNN temelli hedef tespiti işlemini esas alan bir hesaplama problemi çalışılmaktadır [12]. CNN temelli hedef tespiti işlemini analiz edebilmek adına öncesinde SYAO yöntemi detaylıca Bölüm 2.2.1.'de anlatılmaktadır.

2.2.1. İstatiksel Hedef Tespit Yöntemi

Ön işlem blokları tamamlanan bir RF sensör mimarisinde zaman ve frekans alanında örneklenen sinyal verileri içinde hedeflerin otomatik olarak tespit edilebilmesi için bir girdi matrisi oluşturulur. Bu matris eğer bir radar sistemindeki hedef bilgisini tespit ediyorsa eksenleri tanım olarak farklılaşır.

Zaman eksenindeki matris hücreleri hedeflere ait mesafe (menzil) bilgilerini ifade ederken, frekans eksenindeki hücreler ise Doppler hızı değerlerini temsil eder. Menzil ve Doppler bilgilerine ait sinyal şiddetini içeren bu matrise Menzil Doppler/Darbe (MD) matrisi de denilmektedir. Şekil 2.5'te MD matrisinin analog sinyalden sırasıyla nasıl oluştuğunu gösteren bir akış şeması sunulmaktadır.

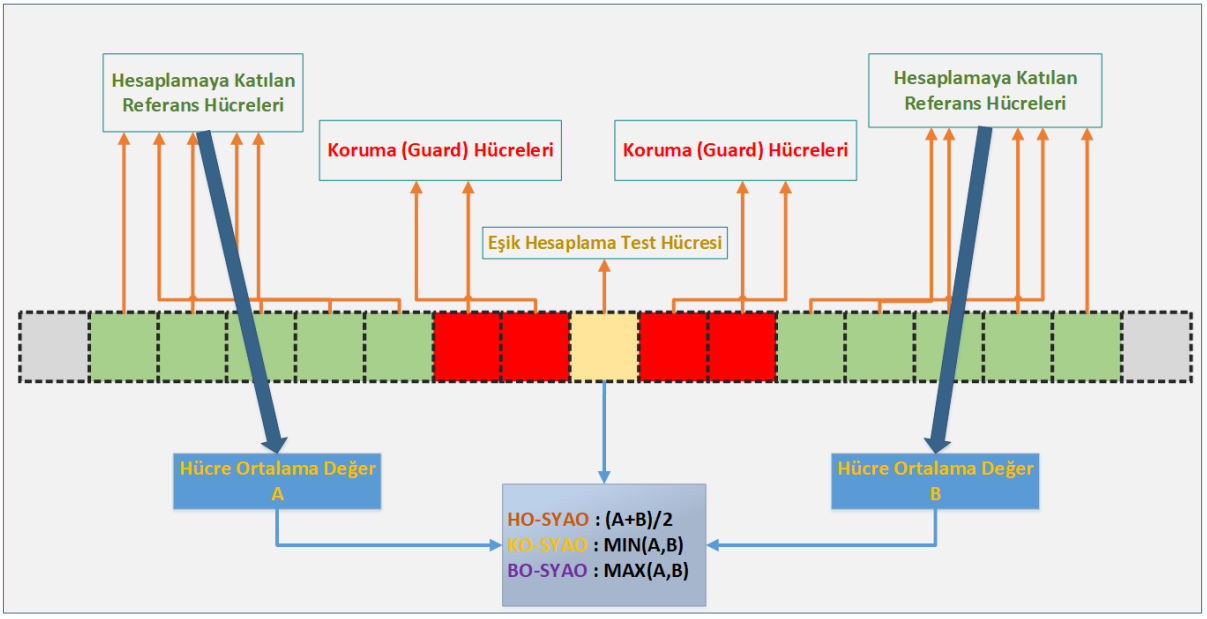


Şekil 2.5: Menzil-Darbe Matrisi Oluşumu

MD matrisinde hedef tespitinde geleneksel bir işlem adımı olan SYAO tekniği kullanılır. SYAO tekniği ortamda sabit bir yanlış alarm olasılığı varsayımıyla tek boyutta ya da iki boyutta tarama yaparak eşik hesaplar [11]. Eşik işleminde test hücresinin komşu hücreleri belirli bir koruma bandı koyularak hesaba katılır. Hücre ortalamalı SYAO (HO-SYAO), hücre ortalaması küçük olan SYAO (KO-SYAO) ve hücre ortalaması büyük olan SYAO (BO-SYAO) gibi çeşitli alternatif yöntem, ortam koşullarına göre tespit yöntemi olarak seçilmektedir [11].

Tez çalışmasında HO-SYAO tekniği ile benzer tarama yapan CNN modeli kullanıldığı için HO-SYAO eşik yöntemi dikkate alınır [1]. Şekil 2.6'da SYAO tekniğindeki bir boyutta (1B) olan hesaplama yaklaşımı anlatılmaktadır. Bu yaklaşım iki boyutta da kullanabileceği gibi ön fikir sunması açısından destekleyici bir gösterim sunmaktadır. SYAO tekniği ile hem menzil hem de Doppler boyutunda iki boyutlu(2B) bir şekilde eşik hesabı yapılarak hedefin menzil ve Doppler eksenlerdeki yayılımı bertaraf edilmiş olur [11].

Hesaplanan eşik değerleri sonucunda MD matrisi ile eşik değerleri karşılaştırılıp eşiği geçen hücreler hedef tespiti olarak kabul edilir. Her bir tespite ait menzil ve Doppler hızı çıktı olarak raporlanır. Raporlanan bu değerler açı kestiriminde kullanılacağı için saklanır. SYAO tekniğindeki adaptif eşik hesabı sadece radar ve elektronik harp sistemlerinde değil istatiksel hedef tespit yapılan tüm sensör sistemlerinde kullanılmaktadır [13].



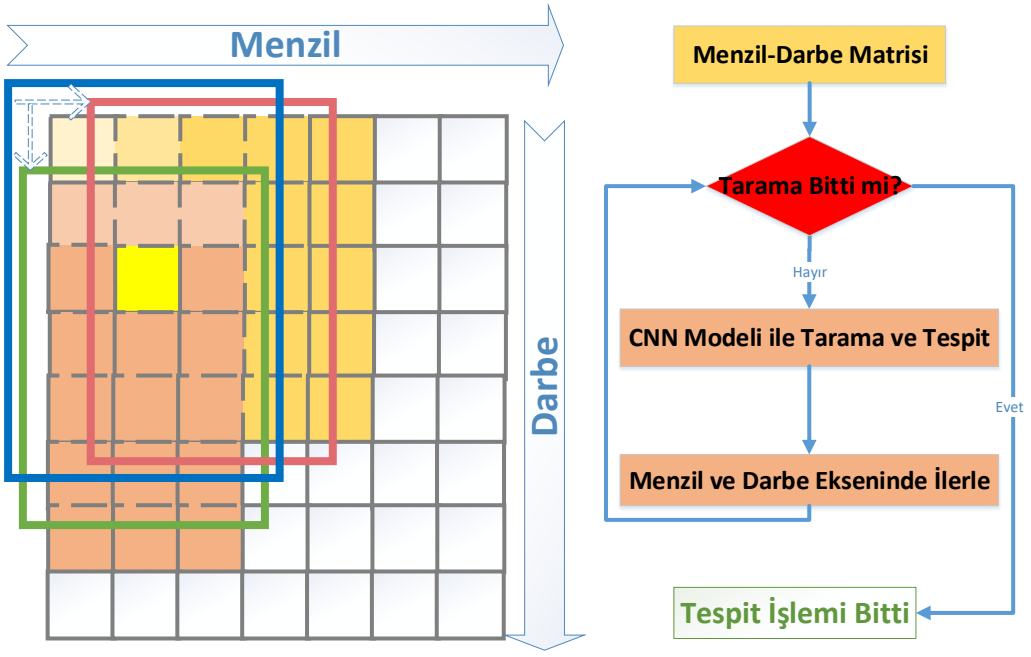
Şekil 2.6: Menzil Boyutunda SYAO Tekniği ve Hesaplama Hücre Bilgileri

2.2.2. Makine Öğrenmesi Uygulaması

SYAO tekniğindeki işlemleri öğrenen ve eğitilmiş bir CNN modeli ile MD matrisi üzerinde ileri yayılımlı işlem yapılarak hareketli hedef tespiti varlığı bulunur [1]. CNN modeli ile yapılan hareketli hedef tespiti işlem akışı Şekil 2.7'de ve CNN modelinin akış şeması Şekil 2.8'de gösterilmektedir. Her bir alt matrisin modelle etkileşimi sonrası bir adet test hücresi için eşik kararı verilir. Eşik kararı verildikten sonra hedef varlığı tespit edilir. Gerçekleşen her bir işlem adımı alt matris üzerinde katmanlar halinde yapılarak maddeler halinde belirtilmektedir:

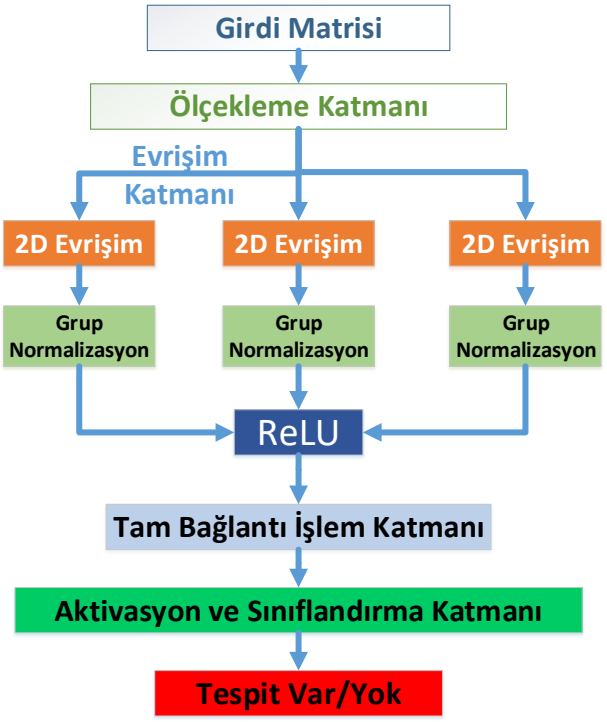
- **Min-Max Ölçekleme Katmanı** :Alt matris boyutunda MD matrisinden bir parça seçilir. Bu katman sayesinde matris istenen veri aralığına ayarlanır.
- **Çok Kanallı Evrişim Katmanı** : Ölçeklenen 2B matris üzerinde evrişim işlemi yapılır. Evrişim işleminden sonra her bir kanal için ofset değerleri ile düzeltme yapılır.
- **Grup Normalizasyon Katmanı** : Evrişim işlemi sonucunda elde edilen her bir çıktı kanal matrisi için skala değerleri ile normalizasyon yapılır.
- **Negatif Kırpma (ReLU) Katmanı** : Modeldeki negatif olan değerler kırpılır.

- **Tam Bağlantı İşlem Katmanı** : ReLU sonrası kanal verileri vektöre dizildikten sonra matris-vektör çarpımı gerçekleşir.
- **Aktivasyon ve Sınıflandırma Katmanı** : Tam Bağlantı İşlem Katmanı'nın çıktısı için aktivasyon fonksiyonu çalıştırılır. Aktivasyon sonucu eşikten geçirilerek tespit kararı verilir.



Şekil 2.7: Hareketli Hedef Tespiti için CNN Modeli ve Tarama Akışı [1]

CNN modeli sayesinde özellikle gürültüye karşı düşük sinyal şiddeti durumlarında SYAO tekniğine göre daha doğruluk başarımlı hedef tespit yapılırken hesaplama ihtiyacı arttığı için gerçek zamanlı sensör uygulaması için zorluk oluşmaktadır [1]. Çizelge 2.2'de hedef tespiti için kullanılan yöntemlerin hesaplama gereksinimleri görece karşılaştırmalı olarak belirtilmektedir. Çizelge 2.2'de MD imge matrisi, 3000 adet menzil hücresinden 16 adet de Doppler hücresine sahip bir boyutta varsayılmıştır. SYAO tekniğindeki referans hücresi sayısı menzil ekseninde 20, Doppler ekseninde 2 iken koruma hücre sayısı menzil ekseninde 4, Doppler ekseninde 2 olarak belirlenmiştir. Hesaplama karmaşıklığı kavramı olarak çarpma-toplama sayısı (multiply-accumulate, MAC) kullanılmıştır. Çizelge 2.2'de görüldüğü üzere CNN modeli ile hedef tespiti yapılırken yaklaşık 23x fazla hesaplama ihtiyacı vardır.



Şekil 2.8: CNN Modeli İşlem Katmanları [1]

Tespit Yöntemi	HO-SYAO	KO-SYAO	BO-SYAO	ML-CNN
Gereken İşlem Sayısı (MAC)	$57.6x10^5$	$58.1x10^5$	$58.1x10^5$	$13.2x10^7$
Görece Hesaplama Artışı	$1x$	$1.009x$	$1.009x$	**22.719x**

Çizelge 2.2: Hedef Tespit Yöntemleri Hesaplama Karmaşıklığı Karşılaştırmaları [1]

2.3. Konveks Optimizasyon tabanlı Geliş Açısı Kestirimi İşlemi

Geliş açısı kestirimi için kullanılan bir anten dizisinde açı noktalarını içeren ızgara yapısı tasarlanmakta ve bu ızgara sayesinde tespit noktaların yoğunluk oluşturduğu açı noktaları alternatif kestirim yöntemleri ile belirlenmektedir. Bu tez çalışmasında geleneksel kovaryans matris hesabı ile geliş açısı kestirimi yapan MUSIC [14] ve gelişmiş bir sinyal işleme uygulaması olan sıkıştırılmış algılama tabanlı konveks optimizasyon içeren LASSO [15] geliş açısı kestirimi işlemleri incelenmektedir.

Konveks optimizasyon yaklaşımı ile açı kestirimi problemi bir eniyileme problemine dönüştürülerek sonuç doğruluğunun başarımı arttırılmaktadır. Genel olarak optimizasyon yaklaşımı belirli bir aralıkta istenen eşitliği ya da ilişkiyi sağlayan değerleri bulmak için yapılan sistematik bir çözümlemeyi içermektedir. Örneğin, istenen bir aralığın içerisinde global minimum ve maksimumları bulmak için bir optimizasyon problemi çözülmektedir. Optimizasyon çözümleri finans, tahminleme, kaynak yönetimi ve nümerik analiz gibi birçok alanda tercih edilmektedir. Geleneksel yöntemlere göre doğruluk başarımı artmasına rağmen iterasyonel akış için yakınsama durumuna kadar tekrarlanan işlem adımları nedeniyle hesaplama yükü artar.

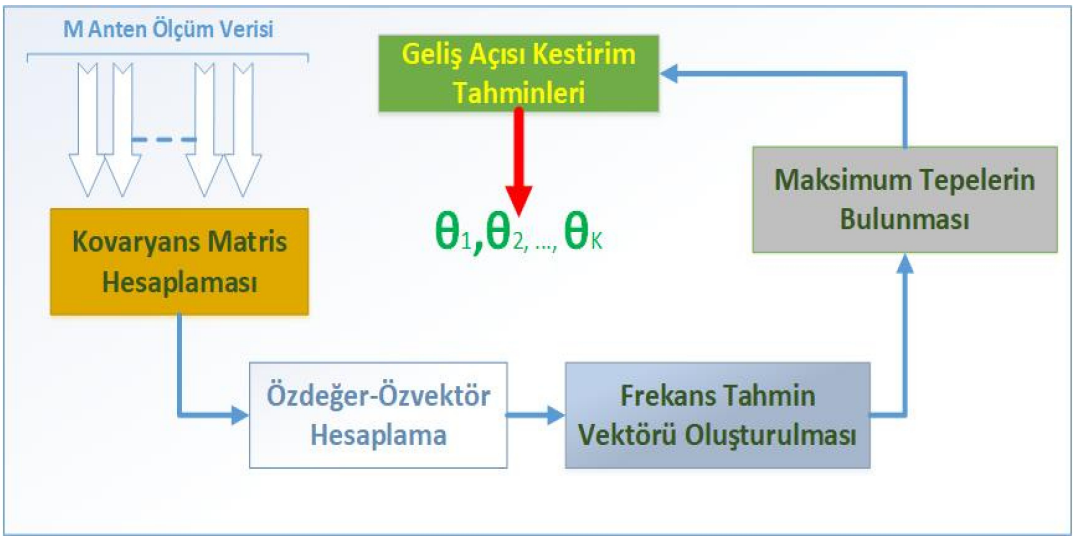
Tez çalışmasında, konveks optimizasyon ile yapılan geliş açısı kestirimi çözümünün sisteme getirdiği yüksek hesaplama yüküne rağmen gerçek zamanlı çalıştırabilmesi için uygulanan paralel hesaplama yaklaşımları incelenmektedir. Bu bölümde ise geleneksel yöntem ile gelişmiş yöntemin açıklamaları verilmekte olup hesaplama karmaşıklıkları karşılaştırılmaktadır.

2.3.1. Çoklu Sinyal Sınıflandırma (MUSIC)

Belirli bir ölçüm sayısı kadar biriktirilen ölçüm verileri kendi devrik matrisi ile çarpılıp ölçüm sayısı ile normalize edildiğinde Kovaryans Matrisi elde edilir. Anten sayısı kadar uzunluğundaki kenar değerleri boyutuna sahip geliş açısı kestirimine girdi olan Kovaryans Matrisi ile özdeğer ve özvektör çıkarımı yapılır. Özdeğerlere bağlı olarak seçilen özvektör matrisi ile frekans tahmini fonksiyonu oluşturulduktan sonra istenen hedef sayısı kadar maksimum tepeler bulunur [16]. Bu tepeler önceden belirlenmiş açı noktaları ile kesiştirilerek geliş açısı kestirimi yapılır. Şekil 2.9'da MUSIC işlem akışına ait bir şema görülmektedir.

Geleneksel bir yöntem olan MUSIC gürültülü ortamda başarılı sonuç vermesine rağmen çok yollu (multipath) bir senaryoda kestirim başarım oranı düşmektedir [17]. Başarımı arttırmak için daha fazla ölçüm alınarak Kovaryans Matrisi'nin daha kararlı bir hale getirilmesi sağlanır. Ölçüm sayısı arttıkça geliş açısı kestirimi hesaplamasında gecikme (latency) artacağı için sensör sistemine olumsuz bir etki olmaktadır. Geliş açısı kestirimi MUSIC dışında Barlett [18] ve MVDR [17] yöntemleri ile de yapılabilmektedir. MUSIC yöntemi bu iki yönteme

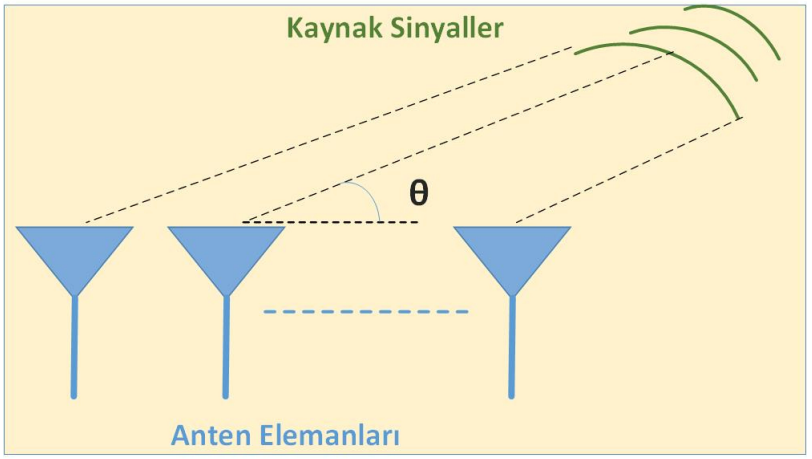
göre fazla hesaplama karmaşıklığına sahip olmasına rağmen çoğu ortam koşullarında daha yüksek kestirim başarımı sağlamaktadır [2].



Şekil 2.9: MUSIC Geliş Açısı Kestirimi İşlem Akış Şeması

2.3.2. Sıkıştırılmış Algılama tabanlı DoA

Çoğu mühendislik uygulamasında, temel ilgi, birçok bilinmeyenli ve daha az denklemli doğrusal bir sistemi çözmektir. Bu tür sistemlerde sonsuz sayıda çözüm sistemi karşılayabilir. Temel amaç bu çözümlerden en iyisini bulmaktır. Genel olarak, minimum enerjiye sahip olan çözüm seçilir. Sıkıştırılmış algılama, seyrek sinyalleri daha az sayıda ölçüm ile çözebilen bir sinyal işleme tekniğidir [19]. Çoğu doğal sinyal bazı dönüşüm alanlarında seyrek olduğu için birçok alana uygulanabilir. Bununla birlikte, en seyrek çözümü bulma matematiksel olarak kolayca ifade edilememektedir. Bu durumu aşmak için sıkıştırılmış algılama ile iyi bilinen bir seyrekleştirme algoritması olan ℓ_1-norm minimizasyonu kullanmak önerilmektedir.
Sıkıştırılmış algılama tabanlı geliş açısı kestirimi tahmini, iyi bilinen bir araştırma alanıdır. Tipik olarak Şekil 2.10'da görüldüğü üzere, sensör dizileri, sinyallerin her bir sensöre farklı zamanlarda ulaşması ve sensörler arasında hedeflerin uzamsal varyasyonunun ölçülmesini sağlayan bir faz değişimine neden olması etkisiyle geliş açısı kestiriminde kullanılır.



Şekil 2.10: Geliş Açısı Kestirimi için Temel Senaryo Gösterimi

Gelişmiş geliş açısı kestirimi uygulaması için ℓ_1-norm minimizasyonu esas alan konveks optimizasyon çözümüyle hata enerjisinin minimize edilmesini amaçlayan En Küçük Mutlak Büzülme ve Seçim Operatorü (Least Absolute Shrinkage and Selection Operator, LASSO) yöntemi ile seyrek geriçatım yapılır [15]. Alternatif Yön Çarpanlar Yöntemi (Alternating Direction Method of Multipliers, ADMM) ile yakınsama problemini daha küçük bölütlerle çözerek daha hızlı kestirim sonucu bulmak mümkündür. Tez çalışmasında DoA tahmini için kullanılan LASSO çözümünün ADMM yöntemi ile daha yalın hale getirilmiş bir alternatifi temel alınmıştır [20]. Seyrek geriçatımda algoritma başarımı artıran ADMM yöntemi için daha önce SAR radar görüntülemede de kullanılmış C-SALSA çözümü kullanılmıştır [21]. LASSO ile geliş açısı kestirimi yapmak kestirim doğruluğunu artırsa da hesaplama yükü açısından MUSIC yöntemine göre daha fazla hesaplama yükü getirmektedir.

Geliş Açısı Kestirimi İşlem Sayısı (MAC)		*MUSIC*	*ADMM tabanlı DoA*
M = 10	N = 10	$5.23x10^4$	$0.73xKx10^7$
	N = 100	$6.62x10^4$	$0.74xKx10^7$
M = 32	N = 10	$8.15x10^5$	$0.89xKx10^7$
	N = 100	$9.55x10^5$	$0.9xKx10^7$
M = 128	N = 10	$3.34x10^7$	$1.6xKx10^7$
	N = 100	$3.56x10^7$	$1.61xKx10^7$

Çizelge 2.3: MUSIC ve ADMM tabanlı DoA Hesaplama Karmaşıklığı Karşılaştırmaları [2]

Çizelge 2.3'te ADMM tabanlı LASSO ile MUSIC geliş açısı kestirimleri için hesaplama gereksinimleri sunulmaktadır. K hedef sayısını belirtmektedir. ADMM tabanlı DoA yönteminde hedef sayısı ile doğrusal artış kaydeden bir hesaplama ihtiyacı olduğu Çizelge 2.3'te görülmektedir. Hesaplama karmaşıklığının artması doğal olarak gerçek zamanlı işlem yapmayı zorlaştırmaktadır. ADMM tabanlı DoA tahmini için çalışmamızda kullanılan çözümü daha ayrıntılı aktarmak açısından geri-çatımı içeren sinyal modeli, ADMM çözümüne ait alt işlemleri ve ADMM iterasyon analizi sırasıyla bu bölümde belirtilmektedir.

Sinyal Modeli : Doğrusal, izotropik ortam ve izotropik anten elemanları ile tek biçimli doğrusal dizi (uniform linear array, ULA) varsayıldığında, ek gürültü altında aynı taşıyıcı frekansında yayın yapan dar bant nokta kaynakları için anlık görüntü indeksi k ile geleneksel DoA sinyal modeli Eşitlik 3'teki gibi yazılır. Sensör dizisi tarafından alınan sinyal x, geliş açılarına bağlı yönlendirme matrisi A, bilinmeyen geliş açılarına sahip kaynaklardan iletilen sinyal s olarak ifade edilir. n ise $n \sim N_c(0, \sigma_n^2 I)$ dağılımına sahip dairesel simetrik toplamsal karmaşık Gauss gürültüsü sinyalidir.

K tane hedef ve *M* tane anten elemanı varsayımı ile x, *M* uzunluğunda bir vektör, s, *K* uzunluğunda bir vektör, n, *M* uzunluğunda bir vektör ve $A(\theta)$ ise *MxK* boyutunda bir matris şeklinde doğrusal cebir formunda ifade edilmektedir. İzotropik sensörlü bir ULA yapısını varsaydığımız için $A = [a(\theta_1)\ a(\theta_2)\ \dots\ a(\theta_K)]$ ve $a(\theta) = [1\ e^{j\omega(\theta)}\ \dots\ e^{j(M-1)\omega(\theta)}]^T$ eşitlikleri yazılır.

$$x(k) = A(\theta)s(k) + n(k) \tag{3}$$

Sıkıştırılmış algılama tabanlı DoA sinyal modeli Eşitlik 3'te verilen geleneksel DoA sinyal modelinden farklıdır. Sıkıştırılmış algılama tabanlı DoA sinyal modelinde hedeflerin geliş açılarını belirlemek için bir sözlük kullanılır. Sözlük, önceden tanımlanmış *L* tane açı ızgara noktalarından gelen hedef sinyallere muhtemel tüm sensör yanıtlarını içerir. Bu yanıtlar $a(\bar{\theta}_1), a(\bar{\theta}_2), ..., a(\bar{\theta}_L)$ şeklinde ifade edilebilir. Eşitlik 4'te olduğu gibi bir D sözlük matrisi ile model oluşturulabilir. D ifade edilmek istenirse $D = [a(\bar{\theta}_1)\ a(\bar{\theta}_2)\ \dots\ a(\bar{\theta}_L)]$ şeklinde açılımı yapılabilir. D matrisinin Eşitlik 3'teki A'dan farklı olduğu açıktır. Eşitlik 3'te, *K* hedeflerinin geliş açıları bilinmediğinden A'ın bilinmediği varsayılır. Ancak, Eşitlik 4'te D bilinen sözlüktür, çünkü *L* ızgara noktalarını hedeflerin (yaklaşık olarak) bu ızgara noktaları

üzerinde yer aldığı varsayımıyla belirlenmektedir. Eşitlik 4'teki x^G ve s^G, Eşitlik 3'te sırasıyla x ve s'ın ızgaralı sürümlerini temsil eder. x^G, M uzunluğunda bir vektör ve s^G ise L uzunluğunda bir vektör ile matematiksel ifade edilir.

$$x^G(k) = Ds^G(k) + n(k) \tag{4}$$

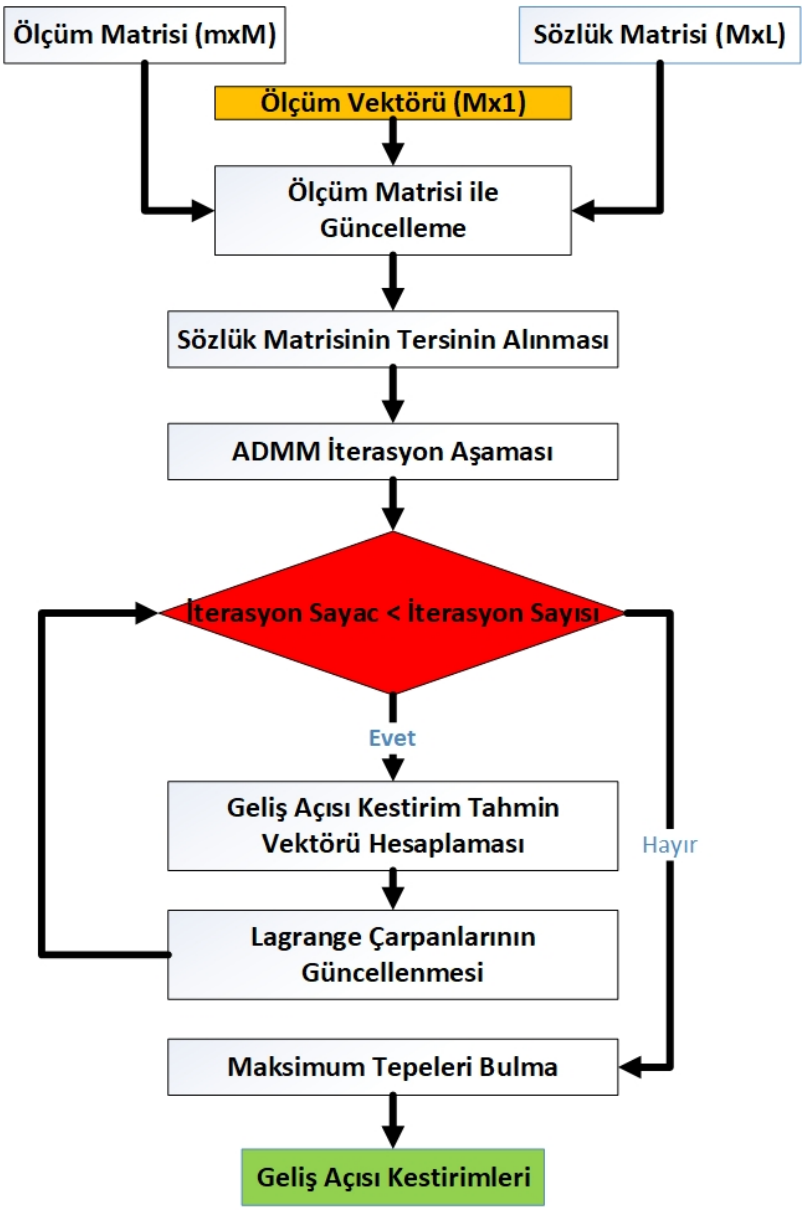
Sıkıştırılmış algılama tabanlı DoA tahmini yönteminin arkasındaki ana sezgi, Eşitlik 5'te belirtilen ℓ_1-norm minimizasyon problemini çözerek geliştirilebilecek olan s^G'nin seyrekliğini kullanmaktır. ε, istatistiksel özellikleri biliniyorsa ortamın gürültü özellikleri kullanılarak seçilebilen bir parametredir. Kısacası, DoA tahmini, aşağıdaki konveks problem çözülerek gerçekleştirilmektedir.

$$\hat{s}^G = \underset{s^G}{\text{argmin}} \parallel s^G \parallel_1 \text{ subject to } \parallel x - Ds^G \parallel_2^2 \leq \varepsilon^2 \tag{5}$$

ADMM Yöntemi : ADMM yönteminde kullanılan C-SALSA çözümü, belirli bir optimizasyon problemini daha kolay alt problemlere bölerek çözen ve daha hızlı yakınsama için bir Gauss-Seidel maliyet terimi ekleyen, arttırılmış Lagrange yöntemi tabanlı bir algoritmadır [21]. ADMM belirli bir iterasyon sayısı kadar hesaplama yapılmasına olanak sağlar. İterasyonlar öncesinde sözlük matrisinin tersi alınır. Matris için ters alma işlemi oldukça hesaplama yükü getiren bir işlemdir.

Ters alma işlemi sonrasında tanımlanan bir adım boyutu ve Lagrange çarpanları ile her bir iterasyonda çarpanlar güncellenir. Güncelleme, iterasyon sayısı kadar yapıldıktan sonra ızgara noktası kadar vektör üzerinde maksimum tepe bulma işlemleri gerçekleştirilir. Şekil 2.11'de ADMM yöntemi ile geliş açısı kestirimine yönelik yapılan alt işlemlerin bir akış sistematiğinde gösterimi sunulmaktadır.

ADMM İterasyon Analizi : Yakınsama için önemli olan ölçü aslında DoA tahminindeki hatadır. Ampirik çalışmalarda, her bir hedefin radar kesiti birbirine yakın olduğunda, %5 hata, geliş açısı kestirim performansını etkilememektedir [20].



Şekil 2.11: ADMM tabanlı DoA Tahmini İşlemi için Alt Akışlar

Çeşitli hata toleransları için yapılan çalışmada [20] her bir hata toleransı için gerekli yineleme (iterasyon) sayısını gösterir. Algoritma 40 yinelemede %5 hataya, 100 yinelemede ise %1 hataya yakınsar. Tez çalışmasında %2'ye yakın yakınsama hatası veren 64 adet yineleme adedi seçilerek optimum değer belirlenir.

3. LİTERATÜR İNCELEMELERİ

Bu bölümde Bölüm 2'de belirtilen geleneksel ve gelişmiş sinyal işleme uygulamaları için literatürde yer alan çalışmaların incelenmesi ve karşılıklı tartışma anlatılmaktadır.

3.1. Sayısal Alt Çevrimi (DDC)

Radar ve elektronik harp alanında sayısal alt çevrim çalışmaları en çok yazılım tanımlı telsiz uygulamalarında karşımıza çıkmaktadır. Yazılım tanımlı telsiz çalışmaları 1990'lı yılların başında askeri uygulamalar için DARPA tarafından desteklenmiştir. Yazılım tanımlı telsiz hakkında geçmişten o yıllara gelen çalışma serüveni, kritik analizler ve geleceğe dair öngörülerin yer aldığı makale [22], akademi ve endüstri dünyasında yazılım tanımlı telsiz çalışmalarındaki yeni gelişmelerin en önemli referanslardan biri olmuştur. GNU telsiz topluluğu da yazılım tanımlı telsiz çalışmaların en önemli destekçilerinden biridir [23].

Sayısal alt çevrimi aynı zamanda yazılım tanımlı telsiz olan radar sisteminde sayısal işleme bloklarında gerçekleştirilir. İkinci bölümde anlatıldığı üzere radar alıcısına gelen analog sinyaller örneklendikten sonra yüksek bant genişliğine sahip geniş bant sinyal, sayısal alt çevrimi işlemi ile dar bant sinyale indirgenir. Bu indirgeme işleminde girdi verisi büyük bir boyutta olduğu için indirgeme işleminin yüksek hızda yapılması radar sistemindeki gerçek zamanlı işleme açısından kritik önem oluşturur. DDC işlemi için sayısal sinyal işleme blokları, bu kritik hesaplama yolunu çözmek için temel olarak iki farklı hesaplayıcı mimarisinde tasarlanır. Bu mimariler DSP çekirdeği içeren çok çekirdekli CPU birimleri ve FPGA birimleridir.

GPU teknolojilerinin gelişmesi ile birlikte GPGPU içeren heterojen mimariler sayısal alt çevrimi için kullanılabilmektedir. İlgili çalışmalar incelenirken geleneksel yöntemler ile FFT tabanlı DDC yöntemi için yapılan çalışmalara ve paralel hesaplama tasarımlarına odaklanılmıştır. Bu kısımda alt başlıklarda maddeler halinde DDC yöntemleri için literatüre kazandırılan çalışmalar hakkında inceleme ve değerlendirmeler sunulmaktadır.

3.1.1. CIC/FIR tabanlı Sayısal Alt Çevrimi Çalışmaları

DDC işlemi aynı zamanda bir almaç yapısında örnekleme hızı değiştirme problemi için de kullanılır. Geniş bant bir sinyalden dar bant bir sinyale geçerken bant genişliğinin daralmasının etkisiyle örnekleme hızı da Nyquist Teoremi'ne uygun şekilde indirgenir. Agarwal ve diğerlerinin çalışmasında, Xilinx Virtex-6 ailesinden XC6VCX240t-2FF484 FPGA kartı üzerinde DDC işleminin tasarımı ve CIC alt indirgeme filtre işlemininin farklı kablosuz haberleşme standartları üzerindeki çeşitlenmesi ve örnekleme hızı değiştirimi anlatılmıştır [24]. Sensör sinyal işlemede büyük boyutlu verinin indirgenmesinde kullanılan DDC işlemi, haberleşme teknolojilerinde örnekleme hızı değişiminde bir adım olarak uygulanır. Agarwal ve diğerleri, CDMA ve GSM gibi çoklu standart içeren bir yazılım tanımlı telsizde çoğullaştırılmış CIC filtreler ile farklı indirgeme faktörleri için DDC işlemlerini gerçekleştirmişlerdir. Çalışmada CIC filtreleme işleminin sayısal tasarımının gerçeklenmesi yapılmış olup donanım kaynak kullanım karşılaştırmaları yapılmıştır. CIC filtreleme sonrası dengeleme (compensation) filtreleme işlemi yapılarak ile DDC işlemlerinde ara filtreleme sonrası oluşan kayıplar azaltılmıştır. İşlemler sırasında CORDIC [25] tabanlı hesaplamalar yapılmıştır. Dengeleme filtreleme işlemi ile kaynak kullanımı yaklaşık 12x artmıştır. Kullanılan FPGA kartının DSP donanım kaynaklarının kısıtlı olması sebebiyle çok kanallı DDC probleminde yenilikçi algoritmalar ile kolay bir şekilde yüksek başarımlar elde edilmesi mümkün değilir. Optimizasyon için düşük seviyeli donanım iyileştirmesi ile sonuçlar alınabilir. Yazarların haberleşme standartlarında GSM için seçenekli örnekleme hızı değişimi kapsamında literatüre katkı sağlamıştır.

DDC işleminde indirgemeden önce geçiş banttan temel banda kaydırma işlemi vardır. Bu işlem gerçekleştikten sonra dar bant indirgeme işlemi gerçekleştirilir. Jiang ve diğerlerinin çalışmasında, geniş bant bir sinyal için temel banda kaydırma sırasında paralel işlemeye uygun yeni bir yöntem önerilmiştir [26]. Geleneksel dikgen karıştırma temel bant çevrimi için paralel hesaplamaya uygun paralel karıştırıcı yöntemini önermişlerdir. Bu yöntemi aynı zamanda Çok Fazlı Alçak Geçirgen Filtre ile birleştirerek sayısal alt çevrimi işlemini yapmışlardır. Çok Fazlı filtreleme işleminde CIC filtre yapıları kullanılmıştır. Önerilen paralel

hesaplama yöntemini FPGA kartında deneysel olarak gerçekleyerek yapılabilirliğini göstermişlerdir. 4 ve 128 arasında ikinin üstel kuvvetleri olan sayıların değerinde indirgemeler yapılmıştır. Çalışmanın en büyük eksikliği gerçek zamanlı bir analiz yapılmamasıdır.
DDC işlemini evrişim filtreleme yöntemi ile yaptığımızda sıralı üç adım ile yapılır:

- Nümerik kontrollü osilatör (NCO) ile frekans kaydırma işlemi,
- CIC filtreleme ile geniş bant sinyalin adım adım dar banda indirgenmesi,
- FIR filtreleme ile gürültü bileşenlerinin elenmesi adımları ile filtreleme gerçekleştirilir.

Bu üç ana işlem için detaylı paralel hesaplamaların yapıldığı çalışmada Shao ve diğerleri, CUDA ve OpenMP programlama yapılarını kullanarak yüksek başarım elde etmeye çalışmışlardır [27]. Çalışmada yazarlar, NCO işlemi için LUT kullanarak sinüs ve cosinüs çarpımlarından dolayı oluşan darboğazları azaltmışlardır. LUT verisi ile sinüs fonksiyonu belirli bir çözünürlükle önceden hesaplanarak her defasında trigonometrik hesapların yapılmasının önüne geçmişlerdir. CIC filtreleme için bölütlere ayrılarak paralelleştirme yapılmıştır. FIR filtreleme ise CIC filtreleme ile oluşan her bölüt sonuç verisi için yapılmıştır. Deneysel çalışmalar çok çekirdekli Intel Xeon E5-2650 v2 CPU ve NVIDIA Tesla k20c GPU birimleri ile yapılmıştır. Hızlanma için GPU ve CPU birimlerinin işlem süreleri baz alınmıştır. CPU'ya kıyasla GPU birimleri ile elde edilen en yüksek hızlanma, NCO işleminde LUT kullanımıyla birlikte yapılmıştır. Çalışmadaki en önemli eksik kısım, LUT kullanımıyla NCO hesabında karşılaşılacak hataların analiz edilmemesidir. Çalışmada, CIC filtrelemeyi paralel hesaplama için bölütlere bölmek verimli bir yaklaşım sunsa bile bölütler arası filtreleme gecikmelerinin analizi çalışmada net belirtilmemiştir.

3.1.2. DFT/FFT tabanlı Sayısal Alt Çevrimi Çalışmaları

DDC problemi geniş bant bir sinyali çözümlerken duyduğu hesaplama gücü ihtiyacı yüksek bant genişliğini içeren diğer problemlerde de karşımıza çıkmaktadır. George ve diğerleri, bir radar mimarisinde çoklu hedef tespiti için yüksek bant genişliğine sahip bir veri üzerinde

paralel hesaplamalar yapmıştır [28]. Hedef tespiti için öncesinde filtreleme, pencereleme ve FFT ile spektrum hesaplama işlem adımlarını gerçekleştirmiştir. Örnekleme ile veri tipi çevrim işlemleri için Xilinx Virtex-5 FPGA mimarisini ve sinyal işleme adımları için ise Tesla C2050 GPU birimini kullanmaktadır. 3 Gsample/s hızına sahip bir radar almaç yapısında yaklaşık 1.25 GHz bant genişliğinde çoklu sinyal çevrimi yapılmaktadır. Radarın gerçek zamanlı çalışma kriteri bu hibrit mimari ile sağlanmış olup anlık dinamik aralığı literatür karşılaştırmalarına göre yüksek bir artış göstermiştir. Çalışmanın önemli sonuçlarından biri FPGA-DSP mimarilerinde yapılabilecek filtreleme, FFT işlemleri gibi sinyal işleme adımlarının yüksek bir başarımla GPU kullanımıyla yapılabilecek olmasıdır. Yüksek güç tüketimi öneren bir çözüm olduğu için mobil sensör mimarileri için pek uygun değildir. Ayrıca FPGA-CPU-GPU veri transferleri ile veri iletişim yükü artmıştır. İletişim için akıntı (stream) yapılar kullanılarak veri transferi iyileştirilebilir. Fakat bu çalışma GPU hesaplamanın sinyal işleme problemleri için kullanılabilmesini göstermesi açısından faydalı bir çalışmadır.

Geniş bant kanallaştırma, DDC ve yeniden örnekleme gibi data yoğun adımları içeren geniş bant yazılım tanımlı telsiz uygulamasıdır. Adhinarayanan ve Feng, geniş bant kanallaştırma için paralel hesaplama problemini incelemişlerdir [29]. Yazarlara göre geniş bant kanallaştırma FPGA ortamında DSP, LUT, FF ve BRAM gibi yapılar ile esnek bir şekilde tasarlanabilirken gerçekleme zor ve zaman alıcı bir duruma dönüşmektedir. Çok çekirdekli CPU'lar ile büyük sayıdaki kanallaştırma problemi için gerçek zamanlı performans gerekleri karşılanamamıştır. Bunun yerine muadillerine göre daha düşük güç tüketen AMD GPGPU birimleri geniş bant kanallaştırma işlemi için performans gereklerini FPGA'lerde olduğu gibi karşılamıştır. Ayrık Fourier Dönüşümü (DFT) tabanlı çok fazlı filtre bankaları sayesinde DDC işlemi her bir kanal için paralel hesaplama ile yapılarak iyileştirme sağlanmıştır. Gerçekleme uygulaması için OpenCL programlama kullanmışlardır. Deneysel çalışmalar sonucunda AMD Radeon HD 6470M GPU ile düşük güç tüketimi sayesinde mobil kullanımlar için de destek verebilecek geniş bant kanallaştırma uygulamasını ayrıntılı GPU optimizasyonları ile gerçekleştirmişlerdir. Çalışmanın literatüre sunduğu önemli katkı, GPU birimlerinin FPGA birimlerinin yerini alarak yazılım tanımlı almaç, radar ve elektronik harp alıcı yapılarında kullanılabilmesinin pratik olarak çalıştırılabildiğinin gösterimidir. Çalışmanın iyileştirilecek kısmı ise düşük veri hızı yerine daha fazla örnek sayısı ile çalışmasını sağlayarak yüksek veri

hızında iletişim sağlayan radar alıcılarına uyarlanabilmesinin mümkün hale getirilmesidir.
GPU teknolojisinin geniş bant kanallaştırma işlemlerinde hesaplama ve algoritma mimarisine azaltılmış karmaşıklık avantajını sağlayacağı birçok çalışmada öngörülmüştür. Bu öngörünün bir örneği olan çalışmada, Kim ve Bhattacharyya, grafik işlemci birimini kanalların ayrıştırılmasında, çok kanallı filtrelemede ve alt örnekleme işlemlerinde yüksek hesaplama verimi sunacak şekilde tasarımlarında kullanmışlardır [30]. Kanallaştırma işlemi, sinyal yukarı/aşağı dönüştürme, örnekleme hızı değişikliği ve filtreleme adımlarından oluşur. Yazarlar çalışmasında, çoklu fazlı kanallaştırma işlemi için alt adımları DFT/FFT tabanlı algoritma iyileştirmeleri ile GPU mimarisinde optimize etmişlerdir. NVIDIA GeForce GTX 680 ile 10 *ms* gerçek zamanlı çalışma süresi için hesaplamaları hedeflemişlerdir. Çalışmada, 60 MHz bant genişliği olan bir geniş bant sinyal 5 MHz'lik 12 adet alt kanallar ile işlenerek iletişim metodolojisi uygulanmıştır. Çok fazlı kanallaştırma işlemi GPU birimindeki veri akısı(data stream) mimarileri sayesinde veri paralelleştirme yapılmıştır. Yapılan gerçeklemeler sonrasında 3 *ms* civarında kanallaştırma işlemi gerçekleştirilmiştir. Ayrıca yazarlar aynı problem için literatüre sundukları önceki çalışmaya göre 70x hızlanma sağlamışlardır [31]. Yazarlar hızlanma sağladıkları çalışmaya göre daha yüksek bant genişliği ile çalışmışlardır. Fakat kanal sayısı azalmıştır. Yazarların bu çalışmadaki başarım kazanımları, yazılım tanımlı telsiz mimarilerinde GPU cihazlarının kullanılmasını motive etmektedir.
Geniş bant sinyal işleme yapan bir sensör mimarisinde ön kat (front-end) işleme yüksek bant genişliğinde bir problem çözümünü barındırdığı için kritik olmaktadır. Bir önceki çalışmada belirtildiği üzere Adhinarayanan ve Feng, geniş bant kanallaştırma işlemini AMD GPU mimarileri ile verimli bir şekilde gerçekleştirmişlerdir. Benzer şekilde Kim ve Bhattacharyya, geniş bant kanallaştırma problemini OFDM ve Evrensel Mobil Telekomünikasyon Sistemleri (UMTS) iletişim yöntemleri için çalışmışlardır [32]. İletişim sistemlerinde ihtiyaç duyulan dinamik ve esnek yeniden örnekleme tasarımları için GPU tasarımları yarar sağlamıştır. Yazarlar, NVIDIA GTX GPU mimarileri ile gerçek zamanlı DFT tabanlı filtreleme içeren geniş bant ön uç işleme adımları gerçekleştirmişlerdir. Özellikle 4G haberleşme sistemlerine avantaj kazandıracak hesaplama ile düşük gecikme süresi ve düşük hata çözünürlüğü barındıran bir çözüm sunmuşlardır. Çalışmanın geliştirilecek en önemli kısmı, daha yüksek bant genişliklerine çıkarılarak ya da daha büyük indirgeme oranı kullanılarak daha fazla alt kanalların

oluşturulmasıdır.

Filtreleme işlemi, FPGA gibi aynı anda çok sayıda DSP kaynağının olduğu hesaplama mimarisinde çok hızlı gerçekleşebilmektedir. Fakat filtre uzunluğunun yüksek olması, kısıtlı DSP kaynakları ve gerçekleme sürecinin uzun olması nedeniyle FPGA birimleri her senaryo için verimli olamamaktadır. Evrişim işlemini doğrusal bir filtreleme yerine FFT tabanlı bir yöntemle çarpım halinde yapmak eleman bazlı bağımsız işlem yapma olanağı sunan GPU gibi hesaplama birimlerinde fayda sağlamaktadır. Borgerding, hızlı evrişim ismi altında örtüşen kaydet (overlap-save, OS) FFT yapısı ile filtreleme işlemini paralel hesaplamaya uygun hale getirmiştir [33]. Büyük uzunluktaki FFT işlemlerinin uzun sürebilmesi nedeniyle işlem öncesi bölütleme yaparak OS-FFT çözümünü hızlı evrişim yöntemi olarak sunmuştur. Öncesinde oluşturulan filtre bankaları sayesinde yüksek başarımlı bir şekilde frekans uzayında filtreleme yapmıştır. Frekans kaydırma yani NCO işlemleri içinde frekans uzayında dairesel kaydırma ile belli bir hata varsayımıyla çözüm önermiştir. Özellikle büyük uzunluklu veriler ile işlem yaparken önerdiği yöntem sayesinde filtre uzunluğundan büyük fakat girdi uzunluğuna göre daha küçük girdi bölütleri ile filtreleme işlemi hem CPU'da hem de GPU'da verimli hesaplama ile gerçekleştirilmesi mümkün hale gelmiştir.

3.1.3. Tartışma

Literatür taramasında görüldüğü üzere sayısal alt çevrimi işlemleri genellikle FPGA/DSP mimarilerinde CIC/FIR tabanlı algoritma adımları ile yapılmaktadır. Buna karşın özellikle çok sayıda hesaplama çekirdeği barındıran GPU mimarilerinde ise DFT/FFT tabanlı sayısal alt çevrimi çözümleri gerçekleştirilmektedir. GPU kullanımında CIC/FIR içeren çözümlerde mevcuttur. Fakat kanallaştırma problemlerinde de görüldüğü üzere FPGA mimarilerinde CIC/FIR ile alınan verim görülememektedir. Tarihsel gelişim olarak DSP teknolojileri ile başlayan yazılım tanımlı telsiz uygulamaları özellikle geniş bant sinyal işleme yapılırken FPGA ve GPU gibi güçlü hesaplayıcılar ile desteklenen kabiliyet ve bant genişlikleri çalışmalarda görüldüğü üzere arttırılmaktadır.

Sayısal alt çevrimi hesaplama başarım artışları için literatür çalışmaları çoğunlukla yazılım

tanımlı telsiz uygulamaları özelinde gerçeklense dahi radar, elektronik harp, optik sinyal işleme ve akustik alıcı sinyal işleme bloklarında da uyarlanması mümkündür. Çok kanallı radar almaç yapısında sayısal alt çevriminin zaman kritik ve data yoğun bir işlem olduğu bilinmektedir. Bu işlemin gerçek zamanlı yapılması radar alıcısının tespit ve takip işlemleri açısından kritiktir. Literatürde, gerçek zamanlılık gereğini karşılamak için hesaplama başarım artışı getiren DFT/FFT tabanlı çoklu fazlı filtreleme üzerinde sıkça durulmuştur. FFT tabanlı çözümler GPU mimarilerinde daha verimli çalıştığı için AMD ve NVIDIA GPU hesaplama birimlerini sayısal alt çevriminde kullanmak FPGA kullanımına göre hem esnekliği arttırmakta hem de donanım ve geliştirme karmaşıklığını azaltmaktadır. Sayısal alt çevrimindeki indirgeme işlemi yeniden örnekleme işlevinin karşılanması olarak ele alındığına literatürde DFT/FFT tabanlı çalışmalarda görüldüğü üzere birçok ilhâm verici yayın görülmektedir. Tez çalışmasında odak konulardan biri olan OS-FFT tabanlı hızlı evrişim yöntemi sayesinde CIC/FIR tabanlı evrişim ile filtreleme yerine hızlı bir filtreleme tekniği incelenmektedir.
Tez çalışmasında, literatürde görülen kaydırma ve sayısal alt çevriminde eksik olarak nitelendirebilecek düşük güç tüketimi sağlayan mobil GPGPU ile hesaplamalar uyarlayarak sensör mimarilerinde başarım artışı sunmak amaçlanmaktadır. NVIDIA Jetson GPU mimarileri gibi 10 ila 30 Watt arasında güç harcayan sistemlerde sayısal alt çevrimi işlemini OS-FFT ile yaparak gerçek zamanlı bir senaryoda çalışmasını sağlamayı planlanmaktadır.

3.2. Makine Öğrenmesi tabanlı Hedef Tespit İşlemi

İkinci bölümde bahsedildiği üzere SYAO tekniğine göre düşük sinyal gürültü oranı durumlarında makine öğrenmesi ile yapılan tespit işlemi, tespit doğruluğu açısından başarılı sonuçlar vermiştir [12]. SYAO tekniğinin çok sayıda FPGA, CPU ve GPU hesaplama ve gerçekleme çalışmaları literatürde bulunmaktadır [34–37]. Bu kısımda sadece makine öğrenmesi ile tespit işlemleri için gerçekleştirilen literatür çalışmalarına yer verilmektedir.
Tez çalışmasında kullandığımız makine öğrenmesi modelinin bir CNN modeli olduğu Bölüm 2'de belirtilmektedir. CNN modeli evrişim filtreleri sayesinde sınıflandırma için verilerin ve yapay sinir katmanların özniteliklerini çıkartır [38]. CNN modelinin tarihsel gelişimine

bakarsak 2012'de düzenlenen ImageNet yarışmasındaki önemli sonucu incelemek gerekir. CNN modelinin imge sınıflandırmada önemli bir başarım artışı sunduğunu literatüre duyuran çalışmada yazarlar, 60 milyon parametre, 650.000 nörona sahip, 5 adet evrişim katmanı (havuzlama dahil) ve 1000 patikalı aktivasyon ve tam bağlantılı üç katmandan oluşan bir modeli eğitmişlerdir [10]. GPU kullanımının özellikle iki boyutlu(2B) evrişim işlemlerinde verimli olabileceğini değerlendiren Krizhevsky ve diğerleri, eğitim sürecini paralel hesaplama ile daha hızlı hale getirmişlerdir. Testler sonucunda yarışmanın birinciliğini %15.2 hata oranı ile ikinci yarışmacıdan çok iyi bir skorla almışlardır. Yazarlar bu skorun henüz yeterli olmadığını daha iyi model tasarımları ile imge sınıflandırma insan kararına yakınlaşabileceğini öngörmüşlerdir. Bu çalışmanın literatüre sunduğu en önemli katkı CNN modelinin GPU mimarilerindeki hesaplama başarım artışı ile imge sınıflandırma yaşanan uzun model eğitim sürelerinin azaltılmasıdır.

Makine öğrenmesi, derin öğrenme ve yapay zekâ teknolojilerinde birçok işleme katmanından oluşan hesaplama adımlarına sahip soyutlama katmanları ile öznitelik ve metriklerin öğrenilmesi sağlanır. Probleme ait kritik bilgilerin öğrenimi sayesinde nesne tespit, konuşma tanıma, yüz tanıma ve imge sınıflandırma uygulamalarında başarım artmaktadır. Derin öğrenme model tasarımlarında genel bir gözden geçirim yapan ve çok sayıda alıntı alan çalışmada, yazarlar Evrişimsel Sinir Ağı (CNN) ve Devirli Sinir Ağı (Recurrent Neural Network, RNN) modelleri için bilgilendirme ve açıklamalar sunmuşlardır [9]. LeCun ve diğerleri, çalışmasında model eğitim sırasındaki geri yayılım için yapılan hesaplama ve tasarımlar hakkında bilgi vermiştir. Yazarlar derin öğrenme için gözden geçirim yaparak, eğitim modelleri, uygulandıkları alanlar ve gelecekteki öngörülerini anlatmışlardır. Çalışmada verilen en çekici öngörü, gelecekteki ilerlemenin pekiştirmeli öğrenme tabanlı RNN ve CNN modellerini birleştiren sistemlerden çıkacağı beklentisidir. Dikkat çeken diğer bir öngörü ise derin öğrenmede model yapısı daha çok etiket tabanlı denetimli (supervised) öğrenme ile yapılsa bile denetimsiz (unsupervised) öğrenmenin, çığır açacak gelişmelere etki edeceğidir.

Araştırmacılar için, derin öğrenme ve makine öğrenmesinde başlangıç çalışması genellikle MNIST veritabanında yer alan karakter ve rakamların otomatik olarak tanınmasıdır [39]. Derin öğrenme literatüründe yaptığı çalışmalardan ilhâm alınan bir araştırmacı olan LeCun,

ekibiyle yaptığı çalışmalarında gradyan tabanlı geri yayılım algoritmasının çok katmanlı sinir ağlarındaki başarımını incelemiştir [40]. Çalışmada, el yazısı karakterini tanımak için uygulanan çeşitli metotlar incelenmiş ve bu metotlar ile rakam tanıma için karşılaştırmalar yapılmıştır. Çalışmada evrişimsel sinir ağların iki boyutlu değişkenlikle başa çıkmada daha iyi başarım kazandırdığı sunulmuştur. Sinir ağı modelinin iyi eğitilmesi başarımı artıran bir faktör olduğu bilindiğinden yazarlar, geliştirdikleri modelin gradyan tabanlı yöntemler ile eğitilmesini sağlamışlardır. Çalışmada MNIST veri tabanında testler gerçekleştirilmiş olup gradyan tabanlı öğrenmenin başarılı sonuçlar verdiği gösterilmiştir. CNN modellerin geri yayılım algoritmaları ile birlikte karakter tanıma, rakam tanıma ve genel olarak nesne tanıma problemlerinde başarılı bir sonuç verebileceği ve bozulan girdi verilerinin ön işlemler ile düzenlenebileceği bu çalışmada ayrıntılı olarak gösterilmektedir.

CNN modeli uygulaması için genel bir literatür taraması sunan çalışmada yazarlar, CNN modelinin katmanlarını, regülasyon problemini ve alternatif CNN modellerini detaylıca anlatmışlardır [41]. Evrişimsel sinir ağı tasarımlarına başlayacak bir araştırmacı için referans çalışması sunan yazarlar, sırasıyla LeNet-5, AlexNet, GoogleNet, VGGNet ve ResNet için ön fikirler sunmuşlardır. Yazarlar Aloysius ve diğerleri tarafından yapılan değerlendirmeler her bir model için maddeler halinde belirtilmiştir:

- **LeNet-5** : El yazısı ile oluşturulan karakter tanıma, rakam tanıma ve kod tanıma çalışmaları için literatüre LeNet-5 yapay sinir ağı modeli sunulmaktadır [42].
- **AlexNet** : ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012 yarışması ile duyuran CNN tabanlı modelde LeNet-5 modeline göre daha derin ve büyük bir sinir ağı modeli sunulmuştur [10]. NVIDIA GTX 580 GPU ile yapılan testler sonucunda CNN modellerinin kullanım motivasyonu arttırmıştır.
- **GoogleNet** : GoogleNet modeli evrişimsel sinir ağı modellerinde kaynak kullanımının kritikliği inceleyen ve AlexNet'e göre yüksek bir kaynak optimizasyonu sunan CNN modelidir [43].
- **VGGNet** : ILSVRC 2014 yarışmasında ikinci olan bir çalışmada yazarlar, evrişimsel sinir ağında derinlik faktörünü inceleyen bir çalışma yapmışlardır [44]. Simonyan, bu

çalışmada $3x3$ evrişim filtreleri ile tüm katmanlarda verimli bir şekilde kontrol yapmaktadır.

- **ResNet** : Yapay sinir ağları tasarımı için eğitim maliyetini azaltan bir model sunan yazarlar, artık (residual) öğrenme ağı sunmuşlardır [45].

Aloysius ve diğerleri, çalışmalarında ConvNet modelleri için karşılaştırmalı alternatifleri sunduktan sonra açık çalışma konularını sıralamışlardır [41]. Özellikle evrişimsel sinir ağı modellerinin model, katman ve parametre optimizasyonu ile daha verimli hale getirebileceği öngörülmüştür.

Evrişimsel sinir ağı modellerinin sunmuş olduğu nesne tespit ve sınıflandırma çözümleri güvenlik ve savunma alanlarındaki kamera, radar gibi sensör tabanlı sistemlerdeki problemlere de uyarlanabilmektedir. Güvenlik alanında şehir içi ortamda araç tespiti ve takibi gibi uygulamalar için kullanılan İHA ve Drone kameraları ile derin öğrenme ağları sayesinde nesne tespiti ve sınıflandırılması yapılabilmektedir. İncelenen bu çalışmada yazarlar, araç tespiti için öznitelik çıkarımında CNN modelini kullanmış olup sınıflandırma için SVM modelini kullanmışlardır [46]. Çalışmada Ammour ve diğerleri, kalabalık nesne içeren imgeler üzerinde araç var/yok sınıflandırmasını CNN sayesinde doğruluk başarımı yüksek ve iyileştirilmiş hesaplama süresi ile literatüre sunmuşlardır.

Güvenlik alanında kamera görüntülerinin yanı sıra radar sensöründen alınan yankılardan da yararlanılır. Wang ve diğerleri, radar sensör mimarisindeki hedef tespiti işlemini bir CNN modeli ile sağlayıp başarım analizleri yapmışlardır [47]. Radar sinyal işleme adımları ikinci kısımda Şekil 2.5'te gösterildiği gibi örnekleme işleminden sonra ön işleme adımları başlamaktadır. Bu adımlar sırasıyla, uyumlu filtreleme, Doppler işleme ve menzil-Doppler matrisinin oluşturulmasıdır. Ön işleme adımlarından sonra oluşan menzil-Doppler matrisi üzerinde geleneksel tespit yöntemleri uygulanarak hedefler bulunurken Wang ve diğerleri ise bu matris üzerinde derin öğrenme ağı tabanlı hedef dedektörü ile tespit işlemini gerçekleştirmişlerdir. Derin öğrenme modeli için CNN modelini kullanmışlardır. Nesne tespitinde kullanılan var/yok sınıflandırma bakış açısı ile radar hedef tespiti probleminin modelinde "tespit var/-tespit yok" sonucunu çıkarmışlardır. Tespit doğruluğunu geleneksel yöntem olan SYAO ile karşılaştırma yaparak analiz etmişlerdir. Sonuçlara göre SYAO tekniğine kıyasla büyük bir

kazanç sağlamamışlardır. Ayrıca hesaplama karşılaştırması yapmamış olup CNN modelinin ayrıntıları sunulmamıştır. Çalışmanın motive sağladığı kısmı radar hedef tespiti için CNN modelinin uygulanabilir olmasıdır. Fakat çalışma gerek model optimizasyonu gerekse de tespit doğruluğu açısından geliştirmeye açıktır.

Radar sensörleri RF yankılardan aldığı sinyali örnekleyerek hedef ve anlamlı veriler bularak tespit, takip ve sınıflandırma fonksiyonlarını gerçekleştirir. Sentetik Açıklıklı Radar (SAR) gibi görüntüleme moduna sahip olan radar sensörleri gece-gündüz keşif ve gözetleme amaçlı görüntü elde edilmesini sağlar [48]. SAR sensörleri uydu teknolojilerini içeren gözetleme uygulamalarında sıkça kullanılır. Görüntülerin değerlendirilmesi makine öğrenmesi uygulamaları ile yapılmakta olup otomatik nesne tespiti (automatic object detection) problemlerine girdi vermesi diğer radarlara göre daha kolay olmaktadır. SAR görüntüleri üzerinde derin öğrenme metotları ile öznitelik öğrenimi yapılarak başarım artışı öngürülen çalışmada yazarlar, görüntü üzerindeki otomatik odaklanma sırasındaki faz hatası problemini çalışmışlardır [49]. Mason ve diğerleri, odaklama sırasındaki faz hatasını öğrenen bir modeli devirli sinirsel ağ (RNN) temeliyle oluşturmuş olup hata giderim işlemlerini gerçekleştirmişlerdir. Yazarlar radar problemlerinde yapay zekâ uygulamaları ile avantaj sağlanabilecek görüntüleme, otomatik odaklanma, değişim tespiti, öznitelik çıkarım ve sınıflandırma gibi uygulama alanlarını belirtmişlerdir. Ayrıca, derin öğrenme modellerinin radar problemlerinde çeşitli alanlara uyarlanması için bir vizyon ortaya koymuş olup, spesifik olarak SAR görüntüleri üzerinde derin öğrenme çalışmalarını literatüre kazandırmışlardır. Bu çalışmanın radar verisi üzerinde yapılacak makine öğrenmesi, derin öğrenme çalışmalarına öngörü kattığı söylenebilmektedir.

SAR görüntülemede CNN modelini kullanarak iyileştirme sağlayan yazarlar, geliştirmiş oldukları model için katmanlara ve parametrelere ait detaylı bilgi vermişlerdir [50]. Deneysel sonuçlar doğrultusunda modifiye ettikleri CNN modeli sayesinde %99 doğrulukla sınıflandırma yapılmıştır. Chen ve diğerleri, Hareketli ve Sabit Hedef Tespiti ve Tanıma (MSTAR) veri seti üzerinde testler gerçekleştirmişlerdir [51]. Geleneksel ConvNet modelini kullanınca sonuçlarda bir aşırı uyum gösterme (overfitting) görüldüğünden modelin başarımının testlerde azaldığını gözlemlemişlerdir. Seyrek evrişim tabanlı bütünsel bir CNN modeli ile başarım artışı sağladıklarını belirtmişlerdir. Test verilerinde askeri imgeler kullanılarak

özellikle savunma sanayinde yapılacak çalışmalara referans kazandırmışlardır. Yazarlar, tam bağlantı katmanlarını geliştirmiş oldukları modelde eleyerek MSTAR data seti üzerinde overfitting durumunun azalmasını sağlamışlardır. Çalışmayı değerlendirdiğimizde özellikle SAR sensörü mimarisinde hedef sınıflandırma için kullanılabilecek bir modeli yazarlar literatüre kazandırmıştır. Bu çalışmada kargaşa verisi kullanılarak modelin hedef tanıma işlevinde daha kararlı ve güvenilir sonuç vermesinin sağlanması çalışmanın kalitesini artıran bir etki yapmıştır.

Otomotiv modellerinde kaza önleme ve güvenlik artırımı için yenilikçi sensör yapılarından yararlanılmaktadır. Mesafe ölçen sensörlerin yanı sıra trafikteki nesne bilgisini veren radar, lidar ve kamera sistemleri tasarlanmaktadır. İncelenen çalışmada yazarlar, otomobilde mevcut olan lidar, kamera ve radar sensörlerinin çoklu-nesne tespiti ve takibi için içerdiği tespit ve takip modelleri ve derin öğrenme model tasarımlarını aktarmışlardır [52]. Ravindran ve diğerleri, çalışmasında bu üç sensörün birlikte çalışması ile daha optimize modeller üretilebileceğini ve muhtemel potensiyelleri anlatmıştır. Yazarlar hedef sınıflandırması için kategorize edilmiş derin öğrenme ve makine öğrenmesi modellerini literatürdeki çalışmalara atıfta bulunarak açıklamışlardır. Çalışmanın değerlendirilme kısmında, yüksek çözünürlüğe sahip radar sistemlerinin her ortam (gece, gündüz, sis vb.) koşulunda verimli bir sonuç sunabileceği ve üç sensörün optimize bir karar algoritması ile başarılı bir sonuç vereceğini belirtmişlerdir. Bu makalenin literatürdeki sensör derin öğrenme modelleri içeren çalışmaları anlatması ve potansiyel çalışma konularını ortaya koymasından dolayı özellikle otonom araçlarda yapılacak çalışma katkı verebileceği düşünülmektedir.

Radar, kamera ve diğer sensör tabanlı sistemler uzaktan algılama teknolojilerine birer örnek teşkil etmektedir. Uzaktan algılama çalışmalarında derin öğrenme modellerini analiz eden ve geliştirme ortamlarını inceleyen araştırma çalışmasında yazarlar, derin öğrenme çalışmalarında başlangıç seviyesinde olan araştırmacılara öğretici ve ilgi çekici bir literatür çalışması yapmışlardır [53]. Ball ve diğerleri, uzaktan algılamada derin öğrenme konuları için insan-hayvan tespiti, trafik akış analizi, hava durumu, gelişmiş sürücü yardımcı sistemleri (ADAS), anomali tespiti, nesne tespiti ve takibi, su altı tespiti ve araç tespit-takibi gibi çeşitli problemleri ele alan çalışmaları inceleyip detaylı analizler sunmuşlardır. Radar sensörü için ise SAR

görüntüleri üzerinde makine öğrenmesi çalışmalarını incelemişlerdir. Çalışmada yazarlar derin öğrenme araçları ile ilgili bilgiler vermişlerdir. Bu araçlar ve özellikleri sırasıyla maddeler halinde belirtilmektedir:

- **Caffe** : Python ve MATLAB arayüzleri içeren C++ tabanlı derin öğrenme modelleri inşa etmeye yarayan bir araçtır [54].
- **Keras** : Yüksek seviyeli bir Python tabanlı sinir ağı kütüphanesidir [55]. Hızlı ve esnek bir prototipleme sunduğu gibi, hem CNN hem de RNN modellerinin tasarımına olanak sağlamaktadır.
- **TensorFlow** : Açık kaynak makine öğrenmesi modellerini geliştirme ve uygulama aracıdır [56]. Açık kaynak olmasının getirdiği esneklik ile çoğu dile destek sunmaktadır.
- **Theano** : Çok boyutlu dizileri içeren matematiksel problemlerin çözümü için optimize sonuçlar sunan ve NumPy desteği sağlayan Python tabanlı geliştirme kütüphanesidir [57].
- **Torch** : LuaJIT kodlama ve CUDA C/C++ desteği veren hesaplama çerçevesidir [58]. Optimize edilmiş doğrusal cebir, sinir ağları ve GPU desteği ile derin öğrenme, makine öğrenme çalışmalarında sıkça kullanılmaktadır.

Ball ve diğerleri, maddeler halinde belirtilen araçları çalışmalarında değinmiş olup geliştirme ortamları hakkında kısaca bilgi vermiştir. Değerlendirmemize göre bu çalışma uzaktan algılama özelliği içeren bir sensör mimarisinde yapılabilecek derin öğrenme veya makine öğrenmesi çalışmalarına başlarken yardımcı olabilecek bir literatür çalışmasıdır.
Uzaktan algılama sensör teknolojilerinde uç hesaplama mimarilerinde enerji verimliliği artırmak için düşük güç tüketen işlemci birimleri ile hesaplamalar gerçekleştirilir. Derin öğrenme ve makine öğrenmesi modelleri gibi uç hesaplama içeren sensörlerde çalışabilecek algoritmalar için gerçek zamanlı çalışmanın gerçekleştirilmesi kritik olmaktadır. Bu bakış doğrultusunda insansız hava aracı sistemindeki kamera sensörünün otomatik insan tespitini gerçek zamanlı yapabilmesini hedefleyen çalışmada yazarlar, tespit doğruluğu yüksek

bir CNN modeli ile verimli ve hızlı bir çözümü literatüre sunduklarını belirtmişlerdir [59]. Golcarenarenji ve diğerleri çalışmalarında, insansız hava araçlarının arama-kurtarma çalışmalarında insan tespiti, hayvan tespiti ve nesne tespiti problemlerinde kamera sensör birimi ile etkin bir şekilde kullanıldığını motivasyon ve problem tanımı olarak anlatmışlardır. Yol Toplama Ağı [60] ile genişletilmiş evrişimlerin kombinasyonu içeren bir derin öğrenme modeli ile insan tespiti başarımını artışmışlardır. YOLOv4 çalışması ile yapılan doğruluk karşılaştırmasında daha az başarım gösterse dahi sonuçları YOLOv4 ile yakın çıkmıştır [61]. Çalışmada ayrıca NVIDIA Jetson AGX Xavier GPU kullanılarak 10 Watt ile 5 fps (frames per second), 30 Watt ile 20 fps sonucunu almışlardır. Çalışmayı değerlendirdiğimizde yazarlar, arama kurtarma operasyonları amacıyla kullanılacak görüntüler üzerinde uzaktan maliyet etkin insan tespiti sağlamak için yüksek doğrulukta ve gerçek zamanlı olarak yenilikçi bir makine öğrenmesi tabanlı insan tespit sistemi önermişlerdir. Bu çalışma makine öğrenmesi ile nesne tespitinde hem model tasarımı hem de gerçek zamanlı uygulama açısından başarılı bir sonuç sunmuştur. Gerçek test verisi ile yapılacak testlerde hem başarım analizi hem de gerçek zamanlı çalışma performanslarının yeniden değerlendirilmesi faydalı olacaktır.

Yapay zeka çalışmalarında eğitim ve sonuç çıkarım (inference) sürelerinin değeri araştırma sürecinin önemli basamaklarını etkilemektedir. Eğitim süresinin azaltılması model başarım analizleri açısından kritik olmakla birlikte sonuç çıkarım süresinin azaltılması gerçek zamanlı uygulamalar için önemli olmaktadır. Derin öğrenme modellerini hızlandırmak için yapılan donanım ve yazılım optimizasyon çalışmalarını inceleyen ve karşılaştırmalı sonuçları veren çalışmada yazarlar, eğitim ve sonuç çıkarım sürelerinin hangi çalışmalar ile azaltıldığını gösteren bir literatür makalesini sunumuşlardır [62]. Capra ve diğerleri, enerji verimliliği esas alan literatür çalışmalarındaki donanım ve yazılım optimizasyonları hakkında çeşitli yöntemlerden bahsetmişlerdir. Çalışmanın denektaşı analizlerinde 2016'dan beri yapılan yapay zeka eğitim ve sonuç çıkarım çalışmalarına atıf yapılarak karşılaştırmalar yapılmıştır. Çalışmayı değerlendirdiğimizde farklı veri setlerini kullanan modeller için alınmış CPU-GPU-FPGA başarım analizleri karşılaştırmalı olarak verildiğinden ilgili model kullanımında yapılabilecek en etkili donanım optimizasyonu hakkında tavsiyeler sunulmuştur. Hesaplama tarafını ilgilendiren kısımda bellek alanında çalışmanın dezavantajları ve bu zorluğun üstesinden gelmek için yapılabilecekler ayrıntılı bir şekilde anlatılmıştır.

Bu tez çalışmasında makine öğrenmesi uygulaması için tasarladığımız güç verimli hesaplama fikrinin literatür çalışmaları incelendiğinde imge üzerinde nesne tespit probleminin düşük güç tüketen mimarilerde gerçeklenmesine dair sonuçlara rastlanmaktadır. NVIDIA Jetson GPU platformlarında derin öğrenme modellerininin gerçeklenmesini tarayan ve karşılaştırmalar yapan çalışmada yazar, sonuç çıkarım işlemi için optimizasyon gerçekleştirmiş olup uygulama çalışmalarını anlatmıştır [63]. Mittal, derin öğrenme hesaplama mimarilerine uyumlu Jetson platformları olan TX1 [64], TX2 [65] ve Xavier GPUlar [66] ile ilgili detaylı bilgiler vermiştir. Çalışmada, Jetson mimarilerini Rasperry Pi [67] ve Intel NUC [68] mimarileri gibi benzer sistemlerle karşılaştıran, avantajlarını ve kısıtlarını vurgulayan literatür çalışmaları tartışılmıştır. Yazar, çalışmada yapay zeka çalışmaları ile bilgisayar mimarilari arasında köprü kurularak iki ayrı teknoloji çalışmasının pratik uygulamalar için örtüşen kısımlarının birlikte optimize edilmesini amaçlayan araştırmalara motivasyon vermiştir. Jetson birimlerinin yerel işlemeyi etkinleştirmesi, düşük ağırlık ve güç tüketimine sahip olmaları ve maksimum başarımlarının çalışmada incelenen diğer mobil CPU hesaplama birimlerine göre avantajlı olması derin öğrenme uygulamalarında tercih sebebi olduğu çalışmada belirtilmiştir. Bu çalışmanın bu tez çalışmasına etki ettiği en önemli kısım, NVIDIA Jetson platformların uç hesaplamada yapılacak makine öğrenmesi tabanlı sonuç çıkarım çalışmalarında düşük güç tüketimi ile başarım artışının gösterimidir.

Bu tez çalışmasında, referans alınan makine öğrenmesi ile radar hedef tespiti çalışmasında, yazar geleneksel hedef tespiti yöntemine HO-SYAO'ye göre gürültülü ortamda doğruluk başarım artışı sunan bir CNN modelini tasarlamıştır [1]. Yavuz, bu çalışmasında hedef tespiti için kargaşa alternatifleri, gürültü modelleri ve farklı konumlara uyarlanabilen sentetik veri seti ile çalışmıştır. Çalışmasında tasarladığı CNN ağı menzil-Doppler imge matrisinde adım adım tarama yaparak tıpkı SYAO tekniği gibi karar vermektedir. Matriste tarama işlemini iki boyutlu şekilde yaptığı için 2B evrişim filtreleme işlemini modelinde gerçekleştirmiştir. Model özellikle gürültü ve kargaşa ortamında HO-SYAO'ya göre daha yüksek tespit doğruluğu vermiştir. Ayrıca CNN modeli kullanması ve adım adım tarama işleminin optimizasyon açık bir paralel hesaplama tasarımına olanak sağlaması çalışmanın literatüre en önemli katkısı olmuştur. Çalışma değerlendirildiğinde, gerçek veriler ile yapılacak eğitim çalışmaları ve model optimizasyonları ile daha başarılı hale getirilebileceği düşünülmektedir.

3.2.1. Tartışma

Makine öğrenmesi ile hedef tespiti çalışmaları için makine öğrenmesi ve derin öğrenme ile ilgili gerçekleştirilen literatürde takip edilen çalışmalar incelenmiştir. Literatürde bilgisayarlı görü temelinde gelişen imge tespiti ve sınıflandırmaya yönelik derin öğrenme çalışmaları ve makine öğrenmesi uygulamaları çeşitli makaleler ile analiz edilmektedir. Dikkat çeken en önemli nokta özellikle CNN ağlarının 2012'den itibaren GPU destekli mimariler ile eğitim ve sonuç çıkarım işlemlerinin performans artışına etki etmesi olmuştur. Derin öğrenme uygulamaları uzun süren eğitim işlemlerinden dolayı pratik uygulamalarda ve araştırma çalışmalarında daha az tercih ediliyordu. Fakat özellikle GPGPU mimarilerin gelişimi, evrişim işlemlerinin optimize hesaplamalar ile gerçekleştirilmesi derin öğrenme ve makine öğrenmesi ile ilgili yenilikçi çalışmaların önünü açmıştır.

Literatür incelemesinde radar sensörleri ile ilgili derin öğrenme çalışmaları genellikle SAR sensöründeki görüntü iyileştirme, nesne tespit ve takibi konularında yoğunlaştığı görülmektedir. SAR sensörü gece-gündüz görev yapma kapasitesi ile keşif-gözetleme görevlerinde askeri ve uzay alanındaki çalışmaların icra edilmesini sağlar. SAR görüntüleri üzerinden anlamlı veri çıkarımı için elde edilen görüntü matrisindeki nesnelerin anlamlandırılması ve sınıflandırılması gerekir. Ayrıca görüntü çıkarımı sırasında görüntü üzerinde kalibrasyon kaynaklı kayıklıklar veya bozulmalar gerçekleşebilmektedir. Benzer durumları bertaraf edebilmek için derin öğrenme eğitim modellerinden faydalanılmaktadır. SAR sensörü dışında, yer altı görüntüleme radarı ve otomobil radarı gibi sensör yapılarında da nesne tespiti, görüntüleme ve sınıflandırma işlemleri için yapay zekâ modelleri kullanılmıştır.

Derin öğrenme çalışmalarında GPU mimarileri ile birlikte sonuç çıkarım süreleri de iyileşmiştir. Bu sonuç uç hesaplama birimlerinde derin öğrenme tabanlı algoritmaların gerçek zamanlı hesaplamaları için motivasyon oluşturmuştur. Uç hesaplama birimi olarak FPGA, CPU birimleri haricinde düşük güç tüketen yüksek başarım sunan Jetson GPU mimarileri ile gerçek zamanlı gereksinimlerin karşılanması mümkün hale gelmiştir. İncelenen çalışmalarda Jetson GPU mimarileri kamera görüntülerinde imge sınıflandırma ve uzaktan algılama çalışmalarında yüksek başarımlı uç hesaplama için kullanılmıştır. Bu anlamda uç hesaplamaya yönelik verimli heterojen programlama tasarımını bu tez çalışmasında irdelemek

literatürdeki gelişmeleri takip eden bir yaklaşım olmuştur. İHA, Drone ve diğer insansız mobil sistemlerde boyut, ağırlık, güç tüketimi ve maliyet (SWaP-C) faktörü sistem tasarımı açısından önemlidir. Bu öneme literatür çalışmalarında değinilmesi, özellikle uç hesaplama birimlerinde SWaP-C faktörü dikkate alınarak Jetson GPU ve düşük güç tüketen mimariler ile makine öğrenmesi ve derin öğrenme hesaplamalarının gerçekleştirilmesi, bu tez çalışmasının katkıları için yapılan vurguya pekiştirici bir motivasyon sağlamıştır.

CNN temelli derin öğrenme modellerinin incelenmesi ve karşılaştırması tez çalışması açısından temel bir yapı oluşturmuştur. Ayrıca derin öğrenme modellerinin eğitim ve gerçekleme işlemleri için kullanılan çerçeve yapılarını incelemek teknolojik gelişim ve farkındalığı artıran bir analiz olmuştur. Tez çalışmasında kullanılan CNN modelinin, LeNet-5 modelinin modifiye ve optimize edilmiş bir alternatifi olması ve diğer CNN modelleri ile olan avantaj-dezavantaj karşılaştırmaları bilgilendirici olmuştur. Sensör tabanlı problemler için geliştirilmiş CNN modeli ile yapılan çalışmalarda genelde doğruluk analizleri ve SNR başarım artışları incelenmiştir. Fakat tez çalışmasında gerçek zamanlı hesaplama ve paralel programlama ilgi odağı olduğu için hesaplama analizleri için yapılan literatür çalışmalarına daha az rastlanmıştır. Hesaplama analizleri için geliştirilmiş temel modellerin karşılaştırmaları ve denektaşı testleri vardır. Fakat uzaktan algılama sensör mimarileri için geliştirilen derin öğrenme uygulamalarında literatürde vurgu az yapılmıştır.

Tez çalışmasında hesaplama analizleri ile gerçek zamanlı çalışma için incelenen ve odak çalışmaları oluşturan makine öğrenmesi tabanlı radar hedef tespitinde [1] CNN ile artan hesaplama karmaşıklığı ile mücadale için tasarlanan paralel hesaplamalar beşinci kısımda detaylı olarak anlatılmaktadır.

3.3. Sıkıştırılmış Algılama tabanlı Geliş Açısı Kestirimi

İkinci bölümde ayrıntıları verilen ADMM-LASSO tabanlı geliş açısı kestirimi için literatürde yapılan çalışmalar bu bölümde anlatılmaktadır. Geleneksel yöntemler için yapılan çalışmalara temel oluşturan ilgili referanslar ikinci bölümde yer verildiğinden bu bölümde değinilmemiştir.

Sıkıştırılmış algılama tabanlı geliş açısı kestirimi çalışmalarında girdi olarak verilen algılama (ölçüm) matrisi alıcı yapısındaki donanım kaynak optimizasyonunu sağlamaktadır. İlgilenilen çalışmalarda, kullanılan ölçüm matrisi sayesinde hem donanım kaynak optimizasyonu sağlanmış olup hem de kestirim başarımı artırılmıştır [69, 70]. Ölçüm matrisi anlık olarak üretilmeyeceği varsayımıyla sensör sisteminin açılışında ve belli periyotlar ile güncelleneceği düşünülerek gerçek zamanlı çalışma hesaplamalarına katılmamaktadır. Fakat adaptif teknikler için avantaj sağlanacağı düşünülerek literatüre kazandırdığımız ve ilk defa olarak bir uygulaması sunulan ölçüm matrisi tasarımının paralel hesaplamaları altıncı bölümde anlatılmaktadır [20].

Sıkıştırılmış algılama teknolojisinin en çok ilgilendiği problem olan seyrek geriçatıma ait matematiksel özelliklerin incelendiği çalışmada yazarlar, geriçatım işlemleri için kararlılık, ölçüm hatalarına karşı prensibin sağlamlığını ve benzer alanlardaki problemlere uyarlanabilirliği tartışmışlardır [71]. Tamamlanmamış frekans örneklerini girdi alarak seyrek geriçatım yöntemiyle nesnenin yeniden oluşturulması sağlayan Candes ve diğerleri, seyrek geriçatım alanında ilhâm veren bir çalışmayı literatüre kazandırmışlardır. Yeniden oluşturmayı tamamlanmamış örneklerden nasıl yapıldığının anlatıldığı, süreksizliklerin belirtilen koşullara uyarlandığı ve dışbükey fonksiyonlarının aza indirildiği çalışmada yazarlar, seyrek geriçatımın matematiksel analizini yoğun bir şekilde yapmışlardır.

Baraniuk, görüntü ve video uygulamalarında görüntü geriçatım işlemlerinde sıkıştırılmış algılama ile iyileştirme önerilerini çalışmasında sunmuştur [72]. Yazar sıkıştırılmış algılama metodolojinin veri sıkıştırma, optimizasyon ve boyut azaltma problemlerinde kullanabileceğini önermiştir. Sıkıştırma algılama kullanımı için verilerin seyrek özellikte yer almasının elzem olduğunu belirtmiştir. Örneğin; N uzunluğunda bir vektör, sıkıştırma algılama probleminde kullanılacak ise K tane hücresi sıfır olmayan bir değere sahip olmalıdır. $N - K$ tane hücresi sıfır olan bir vektör, $K << N$ koşuluyla sıkıştırılabilmektedir. Yazar çalışmasında seyrek geriçatım problemini bir imge üzerinde görüntü yeniden oluşturma için çalışmıştır. Seyrek geriçatım yapmadan önce bir ölçüm matrisi tasarımıyla geriçatıma girdi sağlayacak öncül bilgiler sunmuştur. Ölçüm matrisi için sırasıyla tasarım adımları işletilmiştir. N tane örnek almak yerine $M < N$ koşuluyla M tane örnek alarak geriçatım problemi oluşturulur. Bu problemi çözmek için belirli bir rasgele sürece bağlı bir model ile MxN'lik ölçüm

matrisi tasarımı gerçekleştirmiştir. Yazar, ölçüm matrisi tasarımının Candes ve diğerlerinın literatüre kazandırmış olduğu prensiplere uygun şekilde yapılması gerektiğini belirtmiştir. Baraniuk, ölçüm matrisi tasarımından sonra doğrusal olmayan bir probleme dönüşen geriçatım işlemi için doğrusal programlama yöntemleri ile çözümler aramıştır. Çözümler için minimum ℓ_0-norm, minimum ℓ_1-norm ve minimum ℓ_2-norm geriçatım yöntemleri incelenmiş olup matematiksel analizleri verilmiştir. Bu çalışma seyrek geriçatım probleminin sıkıştırılmış algılama metodolojisi içinde çözümünü arayan değerli bir kaynak olmuştur. Çalışmanın literatüre sunduğu en önemli çıktı, sıkıştırılmış algılama ile gerçekleşen çözümde doğrusal olmayan problemler için sıkça kullanılan en küçük kareler (least squares) optimizasyonu yerine ℓ_1-norm optimizasyon tabanlı çözümlerin gerekliliğinin açıklanmasıdır.

Geliş açısı kestirimini içeren kaynak lokalizasyon probleminde seyrek sinyal geriçatım yaklaşımını sunan çalışmada yazarlar, dar bant ve geniş bant senaryolarında kullanabilecek bir optimizasyon prosedürünü önermişlerdir [73]. Malioutov ve diğerleri, geleneksel yöntemlerden biri olan MUSIC yöntemine göre karşılaştırdıkları başarımlar ile çözünürlük artışı, gürültüye karşı sağlamlık ve sınırlı zaman alıntı sayısı gibi avantajları seyrek geriçatım optimizasyon prosedürü ile sağlamışlardır. Geliştirdikleri optimizasyon yöntemini ℓ_1-SVD yöntemi şeklinde bir ifadeyle belirtmişlerdir. Bu çalışma özellikle seyrek geriçatım modeli sunan sıkıştırılmış algılama tabanlı geliş açısı kestirim modelleri için temel bir referans olmuştur. Geleneksel yöntemlere göre önerilen yöntemin algoritmik başarım artışı sağlaması önemli bir kazanım olarak literatüre geçmiştir. Yazarlar kaynak lokalizayon problemi için artan hesaplama maliyetlerini azaltabilmek adına açgözlü [74] seyrek sinyal yöntemlerin uygulanabileceğini gelecek çalışma olarak önermişlerdir.

Geleneksel anten dizileri yerine sıkıştırılmış mimariye sahip bir anten dizisi tasarlanan çalışmada, düşük donanım karmışılığına sahip alıcı sisteminde geliş açısı kestirimi problemi için eniyilenmiş bir yaklaşım sunulmuştur [75]. Yazarlar sıkıştırma işlemi için bir sistem mimarisi önermişlerdir. Bu mimaride azaltılan anten sayısı ile alıcı yapısı, analog birleştirme (beamforming) yöntemleri ile etkin hale getirilmiştir. Bu kapsamda, uzamsal korelasyon fonksiyonuna dayanan ve kapalı formda çözümü mümkün olan düşük karmaşıklıklı şema ve Cramer-Rao Bound (CRB) [76] sınırlanmasıyla yanlış algılanma olasılığını azaltma şeklinde iki temel yaklaşım ele alınmıştır [77]. Benzetimler sonucunda rasgele seçilmiş ve

birleştirilmiş anten yapılarına göre önerdikleri yöntem üstünlük sağlamıştır. Çalışma geliş açısı kestiriminde sıkıştırılmış anten dizisi ile donanım karmaşıklığı azaltılırken ortaya çıkan geriçatım problemi için çözüm önerileri sunmuştur. Seyrek yaklaşımlar ile karşılaştırma yapılması önemli bir katkı sağlamış olup klasik yöntemler ile yapılan karşılaştırma deneysel sonuçlar ile net gösterilmediği için çalışmada bütünsel karşılaştırım eksik kalmıştır.

Bu tez çalışmasında LASSO yöntemi ile geliş açısı kestirimi için tanımlanan seyrek geriçatım problemine yönelik optimizasyon tabanlı bir çözüm sunan yöntem incelenmektedir. LASSO yöntemi optimizasyon probleminde katsayıların mutlak değerlerinin toplamı doğrultusunda, karesel hataların toplamını en aza indirerek çözümü gerçekleştirir. LASSO yöntemi literatürde sıkça gördüğümüz kestirime dayalı çalışmaların yanı sıra regresyon problemlerinde ve istatiksel problemlerde de çözüm sunmaktadır [78–80]. LASSO yöntemi geliş açısı kestirim uygulamalarında sıkıştırılmış sensör dizisinin oluşturduğu doğrusal olmayan problemi ℓ_1-norm minimizasyon yaklaşımı ile çözmektedir. Sıkıştırılmış algılama tabanlı geliş açısı kestirimi çalışmaları için bir genel gözden geçirme sunan çalışmada yazarlar, geliş açısı kestirimindeki bazı belirsiz konulara dikkat çekerek geniş bant ve dar bant geliş açısı kestirimleri için gelişmiş sensör dizisi tasarımlarına yönelik katkıları sunmuşlardır [81]. Shen ve diğerleri, dar bant ve geniş bant sinyal modellerini iki ayrı başlıkta inceleyerek sıkıştırılmış algılama tabanlı geliş açısı kestiriminin tek bir zaman alıntı ile çoklu zaman alıntısı durumlarındaki ℓ_1 minimizasyon tabanlı LASSO formulasyonlarını göstermişlerdir. Hem geniş bant hem de dar bant için yaptıkları karşılaştırmalı benzetimlerde geleneksel yöntem olan MUSIC yöntemine göre düşük sinyal gürültü oranlarında kestirim doğruluğu başarım artışını sıkıştırılmış algılama ile sağlamışlardır. Sıkıştırılmış algılama tabanlı yöntem geniş bant senaryosunda dar bant senaryosuna göre daha yüksek kestirim doğruluğu vermiştir. Çalışmada detaylı bir literatür gözden geçirmesinin yanı sıra sıkıştırılmış algılamanın geliş açısı kestiriminde etkili bir çözüm olduğu sunulmuştur.

Bu tez çalışmasında algoritma hesaplamaları için ilhâm kaynağı olan çalışmada, sıkıştırılmış algılama tabanlı geliş açısı kestirimleri için uyarlamalı teknikler incelenmektedir [70]. Çalışmanın yazarı Kılıç, geliş açısı kestiriminde ölçüm matrisinin oluşumunda değişen çevresel etkileri sisteme geri besleyerek kestirim başarımını arttırmışlardır. Ölçüm matrisi tasarımı işlemi, çoğu çalışmada bir kez veya periyodik sürelerde (kalibrasyon sırasında) yapılırken

bu çalışmada geliş açısı kestirimi bloğuna her hedef geldiğinde yeniden yapılmıştır. Böylece hedef takip algoritmalarındaki çıktıyı geliş açısı kestirimine besleyen bir bakış açısı ile uyarlamalı bir yaklaşım literatüre sunulmuştur. Geliş açısı kestirimi için LASSO yaklaşımı kullanılmıştır. Ölçüm matrisi tasarımına literatürdeki birçok çalışmada karşılaşmakla birlikte yazarın çalışmasında ayrışan en temel nokta geliş açısı kestirimini adaptif ölçüm matrisi ile daha yüksek doğrulukla çalıştırılması olmuştur. Ayrıca LASSO yönteminin klasik yöntemlere göre artan hesaplama karmaşıklığı azaltmak için ADMM kullanılarak kestirim için yapılan optimizasyon işleminin daha hızlı yakınsamasını sağlamışlardır. Yazarın çalışmasını referans alan bu tez çalışmasındaki konulardan birisi, ADMM yönteminin LASSO tabanlı optimizasyon problemini algoritmik olarak iyileştirmesinde kullanılması sonucu geliş açısı kestirimi hesaplamasına ait paralel programlama tasarımı ile yüksek başarımlı hesaplamaları gerçekleştirmektir.

Literatür incelemelerinde ADMM yönteminin çoğu optimizasyon probleminde yakınsama hızını artırdığı ve doğruluk kaybının yüksek yineleme sayıları kullanılarak azaltılabildiği görülmektedir. ADMM yönteminin istatiksel öğrenme ve optimizasyon alanında çok sayıda probleme etkili ve verimli bir çözüm olabileceğini belirten çalışmada Boyd ve diğerleri, ADMM yönteminin dağıtılmış dışbükey optimizasyon işlemleri için ve özellikle makine öğrenimi alanında ortaya çıkan büyük ölçekli problemlere uygun çözümler sunduğunu göstermiştir [82]. Yazarlar, problemlere ait algoritmanın teorisini sunduktan sonra kovaryans seçimi ve destek vektör makineleri olmak üzere son zamanlarda revaçta olan çeşitli istatiksel uygulamaları tartışmışlardır. Wahlberg ve diğerleri, ADMM yöntemini esas alarak optimizasyon probleminin yapısından esinlenip verimli ve ölçeklenebilir bir optimizasyon algoritması türetmeye çalışmışlardır [83]. Yazarlar temel olarak konveks optimizasyonlar için ADMM yöntemini kullanmışlardır. Çok dönemli bir portföy yönetimi için gerçekleştirilen optimizasyon bu problemlere örnek olarak verilmiştir. Çalışmada ℓ_1-ortalama veya ℓ_1-varyans filtreleme yaklaşımları ile algoritma verileri bölümlenerek uygulanmıştır. Yaptıkları bölümleme ile genel optimizasyon çözücü SDPT3'e kıyasla 10000x hızlanma sağlamışlardır [84].

ADMM yöntemi, içerdiği adım boyutu ve yineleme sayısı parametreleri sayesinde her bir yineleme adımında işlemleri gerçekleştirip güncel sonuçlar üretir. Adım boyutu parametresi

yakınsamanın hızı, tahminlemenin doğruluğu ve hesaplama karmaşıklığı açısından etkili bir faktördür. Bu parametrelerin optimum seçimini konu alan makalede, çalışmanın yazarları parametrelerin yakınsama sürelerine olan nicel etkilerini incelemişlerdir [85]. Ghadimi ve diğerleri, yakınsama faktörünü en aza indiren optimum parametreleri ℓ_2-norm minimizasyon ve kısıtlı ikinci derece programlama (constrained quadratic programming) bağlamında bulmuşlardır. Analitik çözümlemelerden sonra test verileri üzerinde, optimum parametrelere sahip ADMM algoritmasının literatürdeki yöntemlere göre başarımı karşılaştırılmıştır. Yazarlar, literatüre optimum parametre seçimi için referans bir makale sunmuşlardır.

ADMM tabanlı çözümler sıkıştırılmış algılama uygulama alanlarının çoğunda kullanılmaktadır. Sıkıştırılmış algılamanın yoğun kullanıldığı görüntüleme teknolojilerinde, manyetik rezonans görüntülemedeki geriçatım işlemini sıkıştırılmış algılama ile çözen çalışmanın yazarı ADMM yöntemini kullanarak gerçekleştirdiği çözümü literatüre sunmuştur [86]. Güngör, çalışma kapsamında ADMM ile tek kontrastlı geriçatım etkinliğini diğer tek kontrastlı yöntemlerle karşılaştırmıştır. Çalışmada yazar görüntü ile birlikte seyrekleşen dönüşümü bulmak için ortak sözlük öğrenimini esas alan bir yöntem sunmuştur. Çalışmada kullandığı sözlük öğrenme temelli yöntem görüntü kalitesine artırmasına rağmen yüksek hesaplama maliyetini ortaya çıkarmıştır. Yazar manyetik rezonans görüntülemede kullandığı ADMM yöntemini gelecek çalışmalarda paralel hesaplamaya uygun hale getirerek performans iyileştirmeyi hedeflemiştir. ADMM yönteminin geliş açısı kestirimi dışında özellikle görüntülemedeki geriçatım tekniğinde kullanımı ve süper çözünürlük çalışmalarına etkisini göstermesi açısından faydalı bir çalışma olmuştur. Yazarın bahsettiği gibi hesaplama maliyetleri CPU-GPU hibrit paralel hesaplama tasarımı ile azalabileceği düşünülmektedir.

Akıllı trafik, şehir uygulamaları ile araçlar arası iletişim topolojisinde enerji verimliliğini esas alan tasarımlar hedeflenir. Araçlar ve mobil kullanıcılar bir uç mimarisi olarak düşünüldüğü takdirde uçta yer alan hesaplamanın enerji verimli olması fayda sağlar. Trafik ve araçlar arası ağda enerji verimliliğine katkı sağladığı belirtilen çalışmada yazarlar, enerji verimli iş yükü boşaltma problemini incelemişlerdir [87]. Zhou ve diğerleri, belirttikleri bu problemi optimizasyon problemi olarak ele alıp Liang [88] ve diğerlerinın iletişimde kaynak tahsisi için önerdikleri konsensüs ADMM yaklaşımı ile çözüm önermişlerdir. Önerdikleri çözümde

düşük karmaşıklığa sahip bir mimari ile enerji verimli araç uç hesaplama sunmuşlardır. Konsensüs ADMM ile dağtık ve paraleleştirmeye uygun bir tasarım sunulmuş olup gerçek trafik topolojisine yakın benzetimler gerçekleştirmişlerdir. Çalışma değerlendirildiğinde, ADMM yönteminin jenerik olarak çoğu optimizasyon problemine uyarlanabileceği özellikle kaynak tahsisi problemi içeren 5G/6G ve akıllı şehir uygulamalarında bir optimizasyon çözüm alternatifi olacağı düşünülmektedir. ADMM yönteminin bu çalışmada da olduğu gibi konveks optimizasyon algoritmalarının hesaplama maliyetlerini düşürerek daha düşük karmaşıklığa sahip hesaplama imkanı sunması ve paralel hesaplamaya uygun hale getirilmesi tasarım açısından verimlilik sağlamaktadır.

Manyetik Rezonans Görüntüleme teknolojisinde görüntü seyrek çatım için kullanılan optimizasyon metodları modifiye edilerek yenilikçi bir görüntüleme tekniği olan Manyetik Parçacık Görüntüleme [89] (MPI) alanında da kullanılmaktadır [90]. MPI, süpermanyetik özelliğe sahip demiroksit nanoparçacıkların kullanımıyla dağılım tespit eden bir medikal görüntüleme tekniğidir. MPI tekniğinde geriçatım için tıpkı kestirim problemlerinde karşımıza çıkan ölçüm matrisi benzeri sistem matrisi tasarımı tekniği bir alternatif çözüm yaklaşımıdır. İlbey ve diğerleri, MPI tekniğindeki üç boyutlu (3B) görüntüleme problemi için bir optimizasyon çözümü tasarlayıp ADMM yöntemi ile görüntüyü verimli bir şekilde yeniden oluşturmuşlardır [91]. Yazarlar görüntü geriçatım için geliştirmiş oldukları ADMM tabanlı çözüm yöntemini paralel hesaplamalar ile gerçekleyerek görüntüleme süresini önemli ölçüde iyileştirmişlerdir. Artan ADMM iterasyon sayısına bağlı olarak görülen performans iyileştirmesinde NVIDIA GTX 1080 GPU birimiyle Intel Xeon CPU E5-2637 v4 birimine kıyasla 300 adet yinelemede 37x kat hızlanma elde etmişlerdir. Çalışmada 3B görüntülemenin gerçek zamanlı bir kullanımda başarı sağlayacağı belirtilmiştir. ADMM yönteminin grafik işlemciler ile başarılı bir şekilde paralelleştirmesi hesaplama çalışmaları açısından motivasyon sunmuştur. Fakat çalışmadaki CPU ve GPU mimarileri daha ayrıntılı anlatılarak hızlanma analizi daha vurgulu yapılabilirdi. Ayrıca CPU'daki başarımı artırmak için çekirdek seviyesinde ve çekirdekler arası veri-iş paylaşımı ile başarım artışı sağlanabilirdi.

Sıkıştırılmış algılama tabanlı geliş açısı kestirim işleminde nadir çalışmalarında biri olan makalede yazarlar, LASSO optimzasyon yönteminin benzeri olan temel takip gürültü giderme (basis pursuit denoising, BPDN) problemini esas alan tasarımları ile kestirim için

optimizasyon çözümünü incelemişlerdir. [92]. Yazarlar, BPDN probleminin getirmiş olduğu yüksek hesaplama maliyetini azaltmak için ADMM yöntemini kullanmışlardır. Sadece 8 anten elemanın olduğu bir yapı için 180 adet açı ızgara noktası ile modeli oluşturmuşlardır. ADMM yöntemi ile BPDN tabanlı yönteme göre 17x hızlanma sağlamışlardır. Çalışmanın eksik yanları ise hesaplama tasarımlarının verilmemiş olması ve hızlanmanın hangi adımda gerçekleştiğinin net gösterilmemesidir.

3.3.1. Tartışma

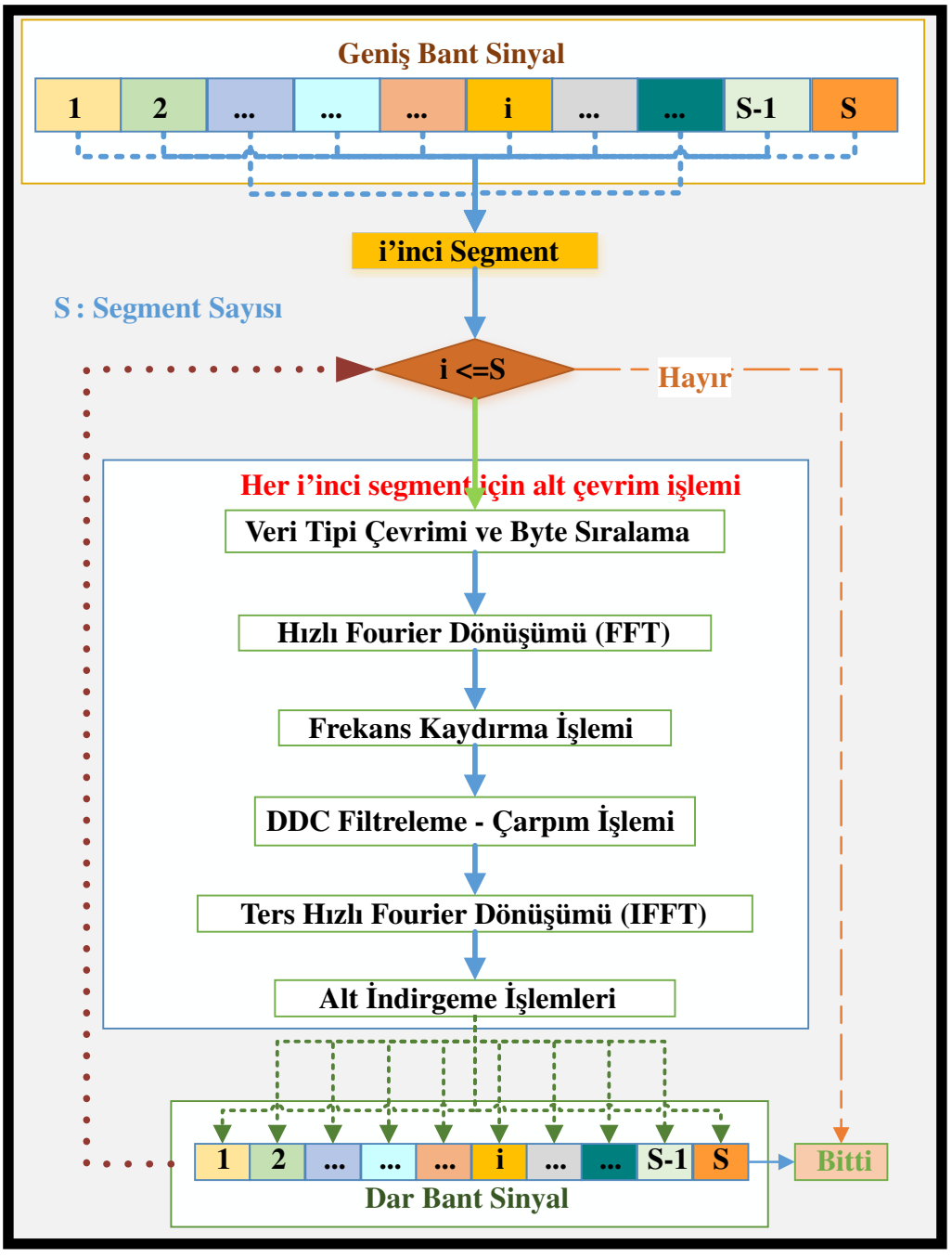
Sıkıştırılmış algılama ile geleneksel örnekleme ve sinyal işleme yöntemlerine daha düşük donanım karmaşıklığına sahip algılayıcılar tasarlamak mümkün hale gelmiştir. Sinyal modelindeki seyreklik özelliği ile sıkıştırma yapılarak daha az örnekleyici, yükseltgeç ve alt birimler kullanılarak sensörlerin hem ağırlığı hem de maliyetleri azaltılabilir olmuştur. Sıkıştırma durumunda sinyali geriçatım ile elde etmek için oluşan doğrusal olmayan problemin çözümüne yönelik çaba sarfedilmektedir. Çözüm için doğrusal programlama ile konveks optimizasyon yapılır. LASSO yöntemi ile geliş açısı kestirimi problemi hatanın minimizasyonu limitiyle bir optimizasyon adımı haline getirilir. Literatürde gözlemlenen çalışmalarda LASSO kestirim doğruluğu arttırsa da sistemin hesaplama karmaşıklığını arttırır.
LASSO çözümünün verimliliğini artırmak için LASSO'nun uygulama alanlarında biri olan medikal görüntülemedeki görüntü geriçatım çalışmaları incelenmiştir. Bu çalışmalarda görüntü geriçatımı için uygulanan LASSO ya da BPDN çözümleri ADMM ile daha verimli hesaplanmaktadır. ADMM tekniği görüntüleme alanında çok sayıda çalışılmasına rağmen geliş açısı kestirimi probleminde literatürde yeterince karşılaşılmamıştır. Bu tez çalışmamızda ADMM tekniğini esas alan sıkıştırılmış algılama tabanlı geliş açısı kestiriminin CPU ve GPU başarımlarını göstermiş olup ve mobil sensörler için sunacağı kazanımlar gösterilerek literatüre katkı sağlanması motivasyon oluşturmuştur. Ayrıca, seyrek geriçatım probleminde temel bir adım olan ölçüm matrisi tasarımı için yapılan çalışmalar incelenmiştir. Tez çalışmasında, ölçüm matrisi için yapılan hesaplama çalışmaları ve değerlendirmeler altıncı bölümde belirtilmektedir.

4. STFT TABANLI SAYISAL ALT ÇEVRİMİ

Bu bölümde, sayısal alt çevrimi işleminde geniş bant veri için büyük uzunluklarda FFT almak yerine önerdiğimiz yöntem olan veriyi eş uzunluktaki küçük segmentlere bölen OS-FFT yaklaşımı tabanlı hesaplamaya ait detaylı açıklamalar ve deneysel çalışmaların sonuçları anlatılmaktadır. Sayısal alt çevrimi için algoritma modeli ve veri akışı ayrıntılı olarak gösterilerek hesaplama analizleri sunulmaktadır. Hızlanma karşılaştırmaları için CPU koşut hesaplama başarımları ile GPGPU başarımları incelenmektedir.

4.1. Algoritma Modeli

Sayısal alt çevrimi için ikinci ve üçüncü bölümde detaylı bir şekilde literatür çalışmaları ve önerdiğimiz yöntemin motivasyonu anlatılmaktadır. Bu bölümde ise önerdiğimiz hızlı evrişim ile sayısal alt çevrimi yönteminin algoritma akışı ve girdi-çıktı veri akışı gösterilmektedir. Geniş bant bir sinyal analog dijital çevrim ile sayısallaştırıldıktan sonra sayısal sinyal üzerinde işleme blokları uygulanır. Haberleşme sistemlerinde çok taşıyıcı içeren modülasyon tekniklerin kullanımında yüksek bant genişliklerine ihtiyaç duyulmaktadır [93]. Bununla birlikte darbe Doppler radar ve frekans modülasyonuna sahip sürekli radarlarda haberleşme sinyal işleme bloklarında bilinen merkez etrafında geniş bant sinyalden filtreleme yapılır. Bu filtreleme sonucunda sinyal yeniden örneklenerek geniş banttan dar banda indirgenir. Benzer şekilde elektronik harp alıcılarında dar bant analizler yapılarak sinyale ait karakteristik özellikler çıkartılır. Tez çalışmasında geniş banttan dar banda indirgemenin yer aldığı bütün sensör mimarilerindeki yeniden örnekleme, hızlı evrişim yöntemi ile ele alınır. Doğrusal evrişim tabanlı CIC filtreleme işleminin FPGA-DSP mimarilerinde yoğun olarak kullanıldığı üçüncü kısımda ayrıntılı olarak aktarılmaktadır. Buna rağmen geniş bant kanallaştırma işlevinin olduğu sensör mimarilerinde FPGA-DSP yerine GPGPU kullanımının gerçek zamanlı sistemler için daha verimli ve hızlı olduğu görülmektedir [32]. Sayısal alt çevrimi için önerdiğimiz yöntem olan STFT tabanlı sayısal alt çevrimi için tasarladığımız algoritma akış modeli Şekil 4.1'de gösterilmektedir.



Şekil 4.1: STFT tabanlı Sayısal Alt Çevrimi Algoritma Hesaplama Akış Modeli

Şekil 4.1'de görüldüğü üzere geniş bant sinyal segmentlere bölünerek, daha küçük FFT uzunluğu sayesinde, bellek verimli akışla ardışık bir şekilde dar bant sinyale indirgenir. Algoritma akışında büyük uzunluklu FFT işleminin yapılması bellek kaynak miktarını arttırmakta ve

önbellek kullanım verimliliğini azaltmaktadır. FFT işlemi çalışmamızın ikinci kısmında anlatıldığı üzere örtüşen kaydet prensibi ile daha verimli yapılır. Bu sayede küçük uzunluklu FFT ile işlemlere ait veriler önbellekte daha çok kullanılarak zamansal ve uzaysal yerelliğin verimi artırılmaktadır. Şekil 4.1'de yer alan algoritma modelinin sözde kodu Algoritma 1'de gösterilmektedir.

Algoritma 1 Önerdiğimiz STFT tabanlı Sayısal Alt Çevrimi için Sözde(Pseudo) Kod

X : Geniş Bant Örnekleme Verisi
FFT_X : X'in FFT'si
Y : FFT_X Verisinin Dairesel Kaydırılmış Çıktısı
H : DDC Yeniden Örnekleme Filtresi'nin FFT'si
P : Filtre Çarpımının Sonucu
$IFFT_P$: Filtre Çarpımının IFFT'si
Z : Dar Bant Sinyal (IQ) Verisi
D : Alt İndirgeme Oranı
S : DDC Segment Sayısı
N : FFT Nokta Sayısı
M : DDC Segment Uzunluğu

for i = *1:S* **do**
  Adım 1 : Örtüşen Veriyi içeren Segment Verisini Al, X_i
  Adım 2 : FFT_{X_i} = X_i'nin FFT İşlemi
  Adım 3 : Y_i = FFT_{X_i} için Frekans Kaydırma
  for j = *1:N* **do**
    Adım 4 : Filtre Çarpımı : $P_{i,j} = Y_{i,j} \cdot H_j$
  end for
  Adım 5 : $IFFT_P$ = P_i'nin IFFT İşlemi
  Adım 6 : IFFT Sonucu Örtüşen Başlangıç Kısmını Atma İşlemi
  for k = *1:M* **do**
    Adım 7 : Alt İndirgeme İşlemleri: $Z_{i,k}$ = $IFFT_P$(i,((k-1)*D+1))
  end for
  Adım 8 : Ardışık Segmenti İşlemeye Geç
end for

Sayısal alt çevrimi işleminde hızlı evrişim yöntemine dayalı yeniden örnekleme filtreleme işlemi yapılmaktadır. FFT tabanlı filtreleme işlemi herhangi bir bölütleme içermezse girdi verisinin bütün uzunluğu kadar nokta sayılı FFT işleminin yapılması gerekir. Fakat bölütleme ile örtüşen kaydet/at FFT (overlap save/discard, OS-FFT) işlemi yapılarak bellek kazancı ve önbellek yerellilik verimi artırılmaktadır. Algoritma 1 ve Şekil 4.1'deki akış modeli birlikte incelendiğinde, her bir segment verisi için adım adım işlemler yapılarak geniş bant ADC

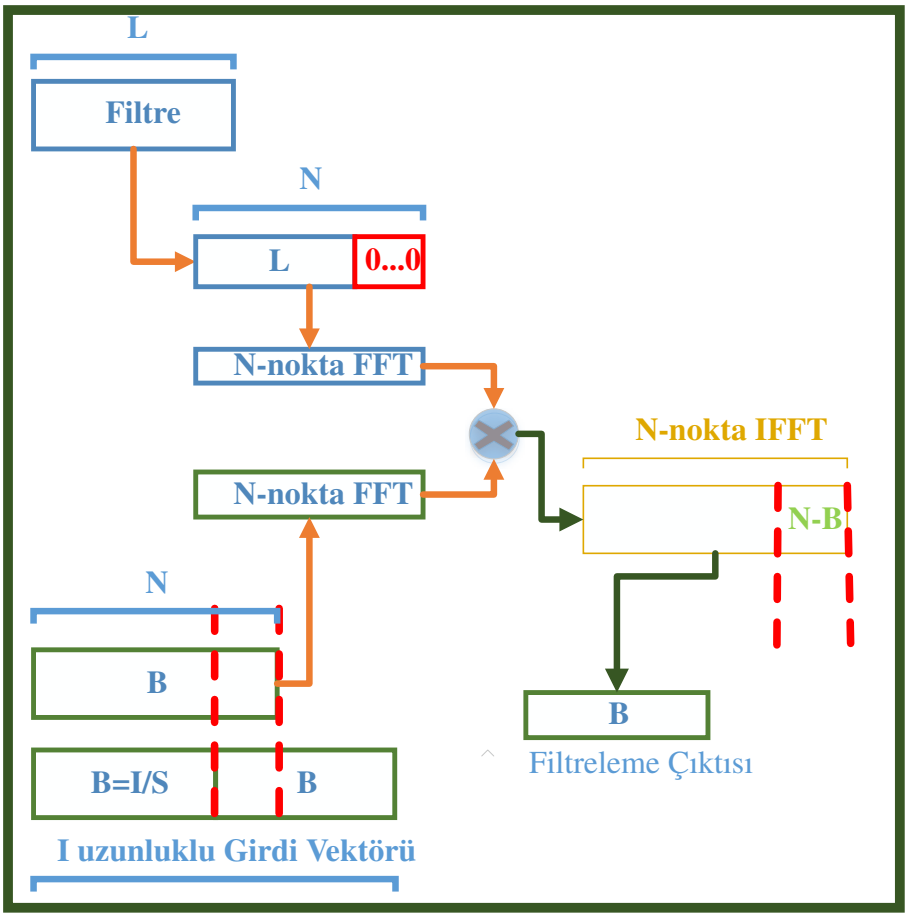
verisinden dar bant IQ (inphase-quadrature) karmaşık veri elde edilir. Segment sayısı kadar iterasyon tamamlandığında sayısal alt çevrimi işlemi sona ermiş olup elde edilen dar bant IQ veri detaylı analiz, tespit, takip ve parametre çıkarımı gibi sinyal işleme bloklarında işlenebilmektedir.

Algoritma adımlarını incelemeden önce OS-FFT tabanlı filtreleme yönteminin matematiksel analizinin sunulması hesaplama mimarisinin anlaşılırlığını pekiştirmektedir. Yeniden örnekleme işleminde büyük bir uzunluklu girdi için filtreleme işlemi yapılırken OS-FFT tabanlı filtreleme kullanımında girdi eş uzunluklu segmentlere bölünür. Yeniden örnekleme filtreleme işleminde;

- Filtre uzunluğu L, L tek sayı,
- Girdi uzunluğu I, $L << I$ varsayımı ile,
- Çıktı uzunluğu O, $O = I + L - 1$,
- Grup gecikmesi G, $G = L/2$ değerleri ile ifade edilir.

Filtrelemeden kaynaklı grup gecikmelerini atmak için çıktı verisinin başından ve sonundan $L/2$ kadar veri silinir. OS-FFT tabanlı filtrelemede ise örtüşen kısımlar filtrelemeye sokulduktan sonra karşılık gelen çıktı sonuçları atılır. İşlemlerde yer alan girdi uzunlukları ve filtre uzunluğu FFT uzunluğu olan N'e göre ayarlanır. Filtreleme sırasında OS-FFT prensibini gerçekleştirmek için Şekil 4.2'de gösterildiği üzere sırasıyla aşağıdaki maddeler gerçekleştirilir:

- Filtre katsayılarının FFT'si alınmadan önce $N - L$ kadar filtrenin sonuna 0 eklenir.
- Segment sayısı (S) kadar segmentlere bölünen girdi verisinin her bir segmenti önü ve arkası ardışık segmentlere örtüşecek şekilde uzunluğu N yapılır.
- Her bir segmentin boyu $B = I/S$ kadar olduğundan, $N - B$ kadar örtüşen veri her bir segment için filtreleme işleminde kullanılır.



Şekil 4.2: OS-FFT tabanlı Filtreleme İşlem Akışı

Şekil 4.2'de akış incelendiğinde OS-FFT tabanlı filtrelemede ardışık segmentler için hızlı evrişim sırasındaki yapılandırma sunulmaktadır. IFFT sonucunda elde edilen çıktı vektöründen örtüşen kısımlar atılmaktadır. Algoritma adımları maddeler halinde açıklanmaktadır:

- **Adım 1** : I uzunluğundaki veri her ardışık segment arasında $N - B$ uzunluklu segmentlere bölünerek işleme alınır.
- **Adım 2** : B uzunluğundaki segmentin sonuna ardışık segmentin başından eklenen veri ile N uzunluklu X vektörü olur. N nokta FFT işlemi X vektörü üzerinde yapılır. N değeri 2'nin pozitif tamsayı kuvveti olmalıdır. N değeri seçilirken $N > (B + L - 1)$ matematiksel eşitsizliği sağlayan en küçük değer seçilir.

- **Adım 3 - Frekans Kaydırma** : FFT işlemi sonucunda elde edilen FFT_X verisi üzerinde frekans kaydırma işlemi frekans uzayında yapılmaktadır. Frekans kaydırma işlemi normalde filtrelemeden önce geçiş banttan temel banda kaydırma amacıyla öncelikli olarak yapılır. Literatürdeki karşılığı nümerik kontrollü osilatör işlemi olan frekans kaydırma için Eşitlik 6'daki gibi işlem yapılmaktadır.

$$y = x \cdot exp(-j2\pi f_c t) \tag{6}$$

Eşitlik 6'da x geniş bant ADC verisini, f_c geçiş bantta yer alan sinyalin merkez frekansını, t ise örnekleme periyodu aralıklı zaman vektörünü ve y ise temel bant IQ verisini temsil etmektedir. Bu işlem yapılırken karmaşık veri içeren üstel fonksiyonun IQ bileşenlerini cosinüs ve sinüs çarpımları ile hesaplanır. Eşitlik 7'de karmaşık üstel fonksiyonun trigonometrik dönüşümü gösterilmektedir.

$$exp(-j2\pi f_c t) = cos(2\pi f_c t) - jsin(2\pi f_c t) \tag{7}$$

Trigonometrik fonksiyonlar, içerdikleri Taylor serisi açılımları nedeniyle işlem süreleri uzun olduğundan gerçek zamanlı sistemlerde sonuçları hızlı çıkaran yöntemler ile hesaplanmaktadır [25]. Frekans kaydırma işlemi için trigonometrik fonksiyonlar kullanmak yerine halihazırda frekans uzayında olan veriye Eşitlik 8'de gösterildiği uzunluk kadar dairesel kaydırma uygulayarak yakınsayan çıktı elde edilebilmektedir [33]. N, FFT nokta sayısını ifade ederken, F_s, geniş bant sinyalin örnekleme frekansını ve N_s ise dairesel kaydırma uzunluğunu ifade etmektedir.

$$N_s = round(f_c/F_s) \times N \tag{8}$$

Frekans uzayında dairesel kaydırma işlemi sayesinde frekans kaydırma işlemi, DDC filtreleme sırasında yapılarak daha verimli bir şekilde hesaplanabilmektedir.

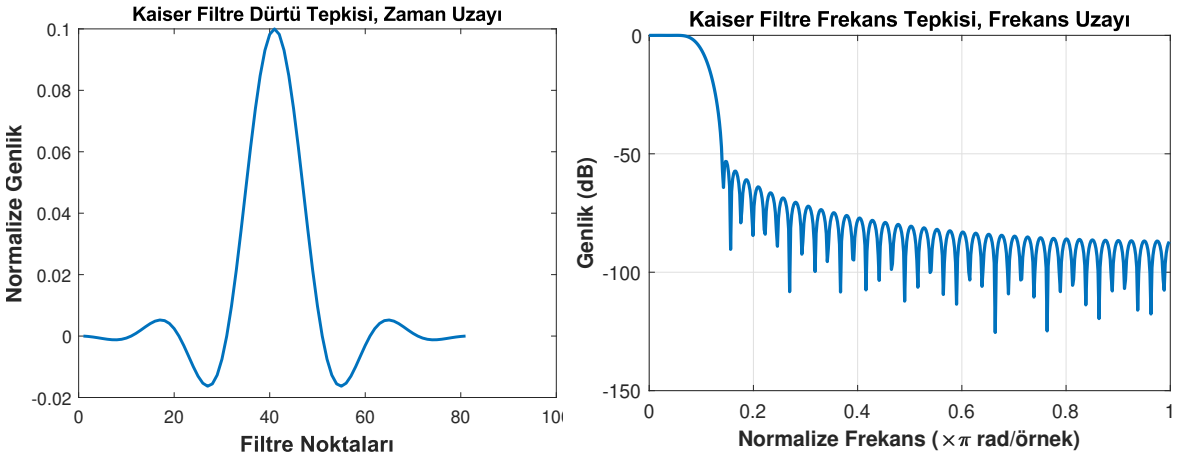
- **Adım 4** : Filtreleme işlemine başlamadan önce FFT'si alınmış filtre katsayıları ile frekans kaydırma işlemleri sonucunda elde edilen veri frekans uzayında çarpılır.

- **Adım 5**: Hızlı evrişim işleminin son adımı olan frekans uzayından zaman uzayına dönüşüm IFFT sayesinde gerçekleşir. IFFT işlemi N nokta için yapılır.
- **Adım 6**: OS-FFT tabanlı filtreleme yönteminde örtüşen kısımların zaman uzayında atılması bu adımda gerçekleşir.
- **Adım 7**: Alt indirgeme oranı kadar atlama ile geniş bant veri dar banda indirgenir.
- **Adım 8**:Ardışık segmente geçiş yapılır.

Filtre Katsayıları: Yeniden örnekleme işleminde filtreleme işlemi veri performansı açısından kritik önem arz etmektedir. Yeniden örnekleme sırasında *Kaiser* filtre tabanlı alt indirgeme filtresi kullanılmaktadır [6]. Filtre uzunluğuna etki eden faktörler filtre seviyesi (N_{ord}) ve alt indirgeme (D) değerleridir. Eşitlik 9'da görüldüğü üzere filtre uzunluğu tek pozitif sayı olacak şekilde hesaplanmaktadır.

$$L = 2N_{ord}D + 1 \tag{9}$$

Alt indirgeme sırasında kullanılan *Kaiser* filtrenin zaman ve frekans uzayında özelliklerinin gözlemi için filtre seviyesi 4 ve indirgeme oranı 10 olan bir alt indirgeme filtresine ait dürtü tepkisi (impulse response) Şekil 4.3(a)'da ve frekans tepkisi Şekil 4.3(b)'de sunulmaktadır.



(a) Dürtü Tepkisi

(b) Frekans Tepkisi

Şekil 4.3: DDC Alt İndirgeme Filtresi Karakteristikleri

Şekil 4.3(a) ve Şekil 4.3(b)'de sunulduğu üzere DDC işleminde kullanılan filtre bir alçak geçirgen filtredir (LPF). Alçak geçirgen filtrenin ancak bir temel bant frekansta kullanılması mümkündür. Bu yüzden frekans uzayında filtreleme öncesinde Adım-3'te yapılan frekans kaydırma işlemi LPF yapılmasına imkân kılmaktadır. STFT tabanlı sayısal alt çevriminde algoritmaya ait bütün alt adımlar filtrelemenin sistematik şekilde yapılmasını sağlamaktadır.

4.2. Hesaplama Analizi

Hesaplama analizi kapsamında STFT tabanlı sayısal alt çevrimi için bellek kullanımı ve hesaplama karmaşıklık analizi yapılmaktadır. Bellek kullanımı için Çizelge 4.1'deki gibi adım adım analiz gerçekleştirilmektedir.

İşlem Adımı/ Bellek Kullanımı	**Tek Segment Bellek Kullanımı**	**Tüm Segmentler Bellek Kullanımı**	**Sadece FFT ile Filtreleme**
Adım 1 - Segmenti Al	*2N*	*2N*	*2I (I ∼NS) = 2NS*
Adım 2 - FFT İşlemi	*2N*	*2N*	*2NS*
Adım 3 - Frek. Kayd.	*2N*	*2N*	*2NS*
Adım 4 - Filtreleme	*4N*	*4N*	*4NS*
Adım 5 - IFFT İşlemi	*2N*	*2N*	*2NS*
Adım 6 - Örtüşen Atma	*2B*	*2B*	*∼2BS*
Adım 7 - Alt İndirgeme	*2M*	*2MS*	*2MS*
Toplam	***12N + 2B + 2M***	***12N+2B+2MS***	***12NS+2BS+2MS***

Çizelge 4.1: STFT tabanlı DDC İşlemi - Bellek Kaynak Kullanımı Değerleri

Çizelge 4.1'deki semboller Algoritma 1'de ve Şekil 4.2'de yer alan semboller ile benzerlik göstermektedir. Çizelge 4.1'e göre bellek kullanım analizi Çizelge 4.2'deki ve 10 ms'lik veri işleme periyodunda nümerik değerlere göre yapılırsa;

- Tek Segment ile 12N+2B+2M : 12*131072 + 2*102400+2*32 = 1777728 örnek,
- Toplam Segment Kullanımı ile 12N+2B+2MS : 1779044 örnek,
- FFT Filtreleme ile işlem yapılırsa, 1777728 * 20 örnek kullanıldığı görülmektedir.

DDC İşlem Uzunlukları ve Parametreler	Semboller	Değerler
Geniş bant örnekleme hızı	F_S	204.8 MHz
Veri işlem periyodu	-	10 ms
Girdi örnek sayısı	I	2048000
Filtre seviyesi	N_{ord}	4
Dar bant örnekleme hızı	F_{S_d}	64 KHz
Alt indirgeme faktörü	D	3200
Filtre uzunluğu	L	2*4*3200 + 1 = 25601
Segment Sayısı	S	20
Segment Girdi Uzunluğu	B	102400
FFT Nokta Sayısı	N	128*1024 > (102400+25600)
Örtüşen Veri Uzunluğu	$N - B$	28672
Segment Çıktı Uzunluğu	M	32

Çizelge 4.2: STFT tabanlı DDC İşlemi için Parametreler

Çizelge 4.1'de analize bakıldığında OS-FFT yerine tüm girdi uzunluğunun içerildiği ve segmentlerin olmadığı FFT tabanlı filtreleme kullanımı takdirinde **20x** daha fazla bellek kaynağı tüketilmektedir.

Karmaşık kayar nokta veri (32-bit floating point) tipinin kullanıldığı bir ortamda OS-FFT ile 1779044*4 $\sim$ 7 MB veri boyutu ile işlem yapılmaktadır. Özellikle Intel CPU mimarilerinde akıllı önbellek yapılarının ortalama 8 MB boyutuna sahip olduğu düşünüldüğünde OS-FFT tabanlı filtreleme önbellek kullanımını artıran bir yöntem olduğu görülmektedir [94].

4.2.1. Hesaplama Karmaşıklıkları

Hesaplama karmaşıklığı analizi her bir adım için yapılıp ölçü birimi olarak çarpım-toplama (MAC) işlem sayısı parametresi kullanılmaktadır. Algoritma 1'de her bir adım için hesaplama durumuna bakıldığında Adım 2 (FFT), Adım 4 (Filtre Çarpımı) ve Adım 5 (IFFT) kısmında çarpım ve toplama işlemleri yer almaktadır. FFT işleminin DFT işlemine göre getirdiği kazanç dikkate alınarak ilgili adımlar için MAC işlem sayısı hesaplanmaktadır:

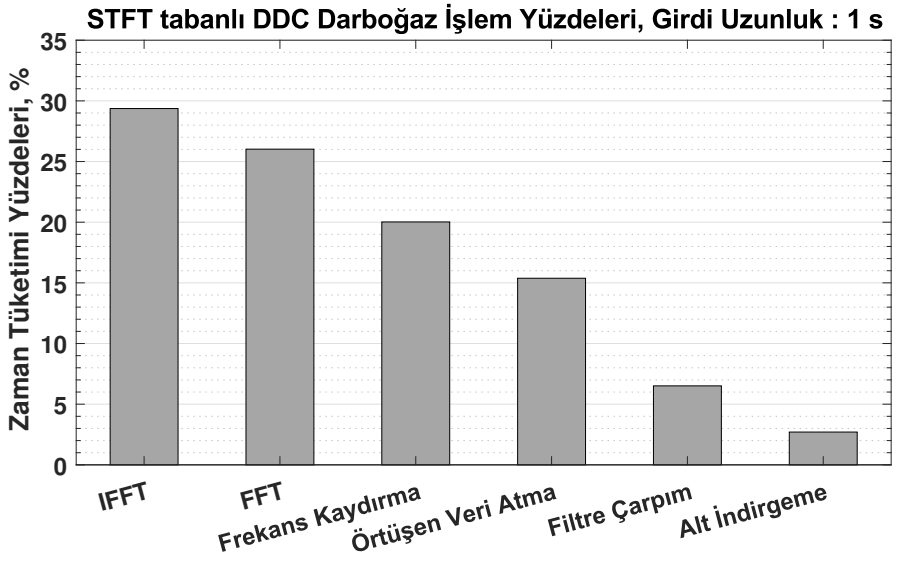
- Adım 2 : $(N/2)log_2N$ çarpım, $Nlog_2N$ toplama $\rightarrow Nlog_2N$ MAC işlem yapılır.
- Adım 4 : N çarpım, 0 toplama $\rightarrow N$ MAC işlem yapılır.

- Adım 5 : $(N/2)log_2N$ çarpım, $Nlog_2N$ toplama $\rightarrow Nlog_2N$ MAC işlem yapılır.
- Her bir segment için toplam MAC işlem sayısı : $2Nlog_2N + N$ yapılırken bütün segmentler için MAC işlem sayısı : $S(2Nlog_2N + N)$ şeklinde olmaktadır.
- Çizelge 4.2'deki FFT nokta sayısı N : 131072 ve segment sayısı S : 20 alındığında DDC işleminin toplam MAC işlem sayısı değeri : $9.18x10^7$ olmaktadır.
- OS-FFT tabanlı yöntem yerine FFT tabanlı filtreleme kullanıldığı takdirde toplam MAC işlem sayısı değeri : $9.02x10^7$ şeklinde olmaktadır.

Görüldüğü üzere OS-FFT tabanlı filtreleme ile yaklaşık %2'lik hesaplama karmaşıklığı (MAC işlem sayısı) artışı olmasına rağmen bellek kazancı 20x kat gibi büyük bir oranda artmaktadır. Verimli ve etkili bir paralel hesaplama mimarisi ile hesaplama karmaşıklığının neden olduğu çalışma ek yükü önemli bir ölçüde azaltılabilir. Paralel hesaplama yapılmadan önce DDC işleminin darboğaz işlem haritası çıkarılarak yerinde ve verimli optimizasyonlar ile paralel hesaplama tasarımları yapılabilmektedir.

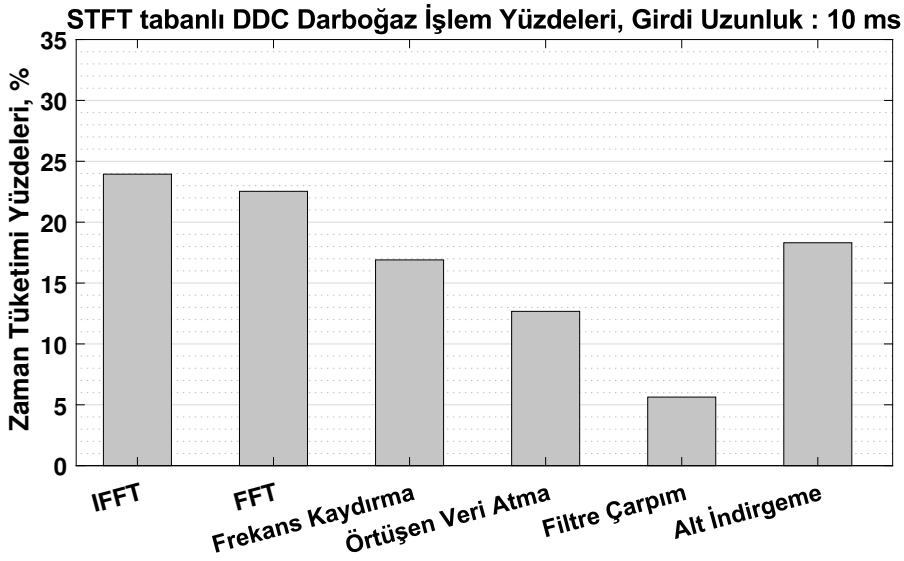
4.2.2. Darboğaz İşlem Haritası ve Değerlendirmeler

Darboğaz işlemler noktaları çıkarılırken STFT tabanlı DDC filtrelemenin CPU birimi üzerinde seri hesaplama tabanlı algoritma çalışma süresi ölçümleri alınmaktadır. Grafik işlemci ile hesaplama öncesinde CPU birimi üzerinde paralel hesaplamalar yapılarak problemin sadece CPU hesaplama mimarisi ile çözülmesi planlanmaktadır.
Hesaplamalar için optimizasyon yapılmadan önce algoritmanın hangi adımlarının yavaş çalıştığı ve paralel hesaplamaya ihtiyaç duyulduğu anlaşılmalıdır. Bu anlaşılırlığı pekiştirmek adına STFT tabanlı DDC filtreleme için Algoritma 1'deki adımlar ve Çizelge 4.2'deki parametre setine göre Şekil 4.4'te gösterilen darboğaz işlem haritası çıkarılmaktadır.
Darboğaz işlem değerleri algoritma çalışma süresine görece süreler oranları ile çıkarılmaktadır.



Şekil 4.4: 1 s Uzunluklu Girdi Verisi için Darboğaz İşlem Yüzdeleri

Şekil 4.4'teki gösterilen 1 s uzunluklu girdi verili senaryonun haricinde 10 ms'lik veri periyoduna sahip bir sayısal alt çevriminde de darboğaz işlem haritası Şekil 4.5'te sunulmaktadır.



Şekil 4.5: 10 ms Uzunluklu Girdi Verisi için Darboğaz İşlem Yüzdeleri

MAC işlem sayısı hesabına katılmayan ancak bellek transferi ve adres geçiş işlemleri içeren Algoritma 1'deki Adım-3, Adım-6 ve Adım-7 işlemleri darboğaz işlem analizinde dikkate alınmaktadır. Darboğaz analizinde yüzdeler her bir adımın çalışma süresinin toplam çalışma süresine oranıyla hesaplanmaktadır. STFT tabanlı DDC işleminde en fazla zaman

tüketimine neden olan FFT ve IFFT işlemleri için FFTW Kütüphanesi'nin fonksiyonları kullanılmaktadır [95, 96]. Şekil 4.4 ve Şekil 4.5'te görüldüğü üzere Adım-3(frekans kaydırma), Adım-6(örtüşen veri atma) ve Adım-7(alt indirgeme) işlemleri herhangi bir optimizasyon yapılmadığında toplamda ortalama %40 gibi önemli bir oranda zaman harcamaktadırlar.
Darboğaz işlem süre ölçümleri işlemci hızı 3.6 GHz olan Intel Xeon W-2123 CPU [97] birimiyle alınmıştır. Darboğaz işlem analizi, seri hesaplama ölçüm sonuçları üzerinde yapıldığından kırınımlı sonuçlar ve açıklamalar sırasıyla maddeler halinde belirtilmektedir:

1. Örnekleme hızı : 102.4 MHz ve dar bant örnekleme hızı 32 KHz olan bir senaryoda 3200 alt indirgeme oranı ile sayısal alt çevrim işlemi yapılmaktadır.

2. Segment uzunluğu 102400 (1 ms veri uzunluğu) varsayımıyla alt maddeler halinde farklı girdi veri uzunlukları için toplam sayısal alt çevrimi çalışma süre ölçüm sonuçları alınmıştır.

    - 1 s için : **5.885 s**
        - Frekans kaydırma : 1.178 s, DDC Filtreleme : 4.707 s
    - 10 ms için : **74.1 ms**
        - Frekans kaydırma : 12.8 ms, DDC Filtreleme : 61.3 ms
    - STFT tabanlı sayısal alt çevriminde DDC Filtreleme öbeği FFT, filtre çarpımı, IFFT, örtüşen veri atma ve alt indirgeme işlemlerini içermektedir.

3. STFT tabanlı DDC filtreleme yerine doğrusal evrişim tabanlı geleneksel CIC/FIR yeniden örnekleme ile DDC filtreleme yapıldığında alt maddelerde belirtildiği gibi ölçüm sonuçları alınmıştır.

    - 1 s için : **7.187 s** $\rightarrow$ (5.484 s frekans kaydırma, 1.703 s DDC filtreleme)
    - 10 ms için : **79.6 ms** $\rightarrow$ (17.2 ms frekans kaydırma, 62.4 ms DDC filtreleme)

CPU seri hesaplama sonuçlarına yönelik değerlendirmeler ve yol haritası öngörüleri maddeler halinde aşağı kısımda ortaya konmaktadır:

- ➤ Ölçüm sonuçlarına dair sonuçlar içeren 2. ve 3. maddelerdeki seri hesaplama ölçümleri karşılaştırıldığında STFT tabanlı sayısal alt çevrimi segment sayısı arttıkça geleneksel yönteme göre sunduğu başarım kazancının makası açılmaktadır. Segment sayısı artıkça önbellek yerelliğinin getirmiş olduğu kazanç artacağından bu sonuç beklendiği gibi çıkmaktadır.

- ➤ Karşılaştırma sonucunda göze çarpan en önemli iyileştirme frekans kaydırma işleminde gerçekleşmiştir. Geleneksel sayısal alt çevrim işleminde nümerik kontrollü osilator ile işlem yapıldığında cosinüs ve sinüs çarpımları ile Eşitlik 6'da anlatıldığı üzere frekans kaydırma yapılmaktadır. Seri ölçüm sonuçları doğrultusunda yapılan karşılaştırmada;

  - ✔ 1 s için **4.65x** hızlanma,
  - ✔ 10 ms için **1.35x** hızlanma gerçekleşmektedir.

- ➤ Darboğaz işlem analizi sonucunda bellek işlemleri içeren ama hesaplama karmaşıklık analizinde etkisi görülmeyen işlemler için hızlandırıcı optimizasyonlar yapılması gerekmektedir.

- ➤ FFT, IFFT ve filtre çarpımı için paralel hesaplamalar yapılarak veri işlem periyodundan daha kısa sürede işlemin yapılması sağlanmalıdır.

4.3. CPU Koşut İşleme

STFT tabanlı sayısal alt çevrimi işleminde filtreleme işlemi ne kadar hızlı yapılırsa dar bant sinyal üzerinde yapılacak tespit, geliş açısı kestirimi ve takip gibi diğer sinyal işleme bloklarının gerçek zamanlı olarak yapılması o kadar mümkün hale gelmektedir. Bölüm 4.2'de yapılan analizler sonucunda segmentli bir yapıda olan DDC filtreleme işleminde FFT ve IFFT için öncelikli olarak iyileştirmeler yapılmalıdır. Frekans kaydırma işlemi frekans uzayında veri transferi şeklinde yapıldığı için bir bellek transferi söz konusudur. Filtre çarpım işleminde ise çekirdek optimizasyonları yapılarak başarım artışı mümkündür. Çizelge 4.3'te algoritmanın her bir adımı için yapılan CPU paralel hesaplama yöntemleri belirtilmektedir.

Algoritma Adımları	**CPU Paralel Hesaplama Yöntemleri**
Adım-2 : FFT	SIMD (Oto-Vektörize), OpenMP (Oto-Paralelleştirme)
Adım-3 : Frekans Kaydırma	Hızlı Bellek Kopyalama (memcpy)
Adım-4 : Filtre Çarpımı	SIMD, Döngü Açma(Loop Unroll), OpenMP
Adım-5 : IFFT	SIMD, OpenMP
Adım-6 : Örtüşen Veri Atma	Hızlı Bellek Kopyalama (memcpy)
Adım-7 : Alt İndirgeme	Döngü Açma

Çizelge 4.3: STFT ile DDC İşlemi için CPU Paralel Hesaplama Teknikleri

Tek işlem çoklu veri (SIMD) prensibi ile aynı anda çoklu veri için işlem yapılmaktadır [98]. Intel işlemcilerinde SSE ve AVX mimarileri [99] ile geliştirilen SIMD yazmaç ve komut setleri, ARM işlemcilerinde NEON [100] mimarisi olarak karşımıza çıkmaktadır. Komut setleri çeşitli veri tipleri için destek sağlamaktadır. SIMD tabanlı bütün iyileştirmeler çekirdek optimizasyonu olarak ifade edilebilmektedir. Diğer bir çekirdek optimizasyonu ise döngü içeren bir problemde döngüsel gecikmeyi azaltan döngü açma yöntemidir. Bu yöntem ile iç içe işlem yapılmasına olanak sağlanarak toplam komut döngüsü düşürülmektedir [101].
Çekirdekler arası veri dağıtımı ile paralel hesaplama imkânı sunan OpenMP yöntemi ile çekirdekler arası optimizasyon sağlanmış olmaktadır [102]. Tez çalışmasında SIMD işlemleri için Intel C++ Compiler (icc) üzerinden oto-vektörize yapılırken, OpenMP işlemleri için de OpenMP hesaplama seçenekleri ve direktifleri kullanılmaktadır. OpenMP'nin verimli çalışabilmesi için problemde data bağımsız işlemlerin olması gerekir. Çizelge 4.3'te sunulduğu üzere FFT ve IFFT işlemleri için SIMD ve OpenMP optimizasyon içeren Intel FFT fonksiyonları kullanılmaktadır [103]. Bellek transferinin hızlandırılması için vektör kopyalamada kullanılan *memcpy* yöntemi kullanılmaktadır. Gerçekleştirilen CPU paralel hesaplama optimizasyonları ile 10 ms girdi veri uzunluklu senaryoda başarım ölçümleri sonucunda yapılan değerlendirmeler maddeler halinde belirtilmiştir:

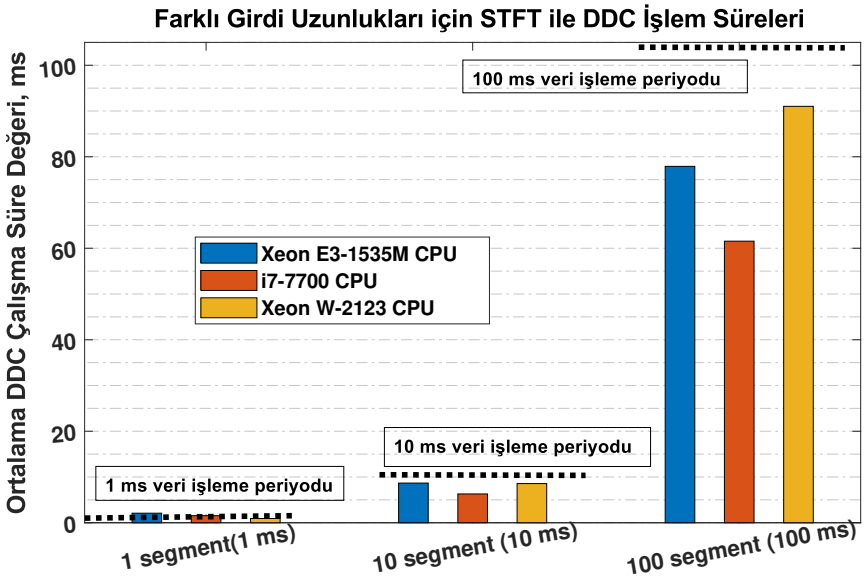
- ✔ Önerilen yöntemin CPU seri hesaplamasına göre frekans kaydırma işlemleri **21.3x** hızlandırılırken, DDC filtreleme öbeği işlemlerinde ise **7.66x** hızlanma sağlanmıştır.
- ✔ STFT tabanlı sayısal alt çevrimi işlemi toplamda CPU seri hesaplamaya göre **8.65x** hızlandırılırken işlem **8.561 ms** sürede tamamlanmıştır. Tek kanal DDC işlemi gerçek zamanlı olarak CPU'da uygulanabilir hale getirilmiştir.

STFT tabanlı sayısal alt çevrimi için çeşitli CPU birimleri ile paralel hesaplama ölçümleri alınarak başarım karşılaştırılması yapılmaktadır. Testler için kullanılan CPU hesaplama birimleri ve özellikleri Çizelge 4.4'te gösterilmektedir. Çizelge 4.4'te gösterilen güç tüketimi değerlerinde her bir işlemci için maksimum değer alınmıştır.

CPU Birimleri	Frekans (GHz)	Çekirdek Sayısı	Güç Tüketimi (Watt)
Intel Xeon W-2123	3.6	4	120
Intel Xeon E3-1535M	2.9	4	45
Intel i7-7700	3.6	4	65

Çizelge 4.4: CPU Paralel Hesaplama için Kullanılan Birimler ve Özellikleri

Çizelge 4.4'te belirtilen birimler ile yapılan başarım süre ölçümleri sonucunda Şekil 4.6'daki gibi sonuç ortaya çıkmaktadır. Şekil 4.6'da görüldüğü üzere farklı girdi uzunluklarına göre yapılan başarım testlerinde 10 ms'lik veri işleme periyodunda bütün birimler ile anlık veri işleme mümkün olmuştur. Ayrıca Intel i7-7700 CPU ile iki kanal DDC işlemi gerçek zamanlı olarak yapılabilmektedir. Güç verimliliği dikkate katıldığında Intel Xeon E3-1535M CPU birimi en yüksek güç/başarım kazancına sahiptir.



Şekil 4.6: Farklı Girdi Uzunlukları ile Çeşitli CPU Birimlerinde DDC Koşut Hesaplama Başarım Karşılaştırmaları

Alınan test sonuçları doğrultusunda CPU birimlerinde paralel hesaplama mimarisi ile seri hesaplamaya göre başarım artışı sağlanmış olup tek kanal geniş bant DDC işlemi gerçek zamanlı yapılabilecek hale getirilmiştir. Seri hesaplama ile yapılan karşılaştırmada 8.65x kat hızlanma özellikle SIMD ve OpenMP optimizasyonları ile sağlanmaktadır. Testlerde görüldüğü üzere 1 ms'lik veri uzunluğunda her birimde gerçek zamanlılık sağlananamaktadır. Intel i7-7700 CPU ile alınan yüksek başarım işlemcinin turbo frekans hızının diğer birimlere göre daha yüksek olması ile açıklanabilmektedir. Çok kanallı DDC uygulaması için GPU birimlerinin yer aldığı paralel hesaplama mimarileri çalışılmaktadır.

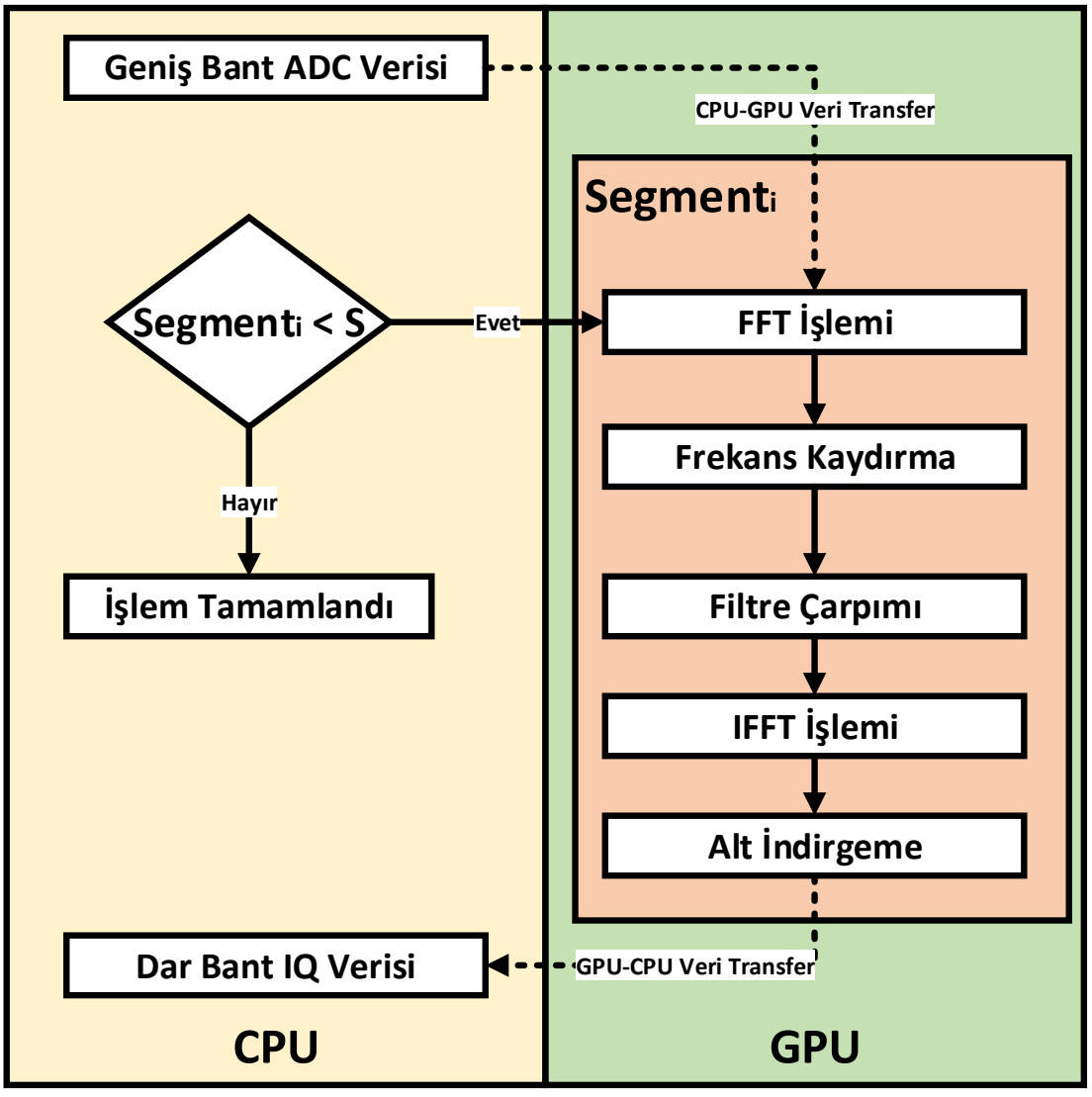
4.4. GPU ile Hızlandırma

Grafik işlemci birimleri sahip oldukları çok sayıda iş parçacığı (thread) sayesinde paralel hesaplama, derin öğrenme ve gerçek zamanlı uygulama teknolojisinde yeniliklere çığır açan bir donanım olarak kabul edilmektedir [10]. STFT tabanlı sayısal alt çevrimi işleminde GPU birimleri ile özellikle FFT ve IFFT işlemlerinde kazanılacak başarım artışı ile işlem süresinin önemli bir ölçüde azalacağı varsayılmaktadır. Bu varsayımı destekleyen bir çalışmada FFT işlemlerinin özellikle çok sayıda *thread* barındıran GPU'da büyük FFT uzunlukları için yüksek başarım sunduğunu belirtilmektedir [104].

GPU birimleri içerdikleri blok ve thread organizasyonu sayesinde aynı komutla çoklu thread (SIMT) mimarisi çalıştırabilmektedirler. GPU biriminde üç boyutlu bir mimari tasarım ile grid, blok ve thread yapısıyla paralel hesaplama yapılması mümkündür. Fakat tez çalışmasında sadece blok ve thread organizasyonuna yoğunlaşılmaktadır. GPU paralel hesaplama tasarımında veri boyutu, thread ve blok tasarımını önemli ölçüde etkilemektedir. Çizelge 4.5'te hem veri boyutu ile ilgili bilgiler hem de blok-thread organizasyonundan bahsedilmektedir. Heterojen mimari tasarımında CPU ve GPU birimleri paralel tasarım ve maksimum fayda için en uygun şekilde kullanılmaktadır. STFT tabanlı sayısal alt çevriminde her bir segment için işlemler kendi içinde veri bağımsız olduğu için GPU paralel programlamaya uygun olmaktadır. Şekil 4.7'de önerilen CPU-GPU heterojen mimari tasarımı sunulmaktadır.

Tasarım Özellikleri	Değerler
Girdi Örnekleme Hızı	102.4 MHz
Girdi Data Uzunluğu	10 ms
Segment Uzunluğu	1024000
Segment Sayısı	10
FFT Uzunluğu	131072 = 128 x 1024
Thread Sayısı	1024
Blok Sayısı	128

Çizelge 4.5: GPU Paralel Hesaplama için Veri Özellikleri ve Blok-Thread Organizasyonu



Şekil 4.7: STFT tabanlı Sayısal Alt Çevrimi için CPU-GPU Heterojen Mimari

Şekil 4.7'de görüldüğü üzere segment sayısı (S) kadar CPU'dan kernel fonksiyonlar çağrılarak GPU işlemleri yapılır. FFT ve IFFT işlemleri için NVIDIA CUDA cuFFT Kütüphanesi fonksiyonları kullanılmaktadır [105]. Frekans kaydırma, filtre çarpımı ve alt indirgeme işlemleri için SIMT mimarisi ile aynı anda çoklu thread kullanılarak hesaplamalar gerçekleştirilmektedir. Tasarlanan CPU-GPU heterojen mimarisi doğrultusunda Çizelge 4.6'da bahsedilen çeşitli GPU birimleri ile alınan başarım ölçümleri Çizelge 4.7'de gösterilmektedir.

GPU Birimleri	**Çekirdek Sayısı**	**Güç Tüketimi (Watt)**
NVIDIA Quadro M4000M	*1280*	*100*
NVIDIA GTX 1050	*640*	*75*
NVIDIA GTX 1080	*2560*	*180*

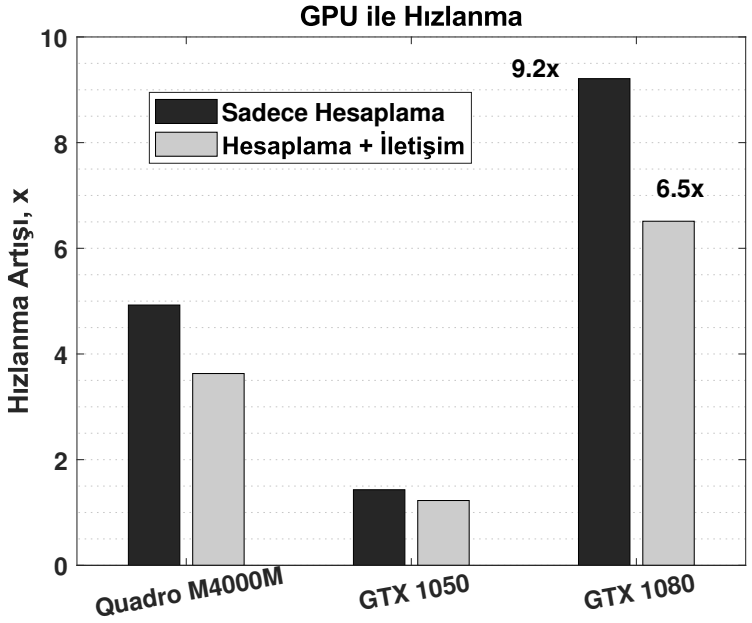
Çizelge 4.6: GPU Paralel Hesaplama için Kullanılan Birimler ve Özellikleri

CPU ve GPU arasındaki veri transferleri işlem hızına negatif etki oluşturmaktadır. Başarım ölçümlerinde CPU-GPU veri iletişimi de ayrıca dikkate alınarak karşılaştırma yapılmaktadır.

Girdi Uzunluk	**1 ms**			**10 ms**			**100 ms**		
Ölçümler (ms)	**H**	**İ**	**T**	**H**	**İ**	**T**	**H**	**İ**	**T**
Quadro M4000	*0.155*	*0.084*	*0.239*	*1.277*	*0.456*	*1.733*	*13.806*	*3.968*	*17.874*
GTX 1050	*0.716*	*0.095*	*0.811*	*4.397*	*0.731*	*5.128*	*20.797*	*5.217*	*26.014*
GTX 1080	*0.115*	*0.111*	*0.226*	*0.683*	*0.283*	*0.966*	*6.604*	*1.821*	*8.425*

Çizelge 4.7: STFT tabanlı DDC İşlemi için Farklı Girdi Uzunlukları için Başarım Ölçümleri Hesaplama (H), İletişim (İ), Toplam (T)

Çizelge 4.7'de gösterildiği üzere GPU birimleri ile Şekil 4.6'da gösterilen CPU performanslarına göre oldukça daha yüksek başarım elde edilmektedir. Hesaplama başarımı CPU-GPU arasındaki veri iletişimi nedeniyle azalsa bile çok kanallı DDC işlemine uygun çözümler sunulmaktadır. GPU ile kazanılan hızlanma artışlarının analizi yapabilmek için Çizelge 4.6'daki her bir GPU için en yüksek başarım sunan CPU koşut işleme performansına göre hızlanma sonuçları Şekil 4.8'de gösterilmektedir. Sadece hesaplama hızlanmasının haricinde CPU-GPU iletişiminin yer aldığı sonuçlara göre hızlanmalar da belirtilmektedir.



Şekil 4.8: 10 ms Girdi Uzunluğu için Intel i7-7700 CPU Koşut İşleme Başarımına göre GPU Hızlanma İyileştirmeleri

Şekil 4.8'e göre GTX 1080 ile 9.2x hesaplama hızlanma artışı alınırken iletişim ek yükü dahil edildiğinde sonuç olarak CPU koşut işlemeye göre 6.5x hızlanma artışı kaydedilmektedir. CPU-GPU veri iletişimi ortalamada %27 gibi bir kayba neden olmaktadır. Başarım karşılaştırmasının yanı sıra GPU kullanımı ile birlikte çok kanallı DDC işlemi yapılması hedeflenmektedir. Bölüm 4.3'te yapılan analizlere göre FFT işlemi yapıldıktan sonra istenilen her bir DDC frekansı için Algoritma 1'de Adım-3'ten Adım-7'ye kadar işlemler tekrarlanır. Bunun yanı sıra GPU'daki kernel fonksiyonların sürelerinin analizi yapıldığında Çizelge 4.8'deki gibi sonuçlar ortaya çıkmaktadır. Çizelge 4.7'deki 10 ms için süre ölçümleri yeniden düzenlendiğinde sırasıyla Quadro M4000M ile **11** kanal, GTX 1050 ile **2** kanal ve GTX 1080 ile **21** kanal DDC işlemi 10 ms'lik veri işleme periyodu içerisinde yapılabilmektedir.

Kernel Fonksiyonlar	**İşlem Süre Yüzdeleri**
FFT İşlemi	%35.9
Frekans Kaydırma ve Filtreleme	%24.2
IFFT İşlemi	%36.4
Alt Örnekleme (Örtüşen Veri Atımı İçerir)	%3.5

Çizelge 4.8: 10 ms Girdi Uzunluğu için Alt İşlemlerine ait Çalışma Süre Yüzdeleri

GPU ile hızlandırma esnasında kernel çağırma ek yükünü azaltmak için frekans kaydırma ile filtre çarpım işlemleri örtüştürülerek aynı kernel içinde gerçekleştirilmektedir. Bu sayede heterojen mimari tasarımının çalışma hızına olumlu katkı sunulmaktadır.

4.5. Güç Verimli GPU Hesaplama

GPU ile hızlandırmada güç tüketimi gibi daha fazla kaynak kullanımı olduğu için tasarım verimliliği tartışılmaktadır. Uç hesaplama algoritmaları için özelleşmiş GPU birimleri ile düşük güç tüketimli uygulamalar tasarlamak mümkündür. Jetson GPU mimarileri ile başarım testleri yapılarak sonuçlar analiz edilmektedir. Çizelge 4.9'da sayısal alt çevrimi için başarım testlerinde kullandığımız GPGPU mimarilerine ait özellikler belirtilmektedir. Güç tüketim değeri 'sudo /usr/sbin/nvpmodel' komutuyla veya arayüz yazılımı üzerinden belirlenmiştir.

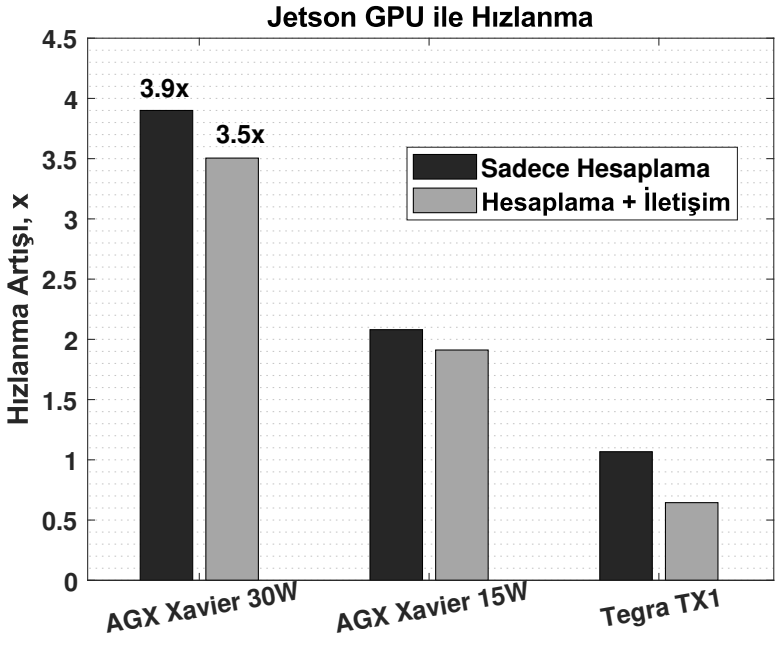
Jetson GPU Birimleri	**Çekirdek Sayısı**	**Güç Tüketimi (Watt)**
AGX Xavier GPU	512	15 ve 30
Tegra TX1	256	6.5

Çizelge 4.9: Güç Verimli Hesaplama Başarım Testlerinde Kullanılan NVIDIA Jetson GPU Birimleri ve Özellikleri

Çizelge 4.10'da yer alan sonuçlara göre AGX Xavier GPU'nun her iki güç modunda ve Tegra TX1 GPU ile 10 ms'lik veri işleme periyodunda en az 1 kanal DDC işlemi gerçek zamanlı olarak yapılabilmektedir. Dikkati çeken en önemli gelişme AGX Xavier GPU ile veri transfer süreleri önemli ölçüde azalmaktadır. Jetson mimarilerinde bellek transferi verimli bir şekilde yapılabilmektedir. 10 ms'lik girdi veri uzunluğu için en yüksek başarımlı CPU koşut işleme performansına göre karşılaştırmalar Şekil 4.9'da gösterilmektedir.

Girdi Uzunluk	**1 ms**			**10 ms**			**100 ms**		
Ölçümler (ms)	**H**	**İ**	**T**	**H**	**İ**	**T**	**H**	**İ**	**T**
AGX Xavier 30 W	*0.208*	*0.098*	*0.316*	*1.613*	*0.182*	*1.795*	*16.585*	*0.795*	*17.380*
AGX Xavier 15 W	*0.318*	*0.185*	*0.503*	*3.024*	*0.267*	*3.291*	*30.598*	*1.294*	*31.892*
Tegra TX1	*0.986*	*0.574*	*1.560*	*5.894*	*3.867*	*9.761*	*56.982*	*20.752*	*77.734*

Çizelge 4.10: STFT tabanlı Sayısal Alt Çevrimi için Jetson GPU Hesaplama Başarım Testleri

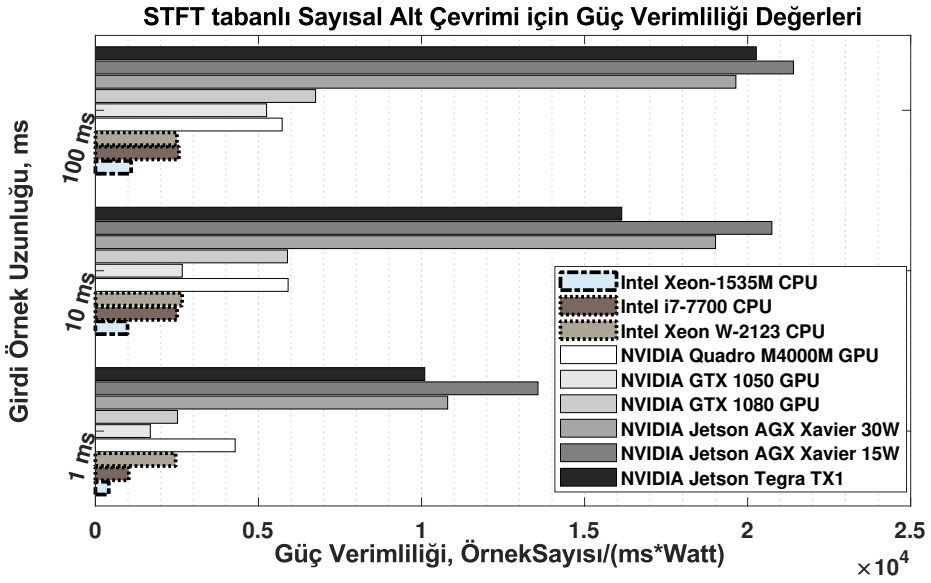


Şekil 4.9: 10 ms Girdi Uzunluğu için Intel i7-7700 CPU Koşut İşleme Başarımına göre Güç Verimli Jetson GPU Hızlanma İyileştirmeleri

Şekil 4.9'da gözlemlenen sonuçlara göre Jetson AGX Xavier GPU 30W modunda yaklaşık 3.9x hızlanma sağlanmaktadır. Veri transferi işlemi dahildiğinde sonuç olarak 3.5x hızlanma sağlanmıştır. Bu hızlanma sonucu sayesinde Çizelge 4.7'de yer alan GPU başarımlarının (GTX 1080 hariç) hepsinden daha üstün başarım (veri iletişimi dahil) Jetson AGX Xavier GPU ile 30W güç tüketimi ile alınmaktadır. Şekil 4.8'daki karşılaştırmadan çıkarılan bir sonuç ise Tegra TX1 GPU ile sadece hesaplama durumunda düşük olsa bile hızlanma sağlanırken veri iletişimi dahil edildiğinde en iyi CPU koşut işlemeye göre başarımının geride kalmasıdır. Bölüm 4.4'te yapılan çok kanallı DDC işlem analizini aynı yaklaşımla Jetson GPU'lar için aldığımızda 10 ms girdi uzunluğu için karşımıza çıkan sonuç:

- AGX Xavier GPU 30 W güç tüketimi ile **8** kanal,
- AGX Xavier GPU 15 W güç tüketimi ile **4** kanal,
- Tegra TX1 ile **1** kanal DDC işleminin gerçek zamanlı yapılabilmesidir.

AGX Xavier GPU birimi Quadro M4000M GPU birimine göre daha yüksek hızlanma sağlamaktadır. Quadro M4000M GPU'da veri iletişim süresi alınan ölçümlere göre AGX Xavier GPU 30W modundaki veri iletişimden daha fazla olduğu görülmektedir. Jetson GPU'lar entegre bir hibrit mimari içerdiği için veri transferi daha verimli olabilmektedir. Jetson GPU'lar ile alınan ölçümler sonrası verimlilik analizleri yapılarak özellikle güç tüketiminin limitli olduğu senaryolarda avantaj sağlayabilecek tasarımının yapılması güç ve maliyet etkin bir çözüm ortaya koyabilmektedir.



Şekil 4.10: STFT tabanlı Sayısal Alt Çevrimi için Güç Verimliliği Karşılaştırmaları

Şekil 4.10'da STFT tabanlı sayısal alt çevriminde hem CPU hem de GPU başarım testleri için kullanılan birimlerin güç verimlilik sonuçları gösterilmektedir. Verimlilik birimi örnek sayısı/(hesaplama süresi*güç tüketimi) oranını verecek şekilde örnekSayısı/(ms*Watt) olarak belirlenmektedir. Veri uzunluğu arttıkça DDC işleminde CPU birimlerinin güç verimi GPU birimlerine kıyasla oldukça düşük çıkmaktadır. NVIDIA Jetson AGX Xavier GPU biriminin 15W güç tüketimi modunda güç verimliliği bütün birimlere göre daha fazladır. Düşük güç tüketimine ihtiyaç duyan sensör mimarilerinde NVIDIA Jetson AGX GPU birimleri SWaP (size, weight and power) olarak bilinen bir ürünün boyut, ağırlık ve güç tüketimi faktörlerini DDC uygulaması özelinde iyileştirmektedir.

4.6. Tartışma

Sayısal alt çevrimi işleminde STFT tabanlı bir yaklaşım ele alınarak paralel hesaplamaya uygun algoritma tasarımı gerçekleştirilmiştir. STFT tabanlı sayısal alt çevriminde frekans kaydırma işlemi zaman uzayı yerine frekans uzayında yapılarak algoritmik hesaplama kazancı sağlanmıştır. Frekans kaydırma işleminde FFT sonrası tam sayı katında bir kaydırma yapıldığı için yuvarlama ve işlem çarpanı hata oranı ile karşılaşılmaktadır. Genlik doğruluk testlerinde frekans uzayında yapılan frekans kaydırma işlemi zaman uzayındaki işleme göre maksimum %2 mutlak hata oranı ile gerçekleşmektedir. Filtreleme işleminin frekans uzayında yapılması her bir DDC kanalı için yapılacak frekans uzayına dönüşüm işlemini engellemektedir. Aynı geniş bant sinyal üzerinde FFT işlemi yapıldıktan sonra ilgili DDC frekansı için kaydırma yapılarak kalan işlemlerin tekrarlanması yeterli olmaktadır.
STFT tabanlı sayısal alt çevrimi için darboğaz işlem adımları belirlenmiş olup hesaplama analizleri yapılmıştır. FFT-IFFT tabanlı bir çözüme dönüşmesi sayesinde özellikle paralel hesaplama mimarileri ile işlemleri hızlandırılmıştır. CPU paralel hesaplama ile en az 1 kanal DDC işlemi veri işleme periyodu içerisinde tamamlanmaktadır. GPU kullanımı ile işlem süreleri azaltılarak maksimum 21 kanal DDC işlemi gerçek zamanlı olarak yapılabilmesi mümkün hale getirilmiştir. STFT tabanlı yaklaşım için düşük güç tüketimi sunan Jetson GPU birimleri ile başarım testleri gerçekleştirilmiştir. Testler sonucunda maksimum 8 kanal DDC anlık olarak işlenebilmektedir. Şekil 4.10'da güç verimlilik karşılaştırmasında Jetson GPU'lar ile verimli hesaplamalar yapıldığı gösterilmiştir. 10 ms'lik girdi uzunluğu için maksimum DDC kanal sayısı güç verimlilik ilişkisi birlikte değerlendirildiğinde Jetson AGX Xavier 30W modu GTX 1080 GPU'ya göre daha yüksek hesaplanan kanal sayısı güç tüketimi oranına sahiptir. Jetson AGX Xavier 30W modunda $1.9x10^4$ örnek sayısı / (ms * Watt) ile 8 kanal DDC ve GTX 1080 ile $6x10^3$ örnek sayısı / (ms * Watt) ile 21 kanal DDC yapılırken, GTX 1080 için güç tüketimi 30W olarak ele alındığında ancak 6 kanal DDC yapılabilmektedir.

Jetson GPGPU platformları ile alınan ölçümler sonrası değerlendirme yapıldığında uç hesaplama ve insansız mobil sensör mimarilerinde Jetson AGX Xavier ile çok kanallı DDC

işleminin gerçek zamanlı yapılabileceği öngörülmektedir. Literatürdeki FPGA, CPU ve GPU birimleri üzerinde yapılan FFT tabanlı sayısal alt çevrimi işlemlerin olduğu üçüncü bölümde gösterilmektedir. Fakat Jetson mimarileri ile yapılan sayısal alt çevrimi işlemlerine literatür incelemelerinde karşılaşılmamış olup GPU tabanlı DFT/FFT tabanlı kanallaştırma çalışmalarına rastlanılmıştır [106]. Tez çalışması bu bakış açısı ile değerlendirildiğinde literatüre mobil sistemler için çok kanallı gerçek zamanlı çalışabilecek düşük güç tüketim ile sayısal alt çevrimi uygulamasını ilk defa sunmaktadır.

Önerilen sayısal alt çevrimi çalışmalarının daha da iyileştirilebilecek kısımlarının olduğu öngürülmektedir. Bu kısımlardan bir tanesi, STFT tabanlı sayısal alt çevrimi işleminde alt örnekleme işlemini IFFT almadan önce yapmaktır. Bu kısımda yapılabilecek iyileştirme, alt örnekleme işlemini zaman uzayında yapmak yerine frekans uzayında yaparak IFFT işleminin girdi uzunluğu indirgeme oranı kadar azaltmaktır [107]. IFFT işleminin çalışma süresi bu sayede azalacağından maksimum DDC kanal sayısı artacaktır. Bu iyileştirme özellikle alt kanal iletişimleri barındıran yeni nesil geniş bant haberleşme uygulamalarına fayda getirecektir. Diğer yapılabilecek bir iyileştirme ise veri transferini asenkron yaparak iletişim ile hesaplama işlemlerinin örtüştürülmesidir. Bu işlem sayesinde iletişim süresinin toplam hesaplama süresine olan etkisi önemli ölçüde azalacak olup GPU birimlerinin başarımı artabilecektir.

5. MAKİNE ÖĞRENMESİ TABANLI HEDEF TESPİTİ

Bu bölümde, sensör mimarisi için geliştirilen makine öğrenmesi tabanlı hedef tespitine yönelik gerçek zamanlı uygulamaların başarılabilmesi için gereken paralel hesaplamalar anlatılmaktadır. Sayısal alt çevrimi sonrası belirli bir bant genişliğine indirilen sinyal, ön işleme bloklarından geçtikten sonra zaman ve frekans uzayında örnekler içeren bir matrise dönüştürülür. Bu matris üzerinde hedef tespiti yapılarak hareketli hedeflerin frekansı ve zaman gecikmesi ölçülür. Bir radar sensöründe ölçülen bu değerler hareketli hedefin Doppler hızı ve mesafesi olmaktadır.

Tez çalışmasında hedef tespiti için algoritma modeli detaylı inceledikten sonra CPU seri hesaplama analizi yapılmaktadır. CPU koşut işleme analizinden sonra başarım hızlandırmaları incelenmektedir. Grafik işlemci ve güç verimli hesaplama analizleri ile düşük güç tüketen mobil sensör mimarilerine uygun paralel hesaplamalar tasarlanmaktadır.

5.1. Algoritma Modeli

Makine öğrenmesi tabanlı hedef tespiti uygulamasına girdi olan matris Bölüm 2.2.1.'de belirtildiği üzere menzil ve darbe vektörlerinden oluşturulmaktadır. Geleneksel yöntem olan SYAO tabanlı tespit yöntemine göre algoritmik başarım artışı sunan makine öğrenmesi tabanlı yöntem aynı matrisi girdi olarak almaktadır [12]. Makine öğrenmesi tabanlı yöntemde hedef tespit işlemi için CNN modeli kullanılmakta olup akış şeması Şekil 2.8'de gösterilmektedir.

CNN modeli herhangi bir tespit konumu etiketi ile eğitilmediği için işlem sonucunda bulunan hedef sayısı modelin hesaplama karmaşıklığını etkilememektedir. Matris girdi boyutundan bağımsız olarak model tespit olasılığı tabanlı bir yaklaşım 2B pencere ile tüm matrisi tarayarak hedef tespitine karar vermektedir [1]. Tezde incelenen makine öğrenmesi tabanlı hedef tespiti algoritması çalışmasında [1] yoğunluk esaslı sinyal genlikleri içeren imge üzerinde 2B pencere ile taramalar yapılarak bölgesel tespit karar verilmesi sağlanmaktadır. Detaylı akış haricinde algoritma adımları Algoritma 2'de anlatılmaktadır.

Algoritma 2 Makine Öğrenmesi tabanlı Hedef Tespiti için Algoritma Adımları [1]

Girdiler

M : Menzil Hücre Sayısı, N : Darbe Sayısı, C : Evrişim Kanal Sayısı,
Thr : Tespit Eşiği, S : Aktivasyon Kanal Sayısı, K : Tespit Sayısı,
X : 2B Pencere Menzil Hücre Sayısı, $X << M$
Y : 2B Pencere Darbe Sayısı, $Y << N$
P : 2B Evrişim Ağırlık Matrisi Menzil Hücre Sayısı,
Q : 2B Evrişim Ağırlık Matrisi Darbe Sayısı,
A : Menzil-Darbe Girdi Matrisi, MxN,
B : CNN Modeli 2B Pencere Matrisi, XxY,
W : Çok Kanallı 2B Evrişim Ağırlık Matrisi, $PxQxC$,
F : Tam Bağlantı İşlem Katmanı Ağırlık Matrisi, $Sx(C(X-P+1)(Y-Q+1))$,
Çıktı : Tespit Raporu (No, Menzil, Hız), $Kx3$.

Algoritma

Tarama Menzil Sayısı $T_M = (M - X + 1)$.
Tarama Darbe Sayısı $T_D = (N - Y + 1)$.
for i = 1:T_M **do**
  for j = 1:T_D **do**
    Adım-1 : (i, j)'in 2B CNN Girdi Pencere Matrisi Oluşturma, Çıktı : XxY.
    Adım-2 : Min-Max Ölçekleme Katmanı, Çıktı : XxY.
    Adım-3 : Çok Kanallı 2B Evrişim Katmanı, Çıktı : CxH_xxH_y.

      $\mathbf{H} = B_{i,j} \times \mathbf{W}+$ Ofset.

      H matrisi menzil hücre sayısı : $H_x : X - P + 1$.

      H matrisi darbe sayısı : $H_y : Y - Q + 1$.

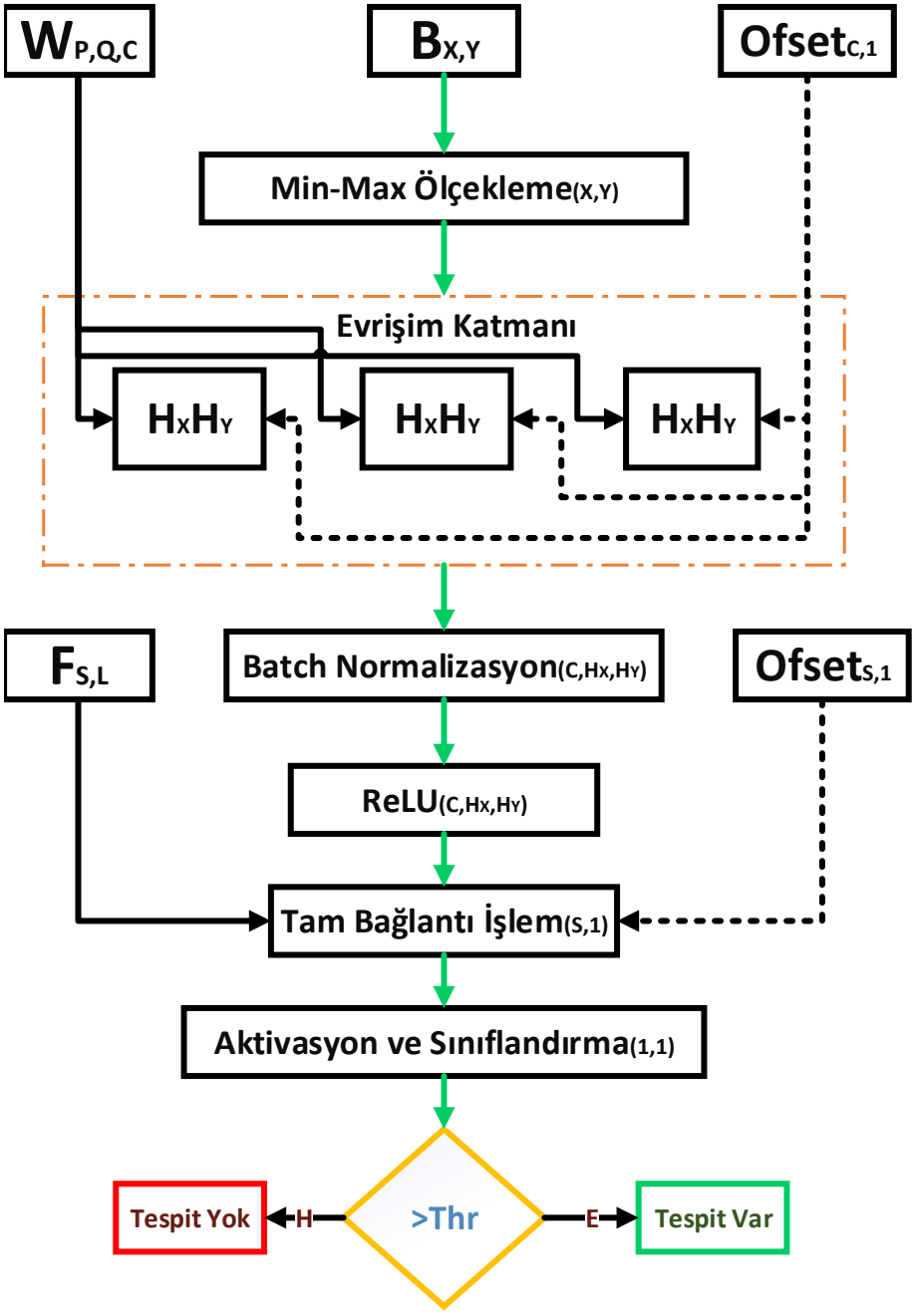
    Adım-4 : Batch Normalizasyon Katmanı, Çıktı : CxH_xxH_y.
    Adım-5 : ReLU Katmanı, Çıktı : CxH_xxH_y.
    Adım-6 : Tam Bağlantılı İşlem Katmanı, Çıktı : $Sx1$.

      $\mathbf{Z} = \mathbf{F} \times \mathbf{H}(:)+$ Ofset.

      L : CH_xH_y.

    Adım-7 : Aktivasyon ve Sınıflandırma Katmanı, O : Çıktı : $1x1$.
    Adım-8 : Eşik İşlemi.
    if $O > Thr$ **then**
      Tespit Raporu$(K) = (K, i, j)$.
      $K = K + 1$.
    end if
    Adım-9 : Ardışık darbe penceresine geç, $j = j + 1$.
  end for
  Adım 10 : Ardışık menzil penceresine geç, $i = i + 1$.
end for

Algoritma 2'de bahsedildiği üzere Menzil-Darbe girdi matrisi, A_{MxN}, üzerinde CNN modeli 2B pencere matrisi, B_{XxY}, menzil ve darbe ekseninde taratılarak CNN modeli her tarama adımında hesaplanır. Adım-8'den sonra o tarama adımı için tespit kararı verilmektedir. Şekil 5.1'de her bir tarama adımında yapılan Algoritma 2 içindeki işlemlerin veri akışı ve bileşenler gösterilmektedir.



Şekil 5.1: Makine Öğrenmesi tabanlı Hedef Tespiti için CNN Modeli Veri-İşlem Akışı [1]

5.2. Hesaplama Analizi

Algoritma 2'deki adımları içeren uygulamada bellek kaynak kullanımı ve hesaplama analizleri yapılarak algoritmanın karmaşıklığı incelenmektedir.

5.2.1. Bellek Kaynak Kullanımı Analizi

Şekil 5.1 ve Algoritma 2'de görüldüğü üzere CNN modelinde sensöre ait menzil-darbe girdi matrisine göre küçük boyuttaki matris ile modelin girdisi sağlanmaktadır. Bu yöntem bellek gereksinimini azaltan bir yaklaşımdır. Bellek gereksinimini analiz etmek için kaynak kullanımı her bir adım için Çizelge 5.1'de Algoritma 2'deki parametrelere göre belirtilmektedir.

Algoritma Adımları	**Bellek Kaynak Kullanımı**
Adım-1	$MN + XY$
Adım-2	XY
Adım-3	$C(X - P + 1)(Y - Q + 1) + CPQ + C$
Adım-4	$C(X - P + 1)(Y - Q + 1)$
Adım-5	$C(X - P + 1)(Y - Q + 1)$
Adım-6	$SL + L + S$
Adım-7	1

Çizelge 5.1: Makine Öğrenmesi tabanlı Hedef Tespit için Bellek Kaynak Kullanımı

Çizelge 5.1'e göre bellek kullanımı, $L = C(X - P + 1)(Y - Q + 1)$ eşitliği ile, toplam $MN + 2XY + 4L + CPQ + SL + C + S$ olmaktadır. Referans alınan çalışmaya [1] göre parametre seti belirlendiğinde Çizelge 5.2'deki gibi parametreler kullanılmaktadır. Bu parametre değerlerine göre veri tipi için 32-bit kayar nokta kullanımında yaklaşık 193 KB bellek alanı yeterli olmaktadır. Bellek kaynak kullanımı için tespit çıktısı dikkate alınmadan bir sonuç hesaplanmaktadır. Bellek kaynağı az kullanılmasına rağmen taramalı bir algoritma yaklaşımından dolayı $T_M T_N$ kadar modelin çağrılması gerekir.

Parametreler	M	N	X	Y	P	Q	C	S
Değerler	3000	16	25	5	5	3	3	2

Çizelge 5.2: CNN Modeli için Parametre Seti

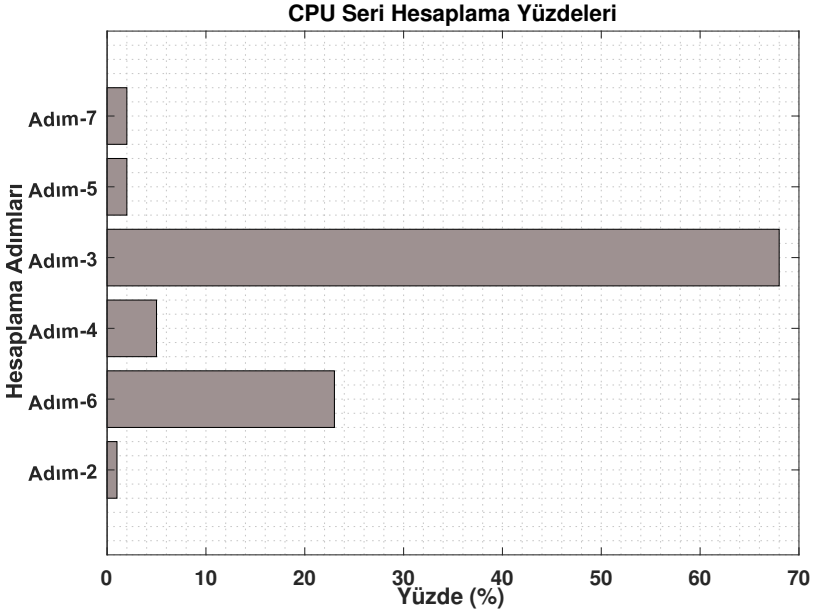
5.2.2. Hesaplama Karmaşıklığı ve Darboğaz İşlemler

Algoritma 2'deki adımlar için Çizelge 5.3'te MAC işlem sayısı değerleri verilerek hesaplama karmaşıklığı sunulmakta olup toplam işlem sayısı $1.3x10^8$ civarındadır. Çizelge 5.2'deki parametrelere göre $T_M = 2976$ ve $T_N = 12$ olmaktadır. CNN modeli 35172 kez aynı menzil-darbe matrisi için çalıştırılarak eşik işlemi gerçekleştirilir.

Algoritma Adımları	**MAC İşlem Sayısı**	**Değerler**
Adım-2	XY	125
Adım-3	$C(X-P+1)(Y-Q+1)PQ$	2835
Adım-4	L	189
Adım-5	L	189
Adım-6	SL	378
Adım-7	S	2
Toplam	$T_M T_N(XY + L(PQ+2+S)+S)$	$13.277x10^7$

Çizelge 5.3: Makine Öğrenmesi tabanlı Hedef Tespiti için Hesaplama Karmaşıklık Analizi

Hesaplama yükü fazla olan Adım-3 (Evrişim Katmanı) ve Adım-6 (Tam Bağlantı İşlem Katmanı) işlemleri en darboğaz işlemler olması beklenmektedir. Şekil 5.2'de CPU seri hesaplama süresi oranları gösterilmektedir.



Şekil 5.2: Makine Öğrenmesi tabanlı Hedef Tespiti - Darboğaz İşlemler

Şekil 5.2'deki sonuçlar Çizelge 5.3 ile örtüşmektedir. Ölçümler ve analizler doğrultusunda Adım-3'teki çok kanallı 2B Evrişim işlemi ile Adım-6'daki matris vektör çarpımı işlemleri darboğaz hesaplamalar olarak belirlenmektedir. Taramalı bir şekilde model hesaplaması, karmaşıklığı artırmaktadır. Intel Xeon W-2123 CPU'da alınan ölçümlere göre Çizelge 5.2'deki parametreler kullanımında yaklaşık **82** s sürmektedir. Bu sonuç gerçek zamanlı hedef tespiti hedefi ile uyuşmamaktadır. Hesaplama tasarımı ve donanım-yazılım iyileştirmeleri ile başarım artışı hedeflenmektedir.

5.3. Paralel Hesaplama Tasarımı

Gözlemlenen darboğaz işlemlere yönelik iyileştirmeler yapılarak hedef tespiti işlemi yüksek başarımla hesaplanabilmektedir. İyileştirmeleri iki aşamada ele alarak etkin çözüm ile bir tasarım planlanmaktadır. Bu iki aşama alt bölümler ile ayrıntılı olarak incelenmiştir.

5.3.1. Paralel Hesaplama tabanlı Tespit İşlemi

Hesaplama karmaşıklığı ve bellek kaynak kullanımı analizinde görüldüğü üzere, taramalı yöntem ile girdi matrisine görece küçük bir matris ile model hesaplaması yapılarak hedef tespiti yapılmaktadır. Bu yaklaşım bellek kullanımını, uzay karmaşıklığını, azaltırken hesaplama karmaşıklığını arttırmaktadır. Tez çalışmasında, bu durumla mücadele etmek için uzay karmaşıklığı arttırılarak hesaplama karmaşıklığının azaltılması incelenmektedir.
Algoritma 2'deki adımlar tarama sebebiyle tekrarlı veri hücreleri içermektedir. Her bir taramada kaydırma nedeniyle menzil ve darbe eksenindeki veriler, önceki taramada yer alan verilerin çoğunluğuyla örtüşecek şekilde eşik hesaplamak için tekrar kullanılmaktadır. Bölgesel model hesaplaması yapılması tespit doğruluğunu artıran bir durumdur [1]. Fakat taramalı yöntem yerine işlemlerin MxN boyuttaki girdi matrisi üzerinden yapılması hesaplama yükünü azaltabilmektedir. Adım-2'deki matris ölçekleme, Adım-4'teki normalizasyon ve Adım-5'teki negatif kırpma işlemleri bağımsız eleman bazlı işlemler olduğu için tüm girdi matrisi üzerinden yapılmasında doğruluk kaybı oluşmamaktadır.

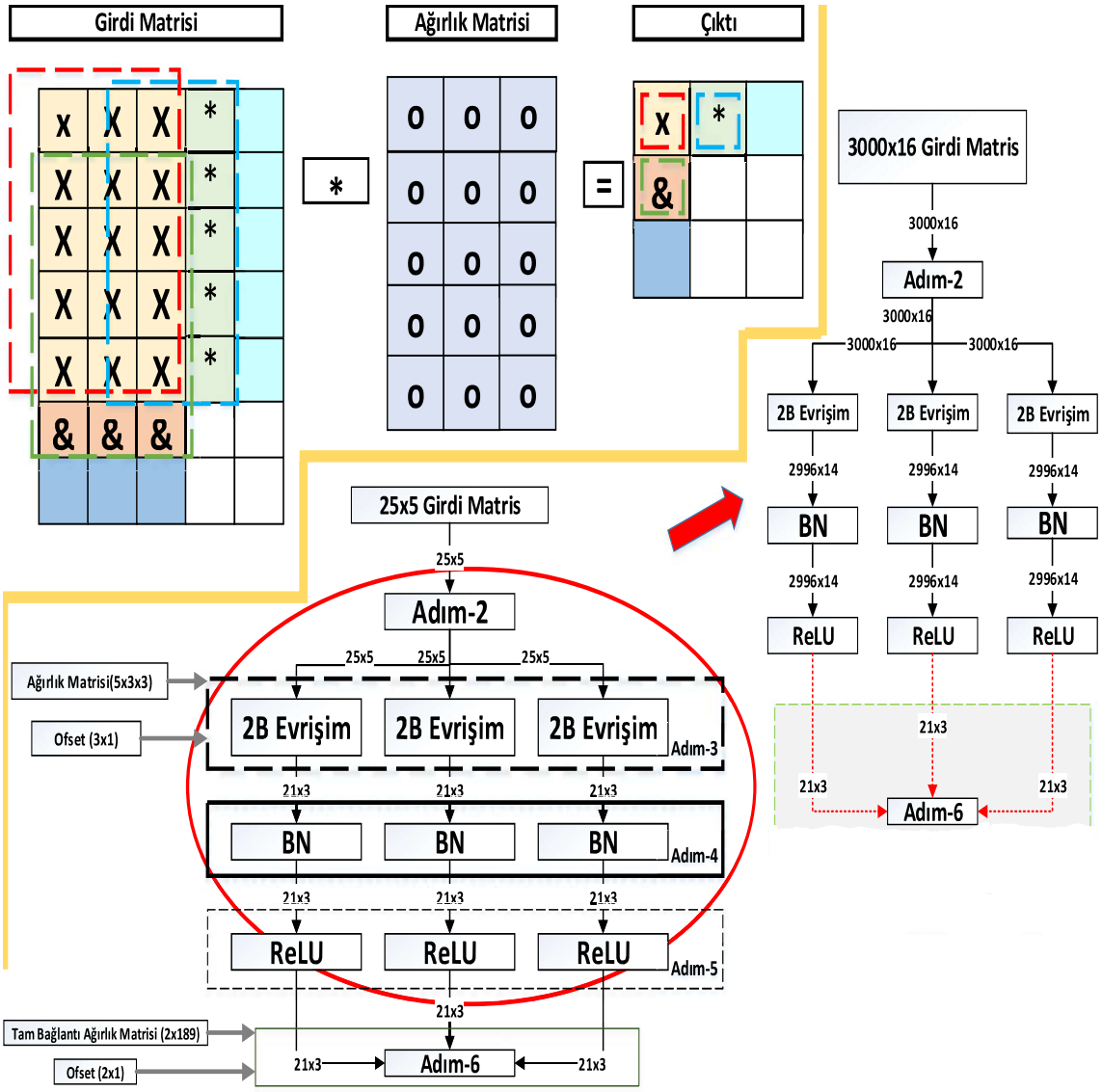
Tüm girdi matris işlemi üzerinden yapılan hesaplamada en kritik kısım Adım-3'teki evrişim işlemidir. 2B evrişim işlemi için yapılan hesaplamada sıfır ekleme (zero-padding) kullanılmadığı takdirde tüm girdi matrisinin evrişim sonucu alt küçük matrisler için evrişim sonuçlarını blok halinde içermektedir. İlgili tarama bloğu seçilerek adımlar devam ettirilebilmektedir. 2B evrişim işlemi filtrelemeden dolayı oluşan grup gecikmelerini eleyerek çıktı oluşturulmaktadır. Bu durum Eşitlik 10'da açıklanırken evrişim sonucu matrisin yükseklik ve genişlik boyutları O_H ve O_W ile gösterilmektedir. Evrişim girdi matrisi için aynı değerler I_H ve I_W ile tanımlanırken, ağırlık matrisinin boyutları ise W_H ve W_W olarak belirtilmektedir.

$$O_H = I_H - W_H + 1\,, O_W = I_W - W_W + 1\,. \tag{10}$$

Eşitlik 10'a göre Çizelge 5.2'deki parametreler Adım-3'e uygulandığında evrişim işleminin çıktısı $21x3$ boyutlu bir matris olmaktadır. Taramalar boyunca her bir tarama adımında 3 kanallı $21x3$'lük matrisler elde edilerek Adım-4 ve Adım-5 işlemleri gerçekleştirilir. Bu işlemlerden sonra 3 kanallı $21x3$'lük matris dizilerek $189x1$'lik vektör elde edildikten sonra Adım-6'daki matris çarpımı yapılır. Adım-3'te elde edilen her bir $21x3$'lük matris tüm girdi matrisi ile yapılacak evrişim işlemi sonucundaki çıktı matrisinden de seçilebilmektedir. $3000x16$'lık girdi matrisi 2B evrişim sonucunda oluşan 3 kanallı $2996x14$'lük çıktı matrisinde menzil ve darbe yönünde bölgesel olarak $21x3$'lük parça olarak alınabilmektedir. Fakat Adım-4 ve Adım-5 veri ve eleman bağımsız işlemler içerdiğinden 3 kanallı $2996x14$'lük matris Adım-6'ya kadar kullanılmaktadır. Şekil 5.3'te bu iyileştirme örnekli bir şekilde gösterilmektedir. Şekil 5.3'te görüldüğü üzere taramalı yapılan Adım-2'den Adım-5'e kadarki işlemler bir kez yapılarak Adım-6 öncesi tarama adımları için hesaplama tamamlanmaktadır.

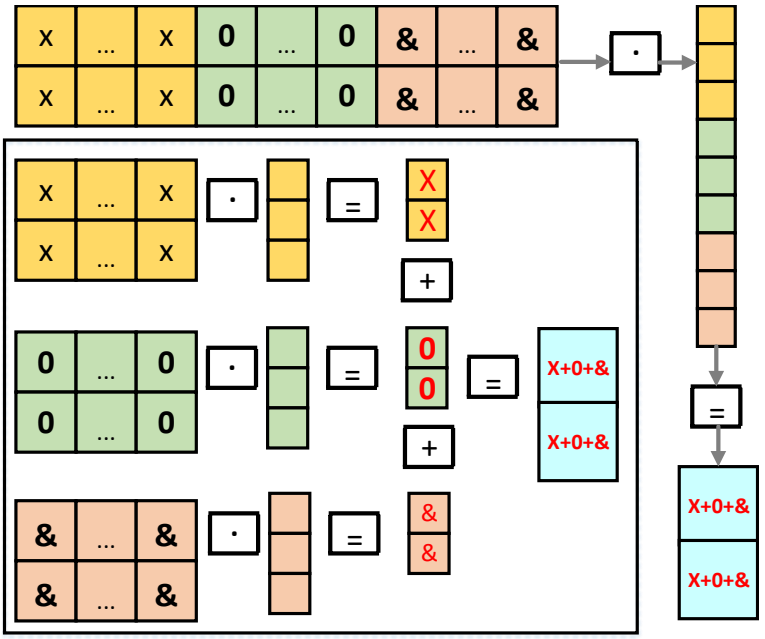
5.3.2. Sinir Ağı Katman Optimizasyonu

Paralel hesaplama tasarımı sonucunda Adım-6'daki matris çarpım işlemine kadar adımlar Şekil 5.3'te gösterildiği üzere döngüsel işlem yerine tek seferde tüm girdi matrisi üzerinden yapılarak hesaplama iyileştirmesi sağlanmıştır. Adım-6'daki matris çarpım yapısından



Şekil 5.3: Hedef Tespiti için Paralel Hesaplama Tasarımı

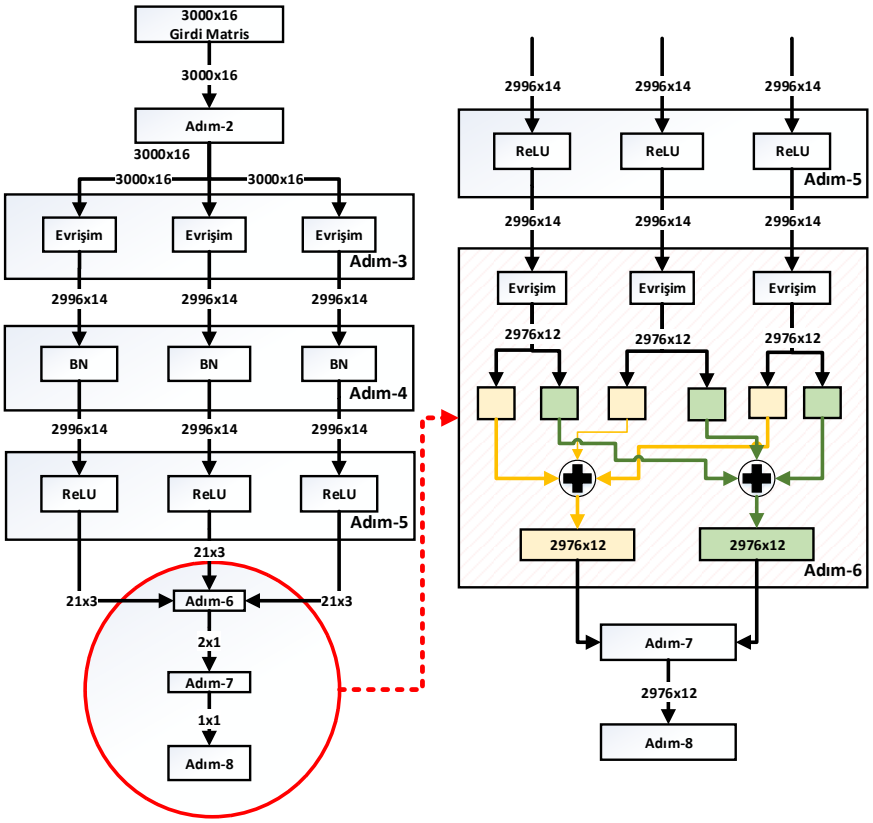
dolayı Adım-5 sonucunda $2996x14$'lük matrislerden $189x1$'lik vektör oluşturmak için ilgili tarama için her birinden $21x3$'lük matrisler seçilerek vektöre dizilir. Bu işlem her bir tarama adımında yapılacağı için hesaplama darboğazı oluşturmaktadır. Bu darboğazı aşabilmek için matris-vektör çarpımı Şekil 5.4'teki gibi bölütlenerek yapılması tezde önerilmektedir. Şekil 5.4'teki kanallara bölünen matris-vektör çarpım işlemi $2x63$'lük 3 adet matris ile bölütlenebilmektedir. Hesaplama darboğazını aşabilmek için, Tam Bağlantılı Katman ile Evrişim Katmanı arasındaki benzerlik incelenerek matris çarpım işleminin dönüşümü, tıpkı literatürde Ma ve arkadaşının önerdiği dönüşüm [108] gibi, tezde çalışılmaktadır.



Şekil 5.4: Tam Bağlantı İşlem Katman'daki Matris-Vektör Çarpımının Kanallara Bölünmesi

Üç kanala bölünen matris çarpma işlemi her birinde 2 adet çıktı kanalı oluşturacak 3 girdi kanallı 2B evrişim işlemi ile aynı sonucu vermektedir. Evrişim işleminde ağırlık matrisi ile örtüşen her bir girdi matris bloğu ile iç çarpım sonucu elde edilen değer, kanalın matris-vektör çarpımı olarak sonuçlandırılır. Üç kanalın ayrı olarak hesaplanan matris-vektör çarpımları Şekil 5.4'teki gibi toplanarak çıktı sonucu olarak elde edilir.

Dönüşüm doğrultusunda Adım-6'ya 3 adet $2996x14$'lük matris girdi olarak sağlanır. Her bir matris 2 kanallı $21x3$'lük ağırlık matrisi ile 2B evrişim işlemine sokulur. Evrişim işlemi sonucunda 3 adet çıktı matris birbirleriyle toplanarak kanal ofseti eklendikten sonra Adım-6 işlemi tamamlanır. Taramalı bir akışta elde edilecek $2976x12$ tane Adım-6 sonucu, dönüşüm ile 2 kanallı $2976x12$'lik matriste hesaplanmış durumdadır. Şekil 5.5'te Adım-6 için yapılan dönüşüm sonrası işlem ve veri akışı gösterilmektedir. Katman mimarisi değişikliği sayesinde taramalı şekilde yapılan Adım-6, Adım-7 ve Adım-8 artık tek seferde yapılarak paralel programlamaya uygun hale getirilmiştir. Çizelge 5.4'teki sonuca göre iyileştirmeler ile 32-bit kayar nokta için yaklaşık **3.17** MB bellek gereksinimine ihtiyaç duyulmaktadır. Çizelge 5.5'te ise iyileştirmeler sonrası oluşan hesaplama karmaşıklığı açıklanmaktadır. İki iyileştirmenin sonucu olarak hesaplama karmaşıklığı, işlem sayısı, 8.5x azaltılmıştır.



Şekil 5.5: Tam Bağlantı İşlem Katman Optimizasyonu - Veri ve İşlem Akışı

Algoritma Adımları	**Bellek Kullanımı**	**Değerler (Örnek Sayısı)**
Adım-1	MN	48000
Adım-2	MN	48000
Adım-3	C(M-P+1)(N-Q+1) + CPQ + C	125880
Adım-4	C(M-P+1)(N-Q+1)	125832
Adım-5	C(M-P+1)(N-Q+1)	125832
Adım-6	SC(M-X+1)(N-Y+1)+ S(M-X+1)(N-Y+1)+ SC(X-P+1)(Y-Q+1) + S	286076
Adım-7	(M-X+1)(N-Y+1)	35712
Adım-8	(M-X+1)(N-Y+1)	35712
Toplam	3C(M-P+1)(N-Q+1)+2MN+CPQ+C+S SC((M-X+1)(N-Y+1) + (X-P+1)(Y-Q+1)) (S+2)(M-X+1)(N-Y+1)	831044

Çizelge 5.4: Hedef Tespiti için İyileştirmeler Sonrası Bellek Gereksinimi

Algoritma Adımları	MAC İşlem Sayısı	Değerler
Adım-2	MN	48000
Adım-3	C(M-P+1)(Y-Q+1)PQ	1887480
Adım-4	C(M-P+1)(N-Q+1)	125832
Adım-5	C(M-P+1)(N-Q+1)	125832
Adım-6	SC(M-X+1)(N-Y+1)(X-P+1)(Y-Q+1)	1349936
Adım-7	S(M-X+1)(N-Y+1)	83328
Toplam	SC(M-X+1)(N-Y+1)(X-P+1)(Y-Q+1)+ MN + C(M-P+1)(Y-Q+1)(PQ + 2) + S(M-X+1)(N-Y+1)	$1.57x10^7$

Çizelge 5.5: Hedef Tespiti için İyileştirmeler Sonrası Hesaplama Karmaşıklığı

Gerçekleşen paralel hesaplama mimarisi ve katman optimizasyonları ile her adım tek seferde yapılarak taramalı hesaplamadan soyutlaştırılmıştır. Hesaplama ve algoritma tabanlı iyileştirmeler paralel programlama ile desteklenerek yüksek başarım hedeflenmektedir.

5.4. CPU Paralel Programlama Tasarımları

Algoritma 2'deki alt adımlar, paralel hesaplama tasarımları ve katman optimizasyonları sayesinde çok çekirdekli CPU mimarilerinde paralel işleme ile yapılabilmektedir. Paralel hesaplamalar ile hesaplama yükü azaltılırken bu başarımı paralel işleme ile birleştirerek yüksek başarımlar elde edilebilmektedir. Çizelge 5.6'da belirtildiği üzere bu bölümde çok çekirdekli CPU tabanlı gerçekleştirilen paralel işleme yöntemleri belirtilmektedir. Intel tabanlı işlemciler için özelleşmiş oneAPI-DNN mimarisi, makine öğrenmesi modellerini etkin ve verimli bir şekilde CPU mimarisinde çalıştırabilmektedir [109, 110]. SIMD işlemleri için Intel C++ Compiler (icc) üzerinden oto-vektörize yapılırken, OpenMP işlemleri için de OpenMP hesaplama seçenekleri ve direktifleri kullanılmaktadır.

Algoritma Adımları	CPU Paralel İşleme Teknikleri
Adım-2	SIMD+OpenMP+Döngü Açma (Loop Unrolling, LU)
Adım-3	OpenMP + oneDNN+SIMD
Adım-4	SIMD+OpenMP+ LU + Döngü Sentezi (Loop Fusion, LF)
Adım-5	SIMD+OpenMP+LU+LF
Adım-6	OpenMP+oneDNN+SIMD
Adım-7	SIMD+OpenMP

Çizelge 5.6: Hedef Tespiti için CPU Paralel İşleme Yöntemleri

oneAPI, heterojen hesaplama mimarileri için bir soyutlama katmanı oluşturarak hızlandırıcı birimlerine işlevsel özellikler sağlamaktadır [111]. Adım-3 ve Adım-6'daki evrişim işlemleri için oneDNN veri yapısı ve mimarisinden yararlanılmaktadır. Çizelge 5.7'de evrişim adımları için hazırlanan veri yapısı ve tasarımı belirtilmektedir.

Algoritma Adımları	N	I_C	I_H	I_W	K_C	K_H	K_W	O_C	O_H	O_W
Adım-3	1	1	3000	16	3	5	3	3	2996	14
Adım-6	1	1	2996	14	2	21	3	2	2976	12

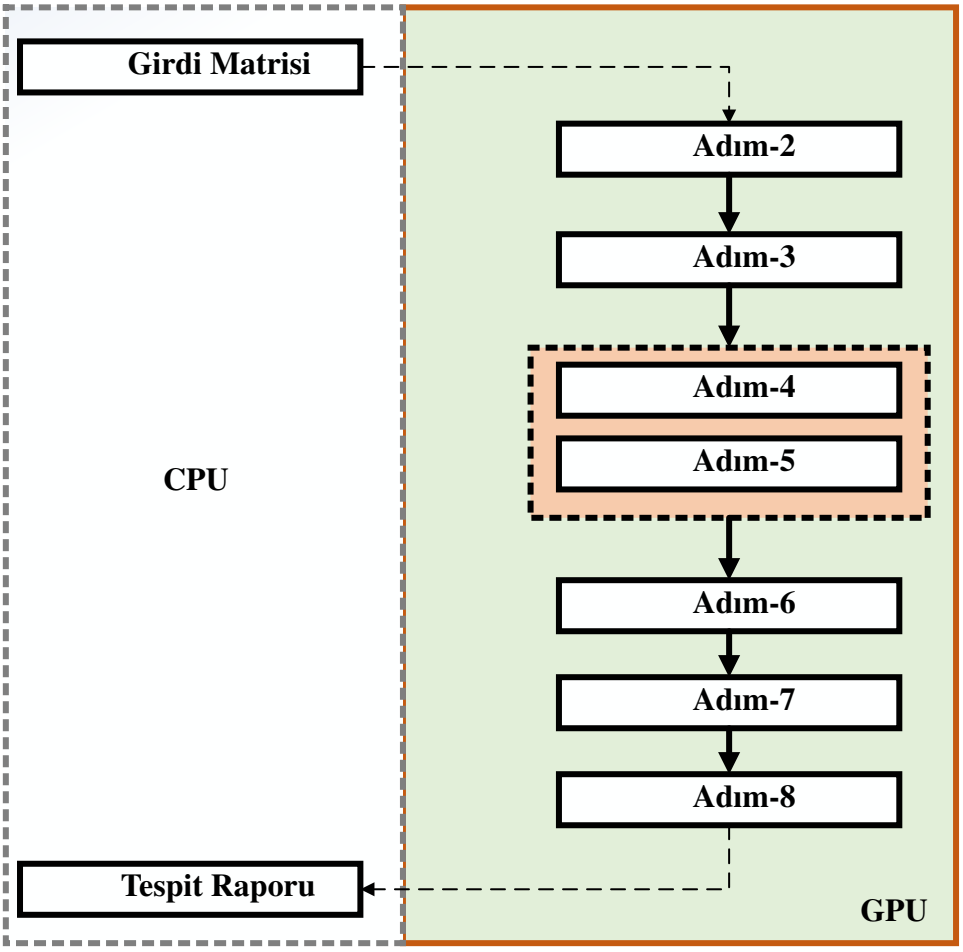
Çizelge 5.7: oneDNN ile Evrişim İşlemleri için Veri Boyutu ve Tasarımı

Çizelge 5.7'de N grup sayısını, I_C girdi kanal sayısını, I_H ve I_W evrişim girdi matrisi yüksekliği ve genişliğini göstermektedir. Evrişim kanal sayısı K_C ile ifade edilirken, K_H ve K_W ağırlık matrisinin yükseklik ve genişliğini belirtmektedir. Evrişim çıktı matrisinin kanal sayısı O_C ile belirtilirken O_H ve O_W ise yükseklik ve genişliğini gösterir.
Intel oneDNN kullanımında evrişim hesaplamasından önce evrişim yapılandırması gerekmektedir. Yapılandırma işleminde önbellek yerelleştirmesi optimizasyonu için yeniden sıralama yapılmaktadır. Yeniden sıralama ile ağırlık matrisi çarpımı verimli hesaplanabilmektedir. Yapılandırma sonrası Çizelge 5.7'deki değerlere göre bir kez hazırlanan evrişim yapıları Adım-3 ve Adım-6 sırasında kullanılmaktadır. Çizelge 5.7'de oneDNN dışında AVX-512 yazmaç mimarisi ile SIMD işleme gerçekleştirilerek tek komutla aynı anda 16 adet 32-bit kayar nokta için hesaplama yapılmıştır [112]. Bu optimizasyonlar Adım-2 ve Adım-4'te gerçekleştirilmektedir. OpenMP ile matrisler bölütlenerek diğer CPU'lar ile aynı anda hesaplanmaktadır. Döngü açma ve sentezi ile önbellek yerelliliği artırılarak gecikmeler azaltılmıştır.

5.4.1. Başarım Testleri

Intel Xeon W-2123 CPU birimiyle başarım testleri gerçekleştirilmektedir. Herhangi bir iyileştirme yapılmadan Algoritma 2'de bahsedildiği gibi adımlar seri hesaplama şeklinde gerçekleştirildiğinde hedef tespit uygulaması Çizelge 5.2'deki parametreler ile yaklaşık **82759** ms sürmektedir. Gerçekleşen sonuçlar maddeler halinde belirtilmektedir:

- Adım-3 için gerçekleştirilen (8.5x) hesaplama optimizasyonu, SIMD AVX-512 kullanımı ve OpenMP kullanımı ile hedef tespit işlemi ortalama 107 ms sürmektedir.
- Adım-6 ile gerçekleşen katman optimizasyonu ile taramalı işlem yerine tek seferde işlem yapılması sayesinde toplam hedef tespit işlem süresi 1.5 ms sürmektedir.
- Darboğaz işlemlerin, gerçekleştirilen koşut işleme tabanlı iyileştirmeler ile gerçek zamanlı başarıma etkisi azaltılmıştır. Bellek gereksinimi 16x artmıştır.



Şekil 5.6: Makine Öğrenmesi tabanlı Hedef Tespiti için Heterojen Hesaplama Mimarisi

5.5. GPU ile Paralel Hesaplama

Hesaplama iyileştirmelerinin GPU hesaplama yapısına uyumluluğu incelendiğinde hesaplamaların SIMT tabanlı bir mimariye uygun işlem adımlarına dönüştüğü için CPU'ya göre

daha avantajlı bir tasarım yapılabileceği görülmektedir. GPU hesaplamada taramalı yöntem ile her bir adım CPU'dan tekrar çağrılarak "kernel çağırma" ek yüküne neden olurken adımların tek seferde yapılarak hesaplamaya ek yükü azaltılmaktadır. GPU mimarileri içerdikleri iş parçacığı, akış çoklu işlemci ve kernel yapısı sayesinde CPU'ya göre aynı anda daha fazla işlem yapılmasına olanak sağlamaktadır [113]. Tez çalışmasında GPU ile paralel hesaplama yaparken hedef tespitindeki tüm işlem adımları GPU'da hesaplanacak şekilde tasarım yapılmaktadır. Bu tasarım için CPU-GPU heterojen mimari akış şeması Şekil 5.6'da gösterilmektedir.

Algoritma 2'deki $3000x16$'lık girdi matrisi Şekil 5.6'daki gösterildiği gibi CPU'dan GPU'ya transfer edilip GPU hesaplamaları gerçekleştirildikten sonra Adım-8'deki karşılaştırma sonucu tespit çıktıları CPU'ya transfer edilmektedir. Adım-4 ve Adım-5 aynı kernel içinde yapılarak kernel çağırma ek yükü azaltılmakta ve GPU önbelleğinin kullanılırlığı artmaktadır. Adımlar için kullanılan mimari yapı Çizelge 5.8'de gösterilmektedir.

Algoritma Adımları	**Blok Sayısı**	**Bloktaki İş Parçacığı Sayısı**
Adım-2	48	1000
Adım-4-5	147	856
Adım-7	93	768
Adım-8	48	744

Çizelge 5.8: Hedef Tespiti için Evrişim Adımları hariç CUDA Mimarisi

Evrişim işlemleri için NVIDIA CUDA cuDNN [114] v7.6.5 sürümü işlevleri kullanılarak evrişim yapısı hazırlanmaktadır. Evrişim parametreleri Çizelge 5.9'da verilmektedir.

Algoritma Adımları	**Girdi (I) Ağırlık (K) Çıktı (O)**	**N**	**C**	**H**	**W**
Adım-3	I	1	1	3000	16
	K	1	3	5	3
	O	1	3	2996	14
Adım-6	I	1	1	2996	14
	K	1	2	21	3
	O	1	2	2976	12
Evrişim Algoritması		IMPLICIT PRECOMP GEMM			

Çizelge 5.9: Hedef Tespiti için Evrişim Adımları için CUDA Mimarisi ve Veri Yapısı

Evrişim işleminde kullanılan algoritma yöntemi $1x1$'lik evrişim filtrelerinde diğer evrişim yöntemlerine göre yüksek başarım göstermekte olup ağırlık matrisi filtreleri $3x3$'lük ve $5x5$'lik matris yapılarında olmadığı için Winograd ve FFT tabanlı yöntemler ile yüksek başarım elde edilememektedir [115]. Matris çarpımının etkin yapıldığı fakat bellek gereksinimin arttığı "IMPLICIT PRECOMP GEMM" yöntemi ile başarımlı sonuçlar alınmıştır. NVIDIA GTX 1080 GPU ile hedef tespit işleminin adımları için "NVIDIA Profiler" aracı ile başarım profilleri çıkarılmıştır [116]. Başarım profillerine ait sonuçlar Çizelge 5.10'da gösterilmektedir. Sonuçlarda görüldüğü üzere evrişim işlemleri diğer işlemlere göre oldukça baskın bir sürede hesaplanmaktadır. Makine öğrenmesi tabanlı hedef tespiti işleminde GPU ile paralel hesaplama başarım artışı sağlanmaktadır.

Algoritma Adımları	**Hesaplama İşlevi**	**Hesaplama Yüzdesi (%)**
Adım-3	maxwell_scudnn_128x32_relu_interior_nn (cuDNN)	12.8
Adım-2	custom_kernel	1.5
Adım-4-5	custom_kernel	2.2
Adım-6	maxwell_scudnn_128x32_relu_large_nn (cuDNN)	78.5
Adım-7	custom_kernel	1.4
Adım-8	custom_kernel	2.4
Adım-1	memset + copy_kernel	1.2

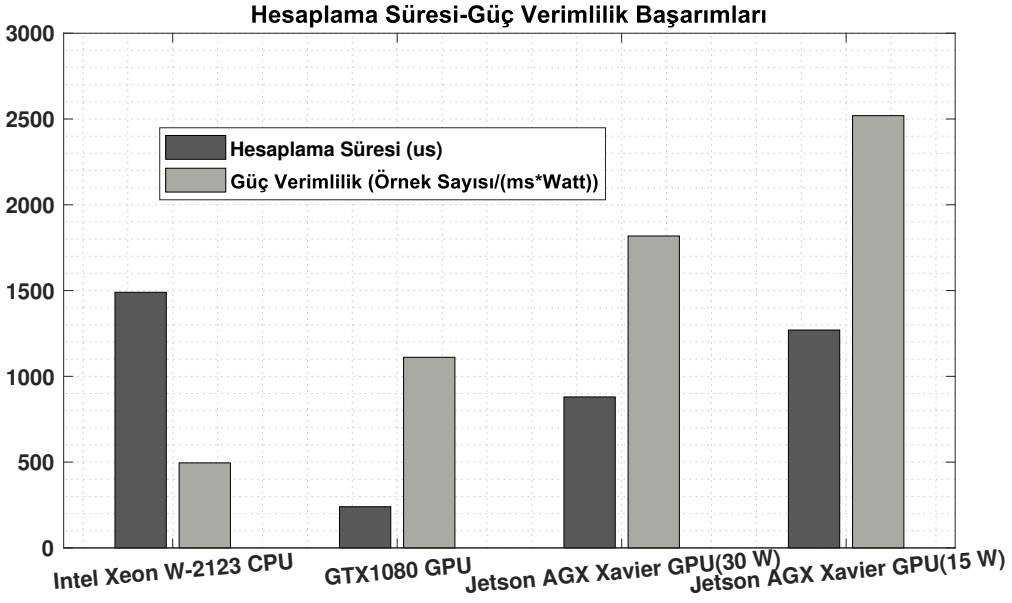
Çizelge 5.10: Hedef Tespiti için GPU Paralel Hesaplama Adım Yüzdeleri

NVIDIA GTX 1080 GPU ile Çizelge 5.10'daki profil sonuçlar doğrultusunda veri transferi dahil **215** μs sürmektedir. Veri transferi süresi ortalama **29** μs sürmektedir. GPU paralel hesaplama ile CPU koşut işlemeye göre hesaplama başarımı **8x** kat hızlandırılmıştır. CPU-GPU iletişim ek yükü dahil edildiğinde **7x** kat başarım iyileştirilmiştir.

5.6. Güç Verimli Hesaplama

Jetson AGX Xavier GPU hesaplama birimleri ile hem 15 W hem de 30 W güç tüketimi seçeneklerinde yüksek başarımlar elde edilmektedir [66]. Düşük güç tüketen Jetson hesaplama mimarisi 30 W güç tüketiminde **875** μs süren hesaplama (5 μs veri iletişimi), 15 W güç modunda **1246** μs (10 μs veri iletişimi) sürmektedir. Jetson GPU ile paralel hesaplama

sayesinde CPU koşut işlemeye göre 30 W modunda **1.7x** kat hızlanma sağlanırken, 15 W güç tüketiminde **1.2x** kat hızlandırılmıştır. Güç verimli hesaplamalar yapılırken verimlilik analizleri ile hesaplama başarımları analiz edilmektedir. Şekil 5.7'de makine öğrenmesi tabanlı hedef tespiti için CPU ve GPU birimleri hesaplama süreleri ve güç verimlilik analizleri (örnek sayısı/(ms*Watt)) gösterilmektedir.



Şekil 5.7: Makine Öğrenmesi tabannlı Hedef Tespiti için Güç Verimlilik Karşılaştırmaları

Makine öğrenmesi tabanlı hedef tespit işlemi için düşük güç tüketen hesaplama birimleri ile paralel hesaplama başarımlı çözümler ortaya koyulmuş olup sensör mimarilerinde etkin ve düşük maliyetli (güç tüketimi, ağırlık, soğutma vb.) birimlerin kullanılmasını sağlanmıştır.

5.7. Tartışma

Bu tezde, CNN modeli ile radar hedef tespiti için gerçek zamanlı geliştirme tasarımları çalışılmıştır. CNN modelinin alt adımları bellek gereksinimi ve hesaplama karmaşıklığı derecesinde ayrıntılı olarak incelenmiştir. CNN modeli tıpkı geleneksel yöntem SYAO tabanlı hedef tespiti ile benzerlik gösterecek şekilde taramalı hesaplama olarak tasarlandığından [1] tekrarlı veri ve işlem nedeniyle hesaplama gecikmelerine neden olmaktadır. Gecikmeleri

incelemek için gerçekleştirilen darboğaz analizleri sonucunda Evrişim Katmanı ile Tam Bağlantı İşlem Katmanı'ndaki işlemlerin hesaplama darboğazına neden olduğu görülmüştür. Darboğazları aşmak için hesaplama tabanlı iyileştirmeler yapılarak hesaplama için gereken işlem sayısı **8.5x** kat azaltılırken, bellek gereksinimi **16x** kat artırılmıştır. Paralel hesaplama tabanlı iyileştirme ile Adım-6'ya kadar olan tüm adımlar taramalı yerine tek seferde hesaplanacak şekilde tasarlanmıştır. Katman optimizasyonu ile Tam Bağlantı İşlem Katmanı'nda matris çarpım işlemleri evrişim işlemlerine dönüştürülerek paralel işlemeye uygun bir yöntem önerilmiştir. Evrişim işlemleri için oneDNN [110] yapıları kullanılmış olup gerçek zamanlılığı sağlayacak hesaplama süreleri elde edilmiştir. Paralel hesaplamaya uygun CPU-GPU heterojen mimari tasarımı sayesinde GPU hesaplama ile CPU koşut hesaplamaya göre **7x** kat hızlanma elde edilmiştir. Güç verimli Jetson GPU mimarileri 15W güç modunda ile CPU koşut hesaplamaya göre **1.2x** hızlanma elde edilmiştir. Ayrıca Şekil 5.7'de gösterildiği üzere Jetson GPU ile NVIDIA GTX 1080 GPU'ya göre yaklaşık **2.4x**, Intel Xeon W-2123 CPU'ya göre ise yaklaşık **5x** güç verimli hesaplama sağlanmıştır.

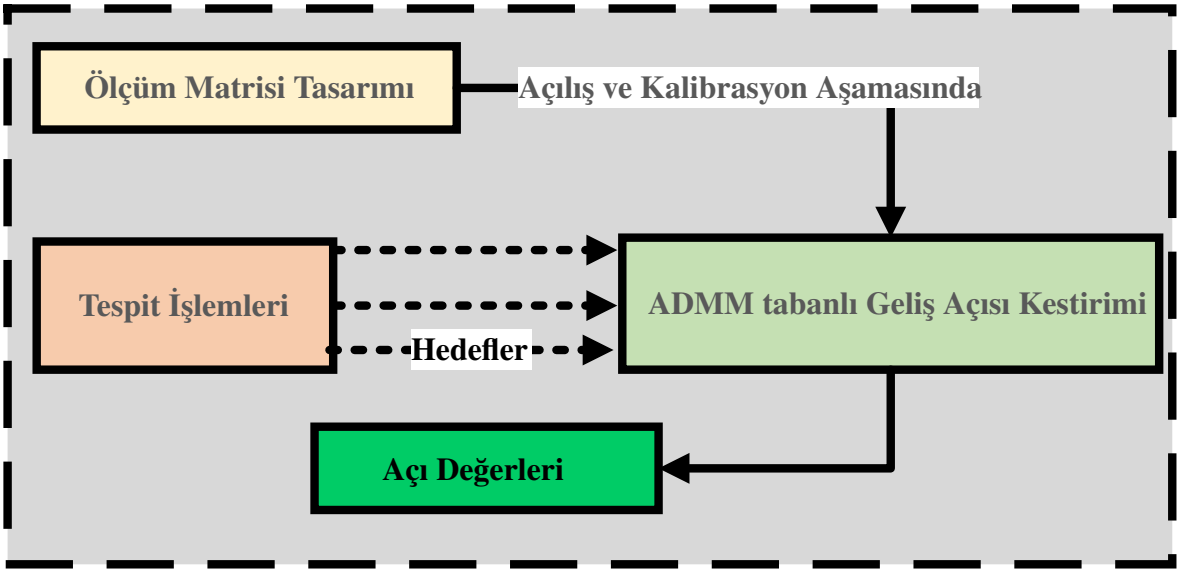
Güç verimli paralel hesaplamalar ile mobil sensör sistemlerinde düşürülen güç tüketimi ile uzun batarya ömürlü hedef tespiti senaryoların gerçekleştirilmesine yönelik tasarımlara esneklik ve kazanım sunulmuştur. Bu bölümdeki çalışmalar, hedef tespiti CNN modelinin gerçek zamanlı bir uygulamada çalışabilmesi için darboğazlara karşı iyileştirmeleri amaçlamakta gerçekleşen hızlanmalar ile hedefine ulaşmıştır. Literatürde CNN tabanlı radar hedef tespitine yönelik hesaplama başarım analizleri için çalışmalar az görülmektedir. SAR radarı için hedef tespitini evrişim ve havuzlama (pooling) katmanlarından oluşan SYAO yöntemini kullanan çalışmada [117], $1478x1784$'lük imge üzerinde hedef tespiti GTX 1080 Ti GPU ile 192 ms sürmektedir. $3000x16$ boyutundan 55x büyük bir imge matrisi üzerinde 192 ms hesaplama süresi doğrusal bir oran ile 3.5 ms civarında yapılabilmektedir. 3x3'lük evrişim filtreleri kullanılmaktadır. Gerçekleştirilen bu hesaplama ile tez çalışmasındaki Jetson GPU (15 W) başarımı karşılaştırıldığında Jetson GPU ile yaklaşık 2.8x hızlı başarım alınmaktadır. Gelecek çalışmalarda 16-bit kayar nokta için GPU başarımlarının incelenmesi planlanmakta olup hesaplama başarımı ve doğruluk analizlerinin karşılaştırılması hedeflenmektedir. Uygulanan hesaplama optimizasyonlarının çeşitli sensör mimarilerinde nesne ve anomali tespiti problemlerinde kullanılması önerilmektedir.

6. SIKIŞTIRILMIŞ ALGILAMA TABANLI GELİŞ AÇISI KESTİRİMİ

Bu bölümde, sensör sisteminde tespit işlemleri sonrası elde edilmiş hedefler için sıkıştırılmış algılama tabanlı geliş açısı kestirimine yönelik gerçekleştirilen hesaplamalardan bahsedilmektedir. Geliş açısı kestirimi işlemi gerçek zamanlı bir sensör mimarisinde yapıldığında tıpkı sayısal alt çevrimi ve tespit işlemi gibi anlık veri işleme periyodunda yapılmalıdır. Bu gerekliliği dikkate alarak geliş açısı kestirimi işlemi için seri hesaplama analizleri yapıldıktan sonra paralel hesaplama ve güç verimli hesaplama tasarımları çalışılmaktadır. ADMM tabanlı geliş açısı kestirimi için algoritma adımları incelenmektedir. Ayrıca geliş açısı kestirimi öncesinde yapılan ölçüm matrisi tasarımı algoritması için gerçekleştirilen hesaplama çalışmalarından bahsedilmektedir. Ölçüm matrisi tasarımı için yapılan hızlandırma karşılaştırılmalı olarak anlatılmaktadır.

6.1. Algoritma Modeli

Sıkıştırılmış algılama tabanlı geliş açısı kestirimi iki temel işlem ile yapılmaktadır. Geliş açısı kestirimi yapılmadan önce ölçüm matrisi tasarımı gerçekleştirilmektedir. Şekil 6.1'de ölçüm matrisi tasarımı ve geliş açısı kestirimi bağlantısı gösteren bir akış bulunmaktadır.



Şekil 6.1: Sıkıştırılmış Algılama tabanlı Geliş Açısı Kestirimi Modeli

Şekil 6.1'de gösterildiği üzere ölçüm matrisi tasarımı anlık olarak çalıştırılamayacağından dolayı hesaplama analizine dahil edilmemektedir. Çünkü bu işlem uygulamanın açılışında veya belirli güncelleme/kalibrasyon anlarında yapılmasına ihtiyaç duyulmaktadır. Fakat uyarlamalı senaryolarda çalışabilmesi düşünülerek ölçüm matrisi tasarımının hızlı ve etkili çalışmasını sağlamaya yönelik darboğaz analizleri ve paralel hesaplamalar gerçekleştirilmektedir.

6.1.1. Ölçüm Matrisi Tasarımı

Sıkıştırılmış algılama yaklaşımıyla ölçüm sayısı azaltılarak sensörlerden sıkıştırılmış ölçümler gerçekleştirilebilmektedir. Sıkıştırılmış ölçümlerin düzgün şekilde alınması açı kestirimi açısından kritik önem arz etmektedir. Sıkıştırma işleminin nasıl yapılacağı, ölçüm matrisi adı verilen bir matris ile belirlenmektedir. Sıkıştırılmış algılama literatüründe genelde ilinti minimizasyonu yaklaşımına dayanan ölçüm matrislerine rastlanmaktadır [118, 119]. İlinti minimizasyonu tabanlı ölçüm matrisi tasarımındaki temel amaç, ilintisiz ölçümler alınmasını sağlayarak en az ölçümle en çok bilgiyi toplayabilmektir. Ölçüm matrisi tasarımı için gerekli matematiksel işlemleri barındıran algoritma adımları Algoritma 3'te açıklanmaktadır. Açıklama kısmında girdi matris ifadeleri kalın punto ile belirtilmiştir.

Algoritma 3 Ölçüm Matrisi Tasarımı Algoritma Adımları [70]

Girdiler : L : Izgara Nokta Sayısı, M : Sensör Dizi Uzunluğu
m : Sıkıştırılmış Sensör Kanal Sayısı, **T** : Girdi Matrisi, LxL
V : Dikgen Model Matrisi, LxM, **invSU** : Sensör Tersinir Devrik Matris, MxM
Çıktı : Φ : Ölçüm Matrisi, mxM
SVD : Tekil Değer Ayrışımı (Singular Value Decomposition)
Algoritma
Adım-1 : SVD Girdi Matrisi Oluşturma, $T_P = V^{'}\dot{T}\dot{V}$, Çıktı Boyutu: MxM
Adım-2 : SVD İşlemi, Çıktı Boyutu : MxM
Adım-3 : SVD Çıktı Matrisi Sıkıştırma İşlemi, Çıktı Boyutu : mxM
Adım-4 : Tersinir Matris(invSU) ile Çarpma İşlemi, Çıktı Boyutu mxM
Adım-5 : Transpoz Çarpım ve SVD İşlemi, Çıktı Boyutu mxm
Adım-6 : Beyazlatma İşlemi ve Matris Çarpım İşlemi, mxM
Adım-7 : Matris Normu Alma ve Ölçüm Matrisi Oluşturma, Φ : mxM

Adım-6'da yapılan beyazlatma (whitening) işlemi ile problemde renkli yerine beyaz Gauss dağılımına sahip ölçümler ile hesaplama yapılması sağlanır [70]. Sıkıştırılmış algılama yaklaşımı ile ölçüm matrisi tasarımı gerçekleştirilerek açı kestirim doğruluğunun başarımı literatürdeki çalışmaya [69] göre yüksek SNR değerlerinde aynı hata oranında ortalama 2 dB kazanç olacak şekilde artmaktadır [70]. Bu tez çalışmasında, [70]'deki ölçüm matrisi tasarımı incelenmekte olup kullanılan tasarım parametreleri Çizelge 6.1'de verilmektedir.

Tasarım Parametreleri	**Değerler**
Izgara Nokta Sayısı (L)	192
Sensör Dizi Uzunluğu (M)	128
Sıkıştırılmış Sensör Sayısı (m)	16

Çizelge 6.1: Ölçüm Matrisi Tasarımı için Donanım ve Sistem Parametreleri

Bellek Kullanımı ve Hesaplama Analizi

Ölçüm matrisi tasarımı uygulaması için Algoritma 3'teki işlem adımlarının bellek kullanımı maddeler halinde gösterilmektedir:

- Adım-1'deki işlemin için karmaşık iki girdi matris, bir ara matris ve bir çıktı matris yapısıyla, $2M^2 + 4LM + 2L^2 = 2(M + L)^2$ tane veri alanına ihtiyaç duyulmaktadır.
- Adım-2'de girdi matris $2M^2$ boyutundadır. İşlem çıktısı yine $2M^2$ olmaktadır. SVD işleminde ara basamakta kullanılabilecek bellek alanları incelenmemektedir.
- Adım-3'de sıkıştırma işlemi gerçekleştiği için bellek alanında indirgeme yapılmaktadır. Bu adım için ekstra $2mM$ boyutunda bir veri alanına ihtiyaç vardır.
- Adım-4'te matris çarpım işleminde $2M^2 + 2mM$ tane ekstra bellek alanı kullanılır.
- Adım-5'te çıktı matris için $2m^2$ tane veri alanına ihtiyaç duyulur.
- Adım-6'da ise çıktı matris için $2mM$ tane veri alanına ihtiyaç vardır.
- Adım-7'de ise algoritma çıktısı ölçüm matrisi için $2mM$ tane veri alanı gerekmektedir.

Yapılan bellek kullanım analizleri sonucunda ölçüm matrisi tasarımı için toplamda $2(M + L)^2 + 4M^2 + 8mM + 2m^2$ tane veri alanı gerekmektedir. 32-bit kayar nokta veri tipi kullanım

durumunda ve Algoritma 3'deki tasarım parametreleri Çizelge 6.1'deki gibi [70] alındığında 1.94 MB bellek kullanımı gerçekleşmektedir. Bellek kaynak kullanımı gelişmiş bilgisayar mimarilerinde önemsenmeyecek kadar azdır. Ölçüm matrisi tasarımında hesaplama karmaşıklık analizi yapıldığında karşımıza Çizelge 6.2'de gibi bir sonuç ortaya çıkmaktadır. Çizelge 6.2'de SVD işlemlerin karmaşıklığı $O(n^3)$ olarak alınmaktadır [120].

Algoritma Adımları	**MAC İşlem Sayısı**
Adım-1	$ML^2 + M^2L$
Adım-2	M^3
Adım-3	0
Adım-4	mM^2
Adım-5	$m^2M + m^3$
Adım-6	mM^2
Adım-7	mM

Çizelge 6.2: Ölçüm Matrisi Tasarımı MAC İşlem Sayısı

Çizelge 6.2'ye göre ölçüm matrisi tasarımı için toplam MAC işlem sayısı $M^3 + 2mM^2 + m^2M + ML^2 + M^2L + mM + m^3$ şeklinde olmaktadır. Çizelge 6.1'deki tasarım parametreleri kullanımında MAC işlem sayısı değeri yaklaşık 1.01×10^7 olmaktadır.

Darboğaz Analizi

Ölçüm matrisi tasarımı uygulamasında hesaplama sürelerini analiz etmek ve hızlandırma çalışmalarını yapmak için öncelikle darboğaz işlem noktaları tespit edilmektedir. Darboğaz işlemleri belirleyebilmek için CPU'da seri işleme çalışma yüzdeleri üretilmektedir. Seri işleme performansları Intel VTune Amplifier aracı ile çıkarılmaktadır [121]. Çalışma süresi yüzde ölçümleri toplam seri çalışma süresi ölçümlerine göre oranlı olarak hesaplanmaktadır. Çizelge 6.3'teki sonuçlara göre SVD işlemi ve matris çarpım işlemleri en büyük darboğaz işlemler olduğu için bu işlemlere yönelik iyileştirmeler çalışılmaktadır. Seri hesaplama Intel Xeon E5-2637 CPU ile gerçekleştirilmiş olup toplamda 65 ms sürmektedir [20].

CPU ve GPU Koşut İşleme Başarımları

Ölçüm matrisi tasarımı uygulamasındaki darboğaz işlemler için hem CPU hem de GPU'da başarım ölçümleri alınarak verimli bir paralel programlama tasarımı hedeflenmektedir. SVD

Darboğaz İşlem Noktaları	Çalışma Süre Yüzde Ölçümü (%)
SVD İşlemi (Adım-2 ve Adım-5)	%60
Matris Çarpım İşlemleri (Adım-1, Adım-4, Adım -5 ve Adım-6)	%32
Diğer İşlemler	%8

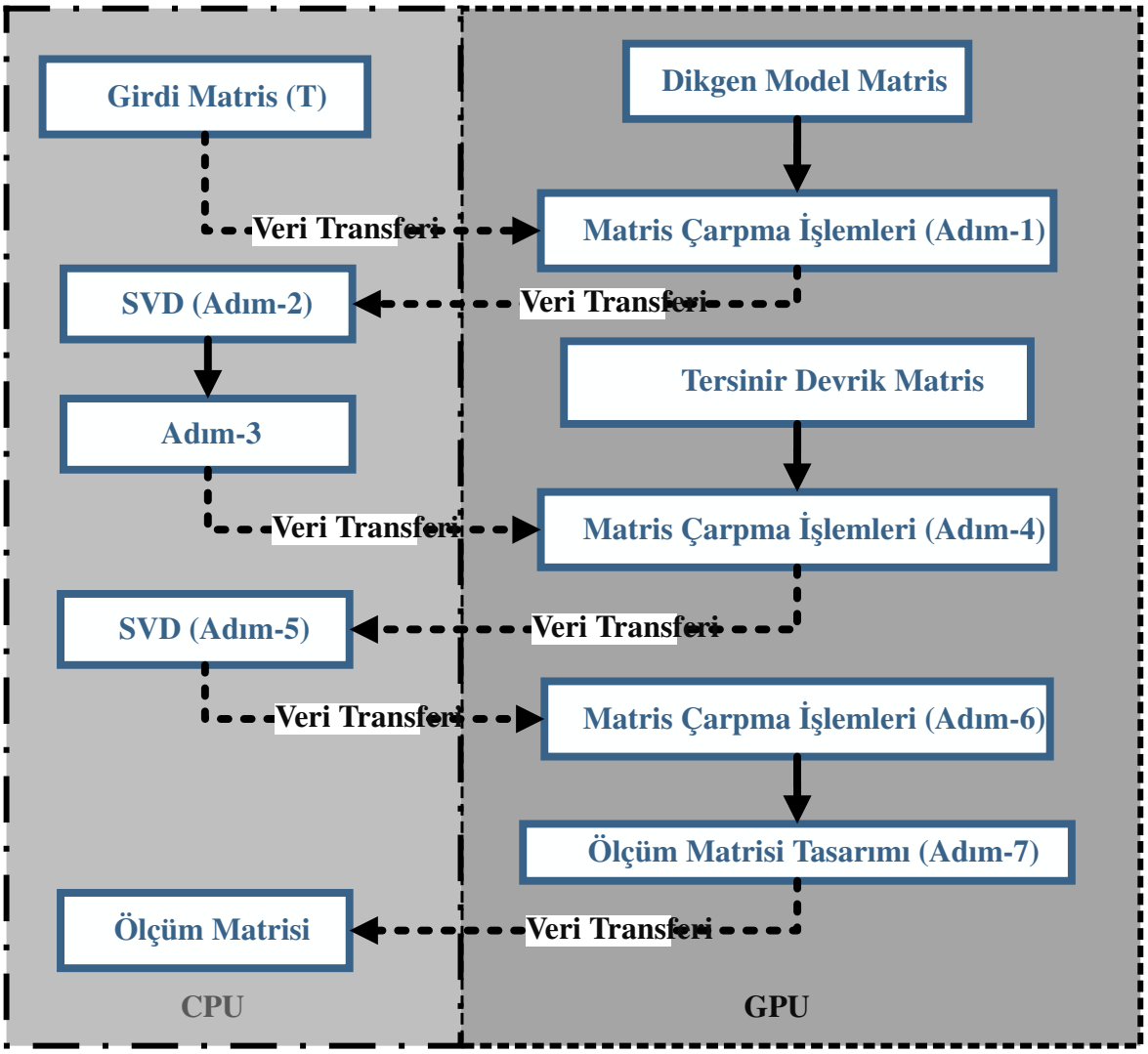
Çizelge 6.3: Ölçüm Matrisi Tasarımı için CPU Seri Çalışma Darboğaz İşlem Yüzdeleri

tabanlı yöntemler istatiksel problemlerin çözümünde kullanılan Temel Bileşen Analizi (Principal Component Analysis, PCA) işlevinde kullanılmaktadır [122]. SVD işlemi için optimizasyon sağlayan kütüphaneler ile yüksek başarım elde etmek mümkündür. Matris çarpım işlemi de donanıma özel kütüphaneler ile hızlandırılmaktadır. Çizelge 6.4'te SVD ve matris çarpım işlemleri için CPU tabanlı Eigen [123], Intel MKL [124] ve GPU tabanlı cuBLAS [125], cuSOLVER [126] kütüphanelerine ait fonksiyonlar kullanılmaktadır.

Çalışma Süreleri (us)	**Matris Çarpım İşlemi**				**SVD İşlemi**			
Matris Boyutu	Eigen	MKL (S)	MKL (P)	cuBLAS	Eigen	MKL(S)	MKL(P)	cuSOLVER
16x16	10	5	5	5	21	24	25	356
32x32	17	8	7	7	157	115	119	639
64x64	68	31	13	9	774	557	671	2292
128x128	289	142	54	11	3350	3181	4521	7341
256x256	2151	985	412	31	12351	24378	13329	32184
512x512	14541	10186	4345	122	45682	218162	51071	150136

Çizelge 6.4: SVD ve Matris Çarpım İşlemleri için Farklı Mimari ve Kütüphaneler ile Başarım Ölçümleri, Tek Çekirdek (S), Çoklu Çekirdek (P)

Çizelge 6.4'te görüldüğü üzere ve literatürdeki çalışmanın [127] sonucu üzere SVD işleminde 512x512 boyutunda ve daha küçük boyutlu matrislerde CPU biriminde MKL ile alınan hesaplama verimi daha yüksektir. Ölçümler doğrultusunda heterojen mimari kullanımında matris çarpım işlemlerinin GPU biriminde, SVD işleminin ise CPU'da çalışması hesaplama süresini azaltmaktadır. Şekil 6.2'de ölçüm matrisi tasarımı için CPU-GPU heterojen mimarisi yer almaktadır. CPU-GPU heterojen mimarisi çok sayıda veri transferi içermesi ve işlemlerin ardışık yapılması gereğinden dolayı hesaplama veriminin düşük olması öngörülmektedir. Ölçüm matrisi tasarımının sadece CPU uygulaması ve CPU-GPU heterojen uygulaması için gerçekleşen hesaplama süresi başarımları Çizelge 6.5'te gösterilmektedir. Hesaplama süreleri için Intel Xeon E5-2637 CPU ve NVIDIA GTX 1080 GPU kullanılmaktadır.



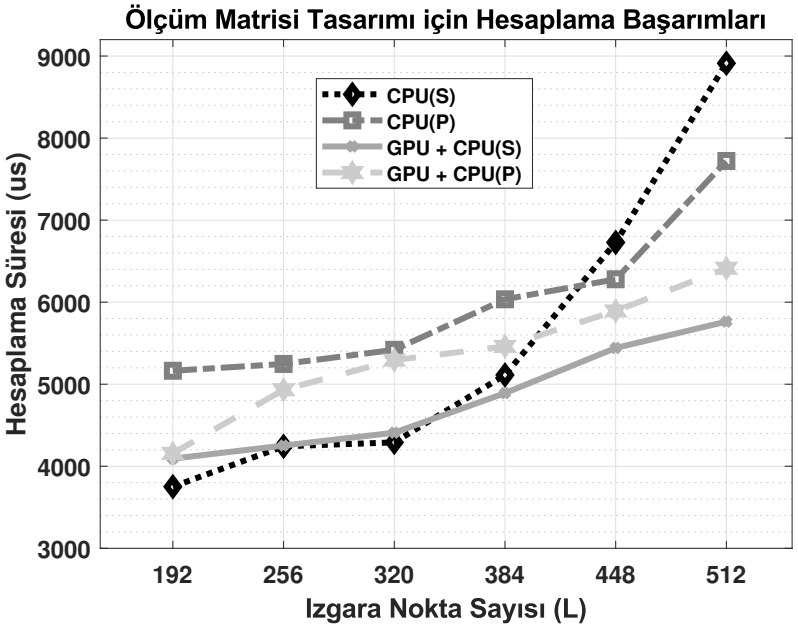
Şekil 6.2: Ölçüm Matrisi Tasarımı için Heterojen Hesaplama Mimarisi

Izgara Nokta Sayısı	CPU (S)	CPU (P)	CPU(S) + GPU	CPU(S)- GPU Transfer	CPU(P) + GPU	CPU(P) - GPU Transfer
192	3751	5164	3782	311	3819	338
256	4243	5247	3928	324	4579	354
320	4291	5419	4074	335	4931	362
384	5112	6035	4506	385	5042	419
448	6728	6279	5017	425	5437	456
512	8912	7721	5312	451	5931	482

Çizelge 6.5: Ölçüm Matrisi Tasarımı için CPU ve CPU-GPU Başarım Ölçümleri (us)
Tek Çekirdek (S), Çoklu Çekirdek (P)

Çizelge 6.5'te gösterildiği üzere CPU-GPU transferi dikkate alındığında hesaplama süresinin ortalama %10'u kadar ek yük toplam çalışma süresine etki etmektedir. Dikgen Model Matris ile Tersinir Devrik Matris hesaplama öncesinde GPU belleğinde kaydedildiği için veri transfer süresi hesabına katılmamaktadır. Çizelge 6.5'te görüldüğü üzere $L = 192$ değerinde CPU tek çekirdek başarımı diğer seçeneklere göre daha iyi sonuç vermektedir. Izgara nokta sayısı arttıkça, matris çarpım işlemlerinde CPU koşut işlemeye göre GPU başarımının yüksek başarım göstermesi ve SVD işlemlerindeki performans hızı yavaşlama makasının kapanması etkisiyle GPU, hesaplama başarımı açısından avantajlı hale gelmektedir.

Şekil 6.3'teki başarım ölçümlerinde Intel Xeon E5-2637 CPU ve NVIDIA GTX 1080 GPU kullanılmaktadır. Izgara nokta sayısı $L \geq 384$, olduğu tasarımlarda heterojen mimari tabanlı GPU hesaplama başarımı tüm CPU hesaplama başarımlarından daha yüksek çıkmaktadır. Çizelge 6.4'teki ölçümler ile örtüşecek şekilde, CPU çok çekirdekli hesaplama CPU tek çekirdek hesaplamaya göre $L \geq 448$ olduğu durumlarda yüksek başarım sunmaktadır. Yapılan paralel hesaplamalar ile hesaplama süresi başarımı seri hesaplamaya göre $L = 192$ için yaklaşık 17x hızlanmaktadır. En yüksek hızlanma CPU tek çekirdek kullanımı ile alınmaktadır. CPU-GPU tabanlı heterojen mimaride veri transferinden ötürü CPU'ya göre düşük performans göstermektedir.



Şekil 6.3: Ölçüm Matrisi Tasarımı Hesaplama Başarım Karşılaştırmaları

6.1.2. Konveks Optimizasyon tabanlı Açı Kestirimi

Bölüm 2.3.2.'de bahsedildiği üzere konveks optimizasyon çözümü olan LASSO yaklaşımı ile geliş açısı kestirimi optimizasyon probleminin çözümü gerçeklenerek hesaplanmaktadır. Optimizasyon modelinin daha hızlı yakınsamasını sağlamak için ADMM yöntemi uygulanarak geliş açısı kestirimi yapılmaktadır. Algoritma 4'te sıkıştırılmış algılama tabanlı geliş açısı kestirimi anlatılmaktadır.

Algoritma 4 ADMM tabanlı Geliş Açısı Kestirimi Algoritma Adımları [20]

Girdiler : L : Izgara Nokta Sayısı, M : Sensör Dizi Uzunluğu, K : Hedef Sayısı
m : Sıkıştırılmış Sensör Kanal Sayısı, Φ : Ölçüm Matrisi, mxM
$\boldsymbol{D}$:Sözlük Matrisi, $\boldsymbol{MxL}$, $\boldsymbol{s_G}$: Sinyal Vektörü, $\boldsymbol{Mx1}$
N : ADMM Iterasyon Sayısı, $\boldsymbol{\mu}$ ve $\boldsymbol{\mu_2}$: Iterasyon adım boyutları
Çıktı : $\boldsymbol{s_{Go}}$: Yakınsayan kestirim vektörü,
Algoritma
Adım-1 : Sözlük Matrisinin Güncellenmesi : $\boldsymbol{G = (\Phi \cdot D)/\mu_2}$, $\boldsymbol{mxL}$,
Adım-2 : Sinyal Vektörünün Güncellenmesi : $\boldsymbol{s_{G_{upd}} : (\Phi \cdot s_G)/\mu_2}$, $\boldsymbol{mx1}$,
Adım-3 : G Matrisinin Tersi Alma İşlemi : $\boldsymbol{M = (I + G^H \cdot G)^{-1}}$, $\boldsymbol{LxL}$,
Adım-4 : Lagrange Toplayıcı ve Çarpanların Oluşturulması,
for i=1:N **do**
Adım-5 : $\boldsymbol{r}$ = Lagrange Toplam Vektörü + $\boldsymbol{G'}$ (Lagrange Çarpım Vektörü), $\boldsymbol{Lx1}$,
Adım-6 : $\boldsymbol{s_{Go} = Mr}$ Çıktı Vektör, $\boldsymbol{Lx1}$,
Adım-7 : Lagrange Güncelleme Vektörünün Oluşturulması, $\boldsymbol{A_x = Gs_{Go}}$, $\boldsymbol{mx1}$,
Adım-8 : Lagrange Toplayıcı ve Çarpanların Güncelleme İşlemleri
end for
Adım-9 : $\boldsymbol{s_{Go}}$ üzerinde $\boldsymbol{K}$ tane maksimum açı değerlerinin bulunması

Algoritma 4'teki sıralı işlem akışında Adım-1 ve Adım-3'teki işlemlerin sadece ölçüm matrisi tasarımı güncellenirken bir kez yapılması yeterli olduğundan bu iki adım gerçek zamanlı hesaplama başarımına dahil edilmemektedir. Adım-6'da iterasyon sayısı kadar döngü sonucunda kestirim vektörü hesaplanmış olup Adım-9'da hedef sayısı kadar açı noktaları tespit edilmektedir. Algoritma 4'teki alt adımlar maddeler halinde detaylı olarak incelenmektedir:

- **Adım-1** : Sözlük matrisi ile inşa edilen iki boyutlu kare bir matris ölçüm matrisi ile güncellenerek sıkıştırılmış sensör kanalı kadar işlem yapılmasına olanak sağlanır.

- **Adım-2** : Hedeflere ait M tane sensör dizisinden alınan sinyal ölçümleri ölçüm matrisi ile güncellenerek sıkıştırılmış ölçüm vektörüne dönüştürülür.
- **Adım-3** : Bu adımda güncellenmiş sözlük matrisinin tersi alma işlemi gerçekleşmektedir. Bu işlemin her hedef sinyalinin geliş açısı kestiriminde yapılmasına gerek yoktur.
- **Adım-4** : ADMM yöntemi ile optimizasyon çözümü, Lagrange çarpanlar ve toplayıcılar ile hızlı yakınsamaktadır. Çarpan ve toplayıcıların tanımı bu adımda gerçekleşir.
- **Adım-5** : Matris-vektör çarpım işlemi sonucunda hesaplanan vektör Lagrange toplayıcı ile toplandıktan sonra kestirim vektörünün hesabında kullanılmaktadır.
- **Adım-6** : Hesaplanan kestirim vektörü N iterasyon sonunda güncellenerek Adım-9'da maksimum noktaları bulma işleminde girdi olarak kullanılır.
- **Adım-7** : Lagrange güncelleme vektörü kestirim vektörünün sözlük matrisi ile çarpımından elde edilir. Bu vektör Lagrange bileşenlerinin güncellenmesinde kullanılır.
- **Adım-8** : Lagrange toplayıcı ve çarpan vektörleri güncellenmektedir.
- **Adım-9** : Global maksimum noktalar bulunarak geliş açıları hesaplanır.

Adım-3'teki ters alma işlemi için Eşitlik 11'de gösterildiği üzere Woodbury Matris Identiy (WMI) dönüşümü [128] ile hesaplama gerçekleştirilir. Dönüşüm ile mxL boyutunda bir matris için L uzunluk yerine m uzunluk kare matris tersi alma işlemi yapılmaktadır.

$$(I + G^H G)^{-1} = I - G^H (I + GG^H)^{-1} G \tag{11}$$

6.2. Hesaplama Analizi

Bu bölümde, sıkıştırılmış algılama tabanlı geliş açısı kestiriminde Algoritma 4'teki işlemler için hem bellek kaynak kullanım hem de hesaplama karmaşıklık analizi yapılmaktadır. Paralel programlama için darboğaz işlem adımları belirlenip uygun hızlandırma çözümleri çalışılmaktadır. Çizelge 6.6'da Algoritma 4'teki her bir adım için bellek kaynak kullanım analizi

yapılmaktadır. Çizelge 6.6'daki kaynak kullanım analizi ve Çizelge 6.1'deki tasarım parametrelerine göre toplamda 288192+K tane örnek için bellek hücresi kullanılmaktadır. Veri tipinin 32-bit kayar nokta olduğu ve hedef sayısının önemsenmediği varsayımında yaklaşık 1.1 MByte bellek alanına ihtiyaç duyulmaktadır. Adım-3'te WMI dönüşümü uygulandığında Adım-3 için kaynak kullanım ihtiyacı $4L^2+2mL+6m^2$ olup gereksinim %30 azalmaktadır.

Algoritma Adımları	**Bellek Kullanımı**
Adım-1	$2(mM+mL+ML)$
Adım-2	$2(M+m)$
Adım-3	$2mL+6L^2$
Adım-4	$2(m+L)$
Adım-5	$4L$
Adım-6	$2L$
Adım-7	$2m$
Adım-8	-
Adım-9	K
Toplam	$=6L^2+2ML+4mL+2mM+6L+2M+4m+K$

Çizelge 6.6: Sıkıştırılmış Algılama tabanlı Geliş Açısı Kestirimi - Bellek Gereksinimi

6.2.1. Hesaplama Karmaşıklığı

Çizelge 6.7'de ilgili adımlar için hesaplama karmaşıklığı analizi sunulmaktadır. Çizelge 6.7'deki sonuç, Çizelge 6.1'deki parametreler ve Bölüm 2.3.2.'deki iterasyon analizi sonucuna göre 64 iterasyon için hesaplama ihtiyacı $2.95x10^6$ MAC işlem sayısı olmaktadır.

Algoritma Adımları	**İşlem Sayısı (MAC)**
Adım-2	mM
Adım-5	NmL
Adım-6	NL^2
Adım-7	NmL
Adım-8	NmL
Toplam	$N(3mL+L^2)+mM$

Çizelge 6.7: Sıkıştırılmış Algılama tabanlı Geliş Açısı Kestirimi - Hesaplama Karmaşıklığı

Gauss-Jordan azaltma yöntemine göre Adım-3'teki matris ters alma işleminin hesaplama karmaşıklığı $O(N^3)$ olmaktadır [129]. Herhangi bir dönüşüm yapılmadığı durumda $L^3 + mL^2$ kadar MAC işlem sayısı yapılırken WMI uygulandığında $L^2m + 2m^2L + m^3$ MAC işlem yapılmaktadır. $m = 16$ ve $L = 192$ parametreleri için WMI sayesinde yaklaşık 11x işlem sayısı iyileştirmesi elde edilmiştir.

6.2.2. Darboğaz İşlem Haritası ve Değerlendirmeler

Geliş açısı kestirimi işleminde iterasyonlar öncesi Adım-2 haricindeki işlemler hesaplama sırasında yapılmayacağı için darboğaz işlem analizinde değerlendirilmemektedir. Diğer işlemler matris-vektör çarpımı ve toplam işlemleri olduğu için baskın darboğaz oluşturacak bir işlem adımı görülmemektedir. Sadece iterasyon sayısına bağlı olarak Adım-6'daki matris-vektör çarpım işlemindeki boyutlardan ötürü en fazla süre olan kısım Adım-6'daki adım olmaktadır. Herhangi bir hızlandırma yapılmadığında geliş açısı kestirim hesaplaması yaklaşık 4.6 ms sürmektedir. Hızlandırma işlemleri için hem CPU'da hem de CPU-GPU heterojen mimaride tasarımlar gerçekleştirilerek başarım artışının sağlanması hedeflenmektedir.

6.3. CPU Koşut İşleme

Koşut işleme tasarımında matris-vektör işlemlerinde Intel MKL tabanlı iyileştirmeler yapılmaktadır. OpenMP-MKL tabanlı matris-vektör çarpımlar ile hızlandırma yapılmaktadır. Çizelge 6.8'de koşut işleme için kullanılan CPU hesaplama birimleri belirtilmektedir. Çizelge 6.8'deki güç tüketimi değerlerinde her bir işlemci için maksimum değer alınmıştır.

CPU Birimleri	Frekans (GHz)	Güç Tüketimi (Watt)	Çekirdek Sayısı (#)
Intel Xeon E5-2637	3.5	130	4
Intel Xeon E3-1535M	2.9	45	4
Intel i9-9900	3.1	65	8
Intel i7-9750	2.6	45	6

Çizelge 6.8: Geliş Açısı Kestirimi için CPU Koşut Hesaplama Birimleri ve Özellikleri

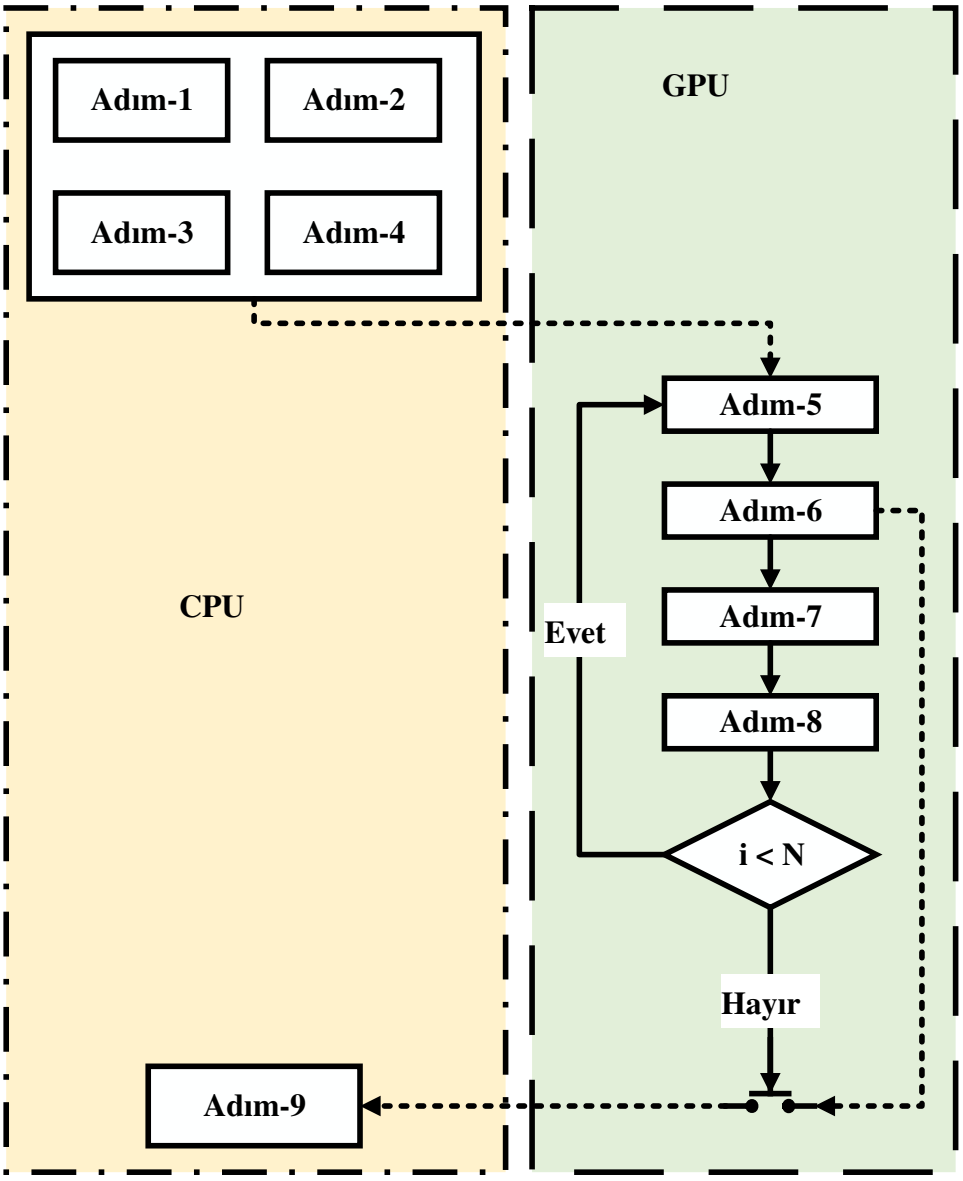
Çizelge 6.8'deki hesaplama birimleri ile çeşitli ADMM iterasyon (yineleme) sayıları için karşılaştırmalı hesaplama başarımları Çizelge 6.9'da gösterilmektedir. Her bir hesaplama için tek çekirdek ile yapılan koşut işleme ve çoklu çekirdekli yapıda OpenMP ile yapılan hesaplama süreleri verilmektedir. 64 adet ADMM iterasyon (yineleme) durumunda Intel i9-9900 CPU çok çekirdekli kullanımda 584 us ile en hızlı hesaplama yapılmaktadır. Matris-vektör çarpımı için Intel MKL çoklu çekirdekli işlevleri kullanılmaktadır. Algoritma 4'teki Adım-8'de Lagrange bileşenleri hesaplanırken norm alma işlemi gerçekleştirilmektedir. Norm alma işlemi için çekirdek hesaplama optimizasyonları ve OpenMP iyileştirmeleri uygulanır. Intel Xeon E5-2637 ile çoklu çekirdekli yapıda alınan ölçümlerde OpenMP kullanımının hesaplama süresini artırdığı gözlemlenmiştir.

Yineleme Sayıları & Birimler	1	2	4	8	16	32	**64**	128
Intel Xeon E5-2637 CPU (S)	23	45	75	148	291	577	**1094**	2143
Intel Xeon E5-2637 CPU (P)	24	39	81	151	294	715	**1324**	2394
Intel Xeon E3-1535 CPU (S)	22	40	72	132	233	450	**790**	1560
Intel Xeon E3-1535 CPU (P)	19	30	60	110	154	301	**642**	1282
Intel i9-9900 CPU (S)	27	38	67	120	190	349	**693**	1420
Intel i9-9900 CPU (P)	25	35	57	103	164	285	**584**	1147
Intel i7-9750 CPU (S)	21	36	66	140	212	480	**835**	1635
Intel i7-9750 CPU (P)	18	31	55	134	195	456	**803**	1495

Çizelge 6.9: Sıkıştırılmış Algılama tabanlı Geliş Açısı Kestirimi için CPU Koşut Hesaplama Başarım Karşılaştırması (us) (S): Tek Çekirdek, (P) : Çoklu Çekirdek

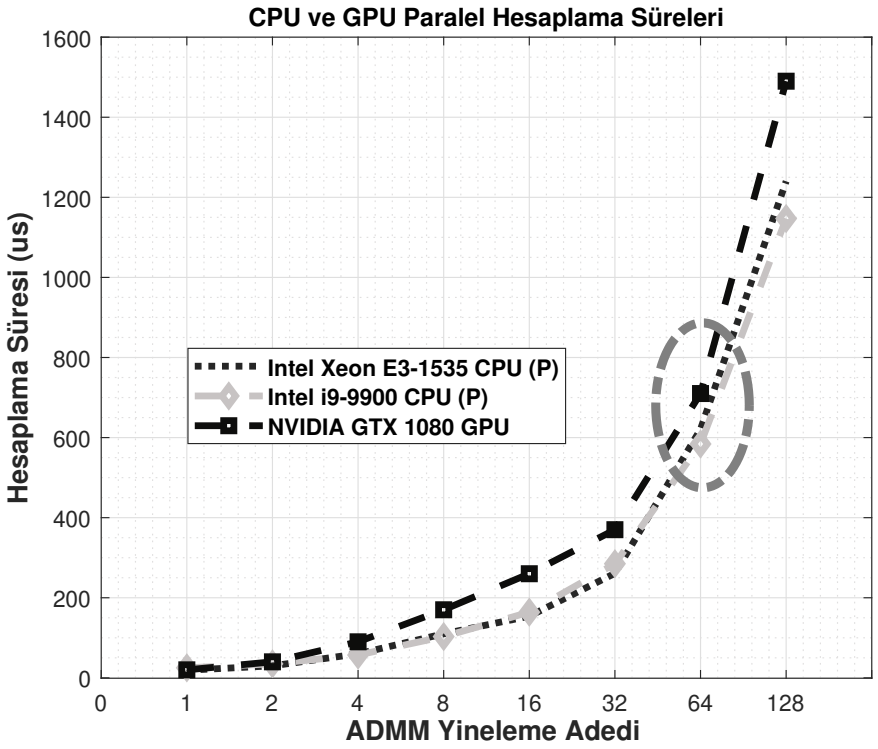
6.4. GPU ile Hızlandırma

Bu bölümde Algoritma 4'teki adımların Şekil 6.4'te belirtilen CPU-GPU heterojen mimari tabanlı paralel hesaplamalar için başarımları analiz edilmektedir. Şekil 6.4'te GPU'da yapılan Adım-5'ten Adım-8'e kadarki işlemler ADMM iterasyon sayısı (N) kadar tekrarlanır. Bu durum CPU'dan kernel çağırmasını gerektirdiği için hesaplama başarımına kernel çağırma ek yükü getirmektedir.



Şekil 6.4: Sıkıştırılmış Algılama tabanlı Geliş Açısı Kestirimi Modeli için CPU-GPU Heterojen Hesaplama Mimarisi

Adım-1'den Adım-4'e kadar işlemler her hesaplama işlevinde bir kez yapıldığından CPU'da gerçekleşmektedir. Güncellenen sinyal ölçüm vektörü Adım-5 öncesinde GPU'ya transfer edilmektedir. ADMM iterasyon sayısı kadar hesaplama sonunda Adım-6 sonucu CPU'ya gönderilmektedir. Adım-9 ile geliş açısı kestirimi tamamlanmaktadır. Şekil 6.5'te GPU biriminin başarımı ile en başarımlı CPU hesaplamaları karşılaştırılmaktadır. GPU birimi olarak Çizelge 4.6'da özellikleri belirtilen NVIDIA GTX 1080 GPU kullanılmaktadır.



Şekil 6.5: Sıkıştırılmış Algılama tabanlı Geliş Açısı Kestirimi için
CPU ve GPU Koşut Hesaplama Başarım Karşılaştırması, (P) : Çoklu Çekirdek

Şekil 6.5'te gözlemlenen sonuçlara göre GPU başarımı CPU başarımına göre düşük çıkmaktadır. 64 iterasyon için GTX 1080 ile **710 us** işlem süresi (veri transferi dahil) içinde geliş açısı kestirimi yapılmaktadır. CPU başarımı 64 iterasyon için GPU'ya göre **1.2x** hızlı çalışmaktadır. Geliş açısı kestirimi işleminin yineleme tabanlı bir hesaplama içermesinden dolayı çok sayıda tekrarlı kernel çağrılması nedeniyle ile GPU başarımı olumsuz etkilenmektedir. CPU'dan çağrılan kernel işlevleri her bir yinelemede NVIDIA GTX 1080 GPU'da ortalama 2 us hesaplama yüküne neden olmaktadır. Her bir yinelemede, kernel çağrılmasında cuBLAS [125] tabanlı işlemler nedeniyle global bellekten veri çekilmesi nedeniyle başarım düşmektedir. Buna karşın GPU başarımı için yapılan kritik iyileştirmeler sırasıyla:

- **Paylaşımlı bellek optimizasyonu** : Adım-8'deki güncellemede norm alma işlemi cuBLAS [125] ile hesaplanırken sonuç CPU belleğine aktarılır. Bu bilgiyi GPU'ya transfer ederek güncelleme devam ettirilmelidir. Veri transferi nedeniyle düşen hesaplama

başarımı işlem girdisinin GPU global bellekten paylaşımlı belleğe alınması ile **6x** hızlandırılmıştır.

- **GPU iç işlev (intrinsic) fonksiyon optimizasyonu** : Adım-8'de gerçekleşen güncelleme işlemlerinde karakök alma, toplama ve çıkarma fonksiyonları kullanılmaktadır. Bu temel işlemler için iç işlev fonksiyonlar kullanılarak yazmaç belleğin etkin kullanılması sağlanarak **1.1x** hızlanma gerçekleştirilmiştir.

Sonuç olarak geliş açısı kestiriminde gerekli doğruluk başarımını sunan 64 ADMM iterasyonu için GPU ile yapılan paralel hesaplama CPU seri işlemeye (4.6 ms) göre yaklaşık **6.5x** hızlanma sağlamıştır. CPU ile yapılan paralel hesaplamalar sayesinde CPU seri işlemeye göre geliş açısı kestirimi **7.9x** hızlandırılmıştır. Ayrıca Adım-3'teki matris ters alma işleminde WMI dönüşümü ile ters alma işlemi GPU paralel hesaplamada ortalama 40x hızlanmıştır.

6.5. Güç Verimli GPU Hesaplama

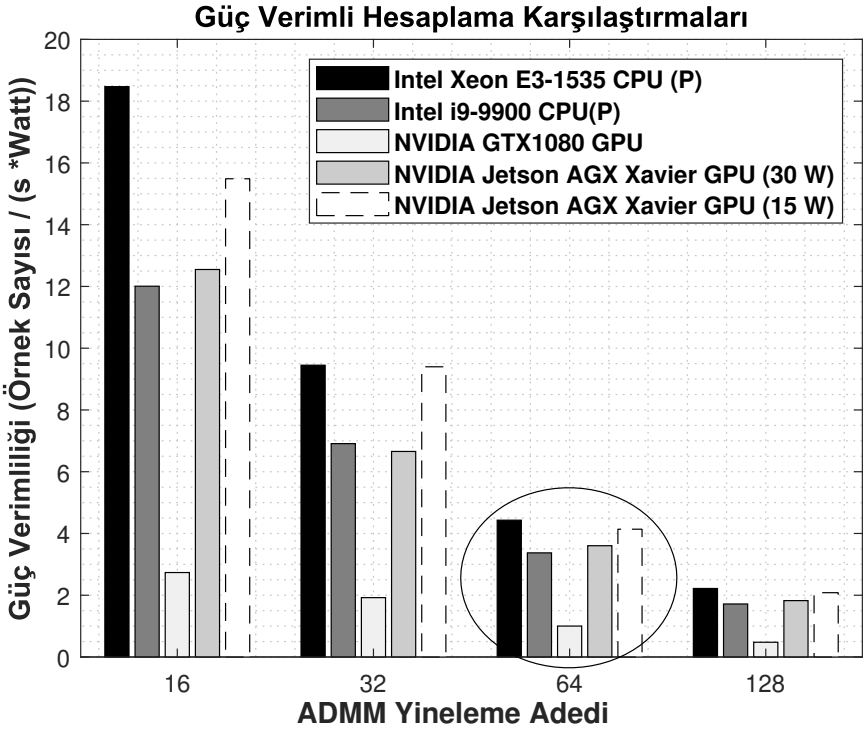
Güç tüketimi avantajı sunan grafik işlemcilerden biri olan Jetson AGX Xavier GPU ile Çizelge 6.10'da gösterildiği üzere hesaplama başarımları incelenmektedir. Çizelge 6.10'a göre 64 ADMM yinelemesi için 30 W güç tüketiminde CPU seri işlemeye (4.6 ms) göre **3.9x** hızlanma olurken, 15 W güç tüketimi ile **2.2x** hızlanma sağlanmıştır.

ADMM Yineleme Sayısı	**1**	**2**	**4**	**8**	**16**	**32**	**64**	**128**
Jetson AGX Xavier 30 Watt	43	91	162	247	340	641	1184	2337
Jetson AGX Xavier 15 Watt	82	115	194	365	551	908	2062	4098

Çizelge 6.10: Geliş Açısı Kestirimi için Güç Verimli Hesaplama Başarımları (us)

Şekil 6.6'da 64 yineleme için GPU tabanlı uygulamalarda Jetson AGX Xavier GPU (15 W) güç tüketiminde en çok verim sağlamaktadır. Düşük yineleme sayılarında Intel Xeon E3-1535 CPU en verimli sonuç üretmektedir. Jetson GPU'larda entegre ve gömülü bir mimari içermesi açısından kernel çağırma süresi (ortalama 1 us (30 Watt) ve 1.5 us (15 Watt))

yükünün GTX 1080 GPU'ya göre daha az olması verimlilik karşılaştırmasında ortaya çıkmaktadır. Veri transferi Jetson AGX Xavier GPU'da 30 Watt güç tüketiminde ortalama 20 us sürerken, 15 Watt güç tüketiminde ortalama 37 us sürmektedir. GTX 1080 GPU ile veri transferi 15 us sürmektedir. 64 adet yineleme için kernel çağırma ek yükleri toplam transfer süresi GTX 1080 GPU'da ortalama 143 us sürerken, Jetson AGX Xavier GPU 30 Watt güç tüketiminde ortalama 84 us ve 15 Watt güç tüketimi ile ortalama 133 us sürmektedir.



Şekil 6.6: Geliş Açısı Kestirimi için Güç Verimlilik Karşılaştırmaları

6.6. Tartışma

Sıkıştırılmış algılama tabanlı geliş açısı kestirimi iki temel başlıkta incelenmiştir. İlki ölçüm matrisi tasarımı uygulaması, ikincisi konveks optimizasyon tabanlı geliş açısı kestirimidir. Ölçüm matrisi tasarımı sensör mimarisinde anlık olarak hesaplanmasa bile kalibrasyon ve güncellemeler sırasında hızlı yapılması için hesaplama analizleri ve paralel hesaplamalar gerçekleştirilmiştir. Ölçüm matrisi tasarımında darboğaz analizi sonucunda SVD işlemi hesaplama süresini önemli ölçüde etkilemiştir. SVD işlemi için gerçekleştirilen analizlerde

CPU başarımı GPU'ya göre yüksek çıkmıştır. Bu sonuç aynı zamanda ölçüm matrisi tasarımının hesaplama süresini de etkilemiştir. 192 adet ızgara nokta sayısı durumunda CPU paralel hesaplama ile CPU-GPU heterojen hesaplamaya göre **1.1x** hızlanma sağlamıştır. Izgara nokta sayısı arttıkça CPU paralel hesaplama heterojen mimariye göre düşük başarım göstermektedir. 512 ızgara nokta sayısı durumunda CPU-GPU tabanlı hesaplama CPU paralel hesaplamaya göre **1.5x** hızlandırma sağlamıştır. Ölçüm matrisi tasarımının CPU paralel hesaplama ile yapılması hesaplama başarımı açısından avantajlı olmaktadır.

Konveks optimizasyon tabanlı geliş açısı kestiriminde daha hızlı yakınsama için ADMM yöntemi kullanılmaktadır. Bölüm 2.3.2.'de yapılan analizlere göre 64 adet ADMM iterasyonu güvenilir yakınsama için yeterli olmaktadır. ADMM tabanlı geliş açısı kestiriminin gerçek zamanlı sistemlerde kullanımı için paralel hesaplamalar ile hızlandırılmalıdır. CPU seri işleme ile alınan başarım işlemci kaynak kullanımları ile arttırılabilmektedir. Kestirim işleminin algılayıcıda son işlemlerden biri olması nedeniyle, iyileştirmeler toplam veri işleme süresini azaltmaktadır. Başarım karşılaştırmaları sonucunda 64 iterasyon için Intel i9-9900 CPU ile en yüksek başarım sağlanmıştır. CPU başarımının GPU başarımına göre daha yüksek çıkmasındaki en önemli nedenler, GPU'da her bir iterasyonun oluşturduğu transfer yükü ve CPU'da iterasyon güncellemelerinde önbellek kullanımının artmasıdır. Küçük girdi boyutunda ADMM ile gerçekleşen çözümde GPU hesaplamanın CPU'ya göre avantaj sağlamadığı literatürdeki bir çalışmanın sonucunda da görülmektedir [130]. Scgubiger ve diğerlerinin çalışmasında, ancak problem girdi boyutu arttıkça GPU başarımı CPU'ya göre avantaj sağlamıştır.

Hesaplama başarımının yanı sıra enerji-güç verimliliği mobil ve uç hesaplama işlevine sahip sensör sistemlerinde kritik önemde olmaktadır. Düşük güç tüketen işlemciler tercih edilerek başarım hedeflenmektedir. Jetson AGX Xavier GPU (15 Watt) ile **2 ms**'de geliş açısı kestirim işlemi tamamlanır. Jetson AGX Xavier GPU mimarileri güç verimli hesaplama imkanı sunsa bile sıkıştırılmış algılama tabanlı geliş açısı kestirimi uygulamasında en çok güç verimini Şekil 6.6'daki sonuçlara göre Intel Xeon E3-1535 CPU sağlamıştır. Yineleme sayısı arttıkça Jetson AGX Xavier GPU mimarisi de yakın oranda güç verimliliği sağlamaktadır.

İterasyon tabanlı çözümlerde GPU'da her bir döngüde CPU kernel çağırmadan dolayı GPU

hesaplama başarımı azalmaktadır. Ayrıca algoritmada birbirini bekleyen adımlar olduğundan GPU paralel hesaplamadan istenilen ölçüde faydalanamamıştır. CPU başarımlarındaki olumlu temel etki ise ardışık işlemler sayesinde önbellek yerelliğinin sağlanabilmesidir. Önbellek kullanımı arttıkça özellikle yüksek ADMM iterasyonlarında başarım artmaktadır.
Geliş açısı kestirimi için geleneksel yöntemlere göre daha yüksek kestirimi doğruluğu sunan konveks optimizasyon tabanlı yaklaşım hesaplama karmaşıklığını artırmaktadır [20]. Bu bölümde yapılan çalışmalar ile artan hesaplama karmaşıklığının işlem süresine etkisi paralel hesaplamalar ile azaltılmıştır. Gelecek çalışmalarda GPU paralel hesaplamaya uygun bir yapıda algoritma hesaplama tasarımı gerçekleştirmek planlanmaktadır.

7. GERÇEK ZAMANLI UYGULAMA ENTEGRASYONLARI

Bu bölümde, tez çalışmasında incelenen uygulamalar olan Kısa Zamanlı Fourier Dönüşüm (STFT) tabanlı Sayısal Alt Çevrimi, Makine Öğrenmesi tabanlı Hedef Tespiti ve Sıkıştırılmış Algılama tabanlı Geliş Açısı Kestirimi için hesaplama başarımlarına ait entegrasyonların gerçek zamanlı kriter için davranışları anlatılmaktadır. Darbe-Doppler radar ve elektronik harp mimarilerinin özellikleri tanımlanarak uygulama senaryoları için konsept tasarım önerileri verilmektedir. Darbe-Doppler radar senaryosunda Makine Öğrenmesi tabanlı Hedef Tespiti ve Sıkıştırılmış Algılama tabanlı Geliş Açısı Kestirimi uygulamalarının entegrasyonu gösterilirken, elektronik harp tabanlı yazılım tanımlı telsiz senaryosunda Sayısal Alt Çevrimi ve Makine Öğrenmesi tabanlı Hedef Tespiti uygulamalarının entegrasyonu yapılmaktadır. Güç verimli hesaplamalara gereksinim duyan mobil ve uç hesaplama sensör mimarileri için entegrasyon tasarımları sunulmaktadır. Entegrasyonlar için gerçek zamanlı işlem kriterleri belirlenerek başarımlar incelenmektedir.

7.1. Sensör Mimarileri ve Entegrasyonlar

Entegrasyon tasarımında hava platformu tabanlı darbe-Doppler radar tipleri olarak ele alınırken, yazılım tanımlı telsiz tabanlı elektronik harp sistemleri de incelenmektedir.

Radar Sensörleri : Radarlar yenilikçi alanlar olan görüntüleme ve sağlık alanında [131, 132] kullanılmasının haricinde insansız hava aracı, uçak ve helikopter gibi platformlarda yer hareketli hedef göstergesi (GMTI) amaçlı da kullanılmaktadır [133]. Hava platformu tabanlı (airborne) radar sistemlerinde boyut, ağırlık, güç tüketimi ve maliyet (SWaP-C) açısından etkin bir hesaplama modeli ile platformlardaki verimlilik arttırılabilmektedir. Düşük güç tüketimi ile daha uzun uçuş görevi sağlanabilmektedir.

Hedef tespitinde kullanılan 3000 adet menzil ve 16 adet Doppler hız hücre sayısı parametreleri ile radar özellikleri belirlenerek hüzme süresi içerisinde GMTI problemi için gerçek zamanlı entegrasyon amaçlanmaktadır. Darbe-Doppler radarlar Darbe Tekrarlama Frekansı (DTF) çeşitlerine göre düşük, orta ve yüksek DTF şeklinde ayrılmaktadır [134]:

Düşük DTF (< 3 KHz) : GMTI probleminde hız ölçümünün başarımı azalmaktadır.

Orta DTF ($3 - 30$ KHz) : Hava radarlarında GMTI tespiti için kullanılabilmektedir.

Yüksek DTF (> 30 KHz) : İncelenen hız aralığı arttığı için takip modunda kullanılmaktadır.

Tez çalışmasında orta DTF modu için senaryolar belirlenerek gerçek zamanlı işleme kriterleri tanımlanmaktadır. Kriter belirlemede ihtiyaç duyulan radar denklemleri yazılarak menzil çözünürlüğü (ΔR), Doppler hız çözünürlüğü (ΔV), darbe sayısı (N) ve radar hüzme süresi (Coherent Processing Interval, CPI) için eşitlikler Çizelge 7.1'de belirtilmektedir. Çizelge 7.1'deki Hüzme Süresi gerçek zamanlı işleme için başarım kriteridir. Işık hızı değeri $c = 3 \times 10^8$ m/s değerinde olmaktadır. Dalga boyu (λ) ise radar çalışma frekansından bulunmaktadır. Merkez frekans değeri 10 GHz olarak alınırken dalga boyu 3 cm olmaktadır.

No	Değer	Birim	Sembol	Eşitlik
1	Menzil Çözünürlüğü	m	ΔR	$\Delta R = \frac{c}{2B}$
2	Darbe Tekrarlama Frekansı	Hz	DTF	$DTF = \frac{2V_{max}}{\lambda}$
3	Darbe Tekrarlama Aralığı	s	DTA	$DTA = \frac{1}{DTF}$
4	Hüzme Süresi	s	CPI	$CPI = N \cdot DTA$
5	Doppler Hızı Çözünürlüğü	m/s	ΔV	$\Delta V = \frac{\lambda}{2 \cdot CPI}$
6	Maksimum Belirsiz Menzil	m	R_{max}	$R_{max} = \frac{c}{2 \cdot DTF}$

Çizelge 7.1: Radar Zaman Bütçesi Tasarımı için Eşitlik Tanımları

Orta DTF değerlerininin değişkenliği ile Çizelge 7.1'deki değerler hesaplanmaktadır. Değişken DTF değerleri bant genişliğinin farklılaşmasıyla elde edilmektedir. Hesaplama sonucu oluşan parametre seti Çizelge 7.2'de gösterilmektedir. Çizelge 7.2'deki tasarımların bir hava platformu radarındaki tespit ve geliş açısı kestirimi için gereken gerçek zamanlılık kriterleri olmaktadır. Açısal tarama mekanizması hüzme süresi hesabında [134] dikkate alınmamıştır.

Bant Genişliği (B)	**ΔR (m)**	**R_{max} (km)**	**DTF (KHz)**	**DTA (us)**	**DeltaV (m/s)**	**V_{max} (m/s)**	**CPI (us)**
B = 10 MHz	15	45	***3.34***	300	3.13	50	**4800**
B = 15 MHz	10	30	***5***	200	4.69	75	**3200**
B = 20 MHz	7.5	22.5	***6.67***	150	6.25	100	**2400**
B = 25 MHz	6	18	***8.34***	120	7.81	125	**1920**

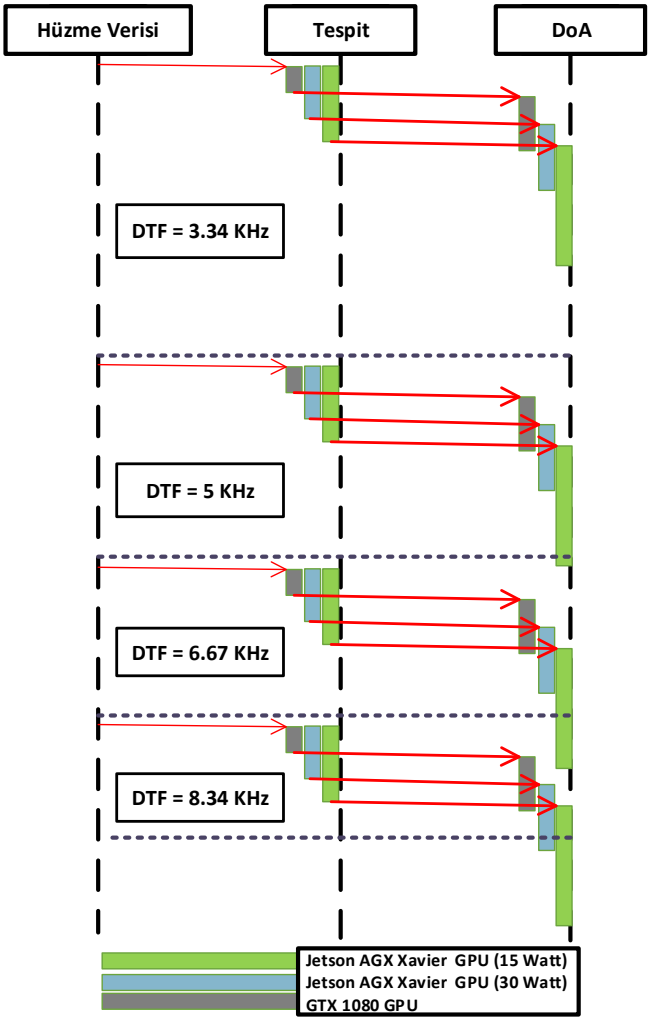
Çizelge 7.2: Çeşitli Darbe-Doppler Radar Tasarımları ve Hüzme Süresi Hesaplamaları

Her bir senaryo adımı için güç verimli hesaplama ve GPU hesaplama entegrasyonları çalışılmaktadır. Senaryolardaki uygulamalara ait başarım ölçümleri Çizelge 7.3'te verilmektedir.

Hesaplama Birimleri & Hesaplama Süreleri (us)		**Makine Öğrenmesi tabanlı Hedef Tespiti**	**Sıkıştırılmış Algılama tabanlı Geliş Açısı Kestirimi**
Jetson AGX Xavier GPU	15 Watt	1246	2062
	30 Watt	875	1184
GTX 1080 GPU		215	710

Çizelge 7.3: Gelişmiş Radar Sinyal İşleme Uygulamaları'nın GPU Başarımları

Çizelge 7.3'teki sonuçlara göre 4 adet DTF senaryousu için gerçek zamanlı başarım kriteri sağlayan entegrasyon akışlarının davranışları Şekil 7.1'de gösterilmektedir.



Şekil 7.1: DTF Senaryoları için Uygulama Entegrasyonların Kriter Başarımları

Gösterim sonuçlarına göre 3.34 KHz DTF değerinde maksimum 180 km/h hedef hız tespiti ve maksimum 45 km menzilde gerçek zamanlı hareketli hedef tespiti ve geliş açısı kestirimi (DoA) Jetson AGX Xavier 15 Watt modunda güç verimli şekilde yapılabilmektedir. DTF değeri arttıkça sadece 30 Watt modu ile güç verimli hesaplama tabanlı gerçek zamanlı entegrasyon mümkün olabilmektedir. Radar sensör mimarisinde örnekleme için kullanılan ADC içeren FPGA yapısında sayısal alt çevrimi, filtrelemeler ve Doppler işleminin yapıldığı varsayılarak son işleme senaryoları entegre edilmiştir.

Yazılım Tanımlı Telsiz Sensörleri : Amatör radyo topluluklarının ilgilendiği yazılım tanımlı telsiz uygulamaları büyük organizasyonlar ile desteklenmektedir [135]. Yazılım tanımlı telsiz uygulamaları aynı zamanda elektronik harp sistemlerinde kullanıldığı için hedef yayınlara ait tespit ve teşhis işlemlerinde kritik rol oynamaktadır [136]. Tez çalışmasında incelenen sayısal alt çevriminde, bir yazılım tanımlı telsiz uygulaması modellenmiş olup geniş bant sinyal için 3200 kat indirgeme yapılarak dar bant sinyal elde edilmiştir.

Tez çalışması ile entegrasyon çalışmasının bütünlük arz etmesi açısından 3000 adet zaman hücresi ile 16 adet frekans penceresi kullanılarak spektrogram matrisi üzerinde hedef tespiti yapılmaktadır. Hedef tespiti yapılmadan önce 16 adet veri içeren vektör sayısal alt çevrimi sonucunda elde edilmektedir. 48000 adet veri içeren spektrogram matrisi 1500 milisaniye süresince oluşturulmaktadır. Elektronik harp mimarisinde ortamdaki sinyalleri tanımlamak için belirli bir boyutta veri kaydedilmektedir [137]. 1500 milisaniye analiz pencere boyutu ile dar bant sinyaller tespit edilebilmektedir.

Sayısal alt çevrim işlemi 100 ms'lik pencerelerde 15 adet adımla yapılarak 1500 milisaniyelik veri için 48000 uzunluğunda dar bant sinyal vektörü elde edilir. Çizelge 7.4'te 100 ms'lik pencereler için hesaplama birimlerinin başarımı verilmektedir.

Hesaplama Birimleri & Hesaplama Süreleri (ms)	**Sayısal Alt Çevrimi**		**Makine Öğrenmesi tabanlı Hedef Tespiti (1500 ms)**
	100 ms	**1500 ms**	
Jetson AGX Xavier GPU (15W)	31.892	478.38	1.246
Jetson AGX Xavier GPU (30W)	17.380	260.7	0.875
GTX 1080 GPU	8.425	126.375	0.215

Çizelge 7.4: Yazılım Tanımlı Telsiz Uygulamaları'nın GPU Başarımları

Çizelge 7.4'teki sonuçlara göre en düşük hesaplama hızı olan Jetson AGX Xavier GPU 15 Watt modunda yapılan hesaplama ile gerçek zamanlı kriterin %32'sinde sayısal alt çevrimi ve hedef tespiti işlemi bitirilmektedir. Tespit edilen her bir frekans aralığı ve zaman bölgesi için ayrıntılı yayın analizlerinin yapılabilmesi için %65'e yakın diğer işlemler için hesaplama süresi yüzdesi üst limiti sağlanmıştır.

7.2. Değerlendirme

Paralel programlama tabanlı yaklaşımlarla seri hesaplamalara göre önemli ölçüde hızlandırılan gelişmiş sinyal işleme uygulamalarının radar ve elektronik harp sensör mimarilerinde entegrasyonları gerçekleştirilmiştir. Entegrasyonlar sonucunda, deneysel hesaplamalar ile elde edilen hızlandırmaların gerçek zamanlı başarım kriterlerini sağlama davranışları gözlemlenmiştir. Ayrıca tez çalışmasının literatüre en önemli katkısı olan güç verimli heterojen paralel hesaplamalar ile belirli senaryolarda gerçek zamanlı sensör sinyal işleme olanağı sunulmuştur. Güç verimli hesaplamalar haricinde güçlü grafik işlemci mimarisinin gerçek zamanlı entegrasyonları incelenmiştir. Geliş açısı kestirimi uygulaması dışında grafik işlemci mimarilerinin CPU mimarilerine göre hesaplama başarımı her zaman yüksek çıkmıştır. Geliş açısı kestiriminin yer aldığı entegrasyon senaryosunda ölçüm matrisi tasarımının hesaplanması anlık olarak yapılmayacaktır. Bu yüzden gerçek zamanlı kriter analizinde ölçüm matrisi tasarımının hesaplama süresi dikkate alınmamıştır.

Güç verimli hesaplamalar ile gerçek zamanlı entegrasyonlar başarılarak düşük güç tüketimine ihtiyaç duyan sensör platformları için avantaj sunabilecek hesaplama tasarımlarının pratik uygulamalarda kullanımı önerilmiştir. Ağırlık ve güç tüketimi gibi kritik tasarım limitleri olan insansız hava araçları, uç hesaplama birimleri ve mobil sensör mimarileri için 15 Watt gibi düşük bir güç tüketimi ile gelişmiş sinyal işleme uygulamalarının gerçek zamanlı entegrasyonları sağlanarak sensör kabiliyetleri arttırılmıştır.

8. KAZANIMLAR, ZORLUKLAR VE GELECEK ÇALIŞMALAR

Gelişmiş sinyal işleme uygulamaları için güç verimli heterojen hesaplama çalışmasında düşük güç tüketen GPGPU mimarileri ile gerçek zamanlı hesaplama gereksinimine yönelik enerji verimli çözümler literatüre sunulmuştur.
Sayısal alt çevrimi için GPGPU'larda yüksek başarım sunabilen STFT tabanlı sayısal alt çevrimi hesaplaması ile çok kanallı geniş bant haberleşme ve alt örnekleme çözümlerinin gerçek zamanlı yapılabilmesi literatüre kazandırılmıştır. Derin öğrenme ve makine öğrenmesi tekniklerinin sınıflandırma problemlerinde sunmuş olduğu başarımı radar ve elektronik harp sensörlerinde uygulayan çalışmanın [1] gerçek zamanlı sensör mimarilerinde çalışabilmesi için paralel hesaplamalar gerçekleştirilmiştir. Sıkıştırılmış algılama ile sensör mimarisindeki filtre, yükseltgeç ve örnekleme birimlerinin sayısını azaltmaya yönelik yaklaşımın yer aldığı bir senaryoda geliş açısı kestirimi uygulamasında artan hesaplama karmaşıklığı yükü paralel hesaplamalar ile aşılarak gerçek zamanlı radar senaryosunda entegre edilmiştir.
Bu bölümde tez çalışmasının literatüre sunduğu kazanımlar, hesaplama başarımları için karşılaşılan zorluklar ve gelecek çalışma planları aktarılmaktadır.

8.1. Kazanımlar

Gelişmiş sinyal işleme uygulamaları için gerçekleştirilen paralel hesaplamalar ve güç verimli hesaplamalar sonucu literatüre katkı ve kazanımlar sunulmuştur. Bu kazanımlar sıralı bir şekilde maddeler halinde belirtilmektedir:

1. Gelişmiş sinyal işleme algoritmaları için gerçek zamanlı uygulamalar 15 Watt gibi düşük güç tüketimi ile enerji verimli olarak sensör senaryoları için entegre edilmiştir.
2. GPGPU mimarilerinin diğer hesaplama birimlerine göre sunmuş olduğu FFT işlem hızı kazancı nedeniyle sayısal alt çevrimi işleminin paralel hesaplamaya uygun STFT tabanlı yaklaşımla gerçek zamanlı bir senaryoda çalışması mümkün olmuştur.

3. Makine öğrenmesi tabanlı hedef tespitinde CNN modeli için bölgesel hesaplama yerine işlemlerin doğrusallık özellikleri kullanılarak tüm girdi üzerinden hesaplamalar gerçekleştirilerek bölgesel çıktılar elde edilmiştir. Ayrıca matris işleminin evrişim işlemine dönüşümü sonucu katman optimizasyonu gerçekleştirilmiştir. Gerçekleşen hesaplama optimizasyonları sonucu Jetson GPU mimarisi ile CPU paralel hesaplamaya göre yaklaşık 1.2x hızlanma elde edilmiş olup güç verimli bir uygulama tasarlanmıştır. Böylece geleneksel tespit yöntemlerine göre yüksek hesaplama karmaşıklığına sahip makine öğrenmesi hesaplama modelleri gerçek zamanlı entegrasyonlarda çalışabilecek hale getirilmiştir.

4. Sıkıştırılmış algılama tabanlı geliş açısı kestirimi ile hedef tespiti sonuçlarının gerçek zamanlı olarak yönlerinin bulunması sağlanmıştır. Ayrıca literatüre ilk defa gerçek zamanlı sıkıştırılmış algılama tabanlı geliş açısı kestirimi uygulaması sunulmuştur.

Tez çalışmasının sonuçları, özellikle hava platformu ve mobilite ihtiyacı duyan sistemlerde düşük güç tüketimi tabanlı hesaplamalar ile daha uzun batarya ömrü ile gerçek zamanlı entegrasyonların çalışmasını sağlamıştır. Entegrasyon planlayıcı ile uygulamaların entegrasyonları benzetilerek tasarım fikirleri sunulmuştur. Entegrasyon sonuçları aynı zamanda uç hesaplama mimarisi için donanım ve yazılım tasarım belirlemede öneriler vermiştir.

8.2. Zorluklar

Gelişmiş sinyal işleme uygulamaları için gerçekleştirilen iyileştirmelerde geliş açısı kestirimi için döngüsel işlemlerden dolayı yaşanan kernel çağırma ek yükü bir başarım zorluğu yaşatmıştır. Bir optimizasyon problemi çözülürken yineleme yaklaşımı ile çözüm için belirlenen kritere yakınsaması kontrol edilmektedir. Bu nedenden ötürü yineleme yaklaşımı ile girdi ve bileşenler güncellendiğinden kernel işlemlerin her defasında yapılması gerekmektedir. Sıkıştırılmış algılama tabanlı geliş açısı kestiriminde yineleme sayısı arttıkça CPU başarımının GPU başarımından daha yüksek olması, yinelemeden dolayı artan kernel çağırma ek

yükü, CPU önbellek kullanımının getirdiği kazanç ve GPU global bellek işlemlerinin başarımı olumsuz etkilemesi ile açıklanabilmektedir. Bu zorluğun üstesinden gelmek için kernel içi iyileştirmeler yapılmıştır. Bu iyileştirmeler gerçekleşse dahi CPU başarımına göre daha düşük bir başarım elde edilmiştir. Yinelemeli yaklaşım içeren algoritma hesaplamalarında CPU birimlerinin önbellek kazancı sağlamasından dolayı yüksek başarım sağlaması durumu gözlemlenmiştir.

8.3. Gelecek Çalışmalar

Gelecek çalışmalarda çok kanallı bir sayısal alt çevrimi uygulaması 4G, 5G ve 6G gibi çalışmalar için modellenip gerçek zamanlı entegrasyonlar hedeflenmektedir. Sayısal alt çevriminde hesaplama iyileştirmeleri olarak frekans uzayında alt örnekleme ve asenkron veri transferi tasarımları planlanmaktadır. Makine öğrenmesi modelinin kayar nokta 16 bit (FP16) çözünürlüğü ile GPGPU mimarilerinde başarımları incelenerek doğruluk ve hesaplama başarımları karşılaştırılacaktır. Sıkıştırılmış algılama tabanlı geliş açısı kestiriminde paralel hesaplamalara uygun hesaplama karmaşıklığını düşürebilecek yöntemlerin çalışılması hedeflenmektedir. Güç verimli hesaplamayı sadece GPGPU mimarileri ile sınırlandırmayıp düşük güç tüketimi sunan FPGA ve CPU birimleri ile çalışmak için araştırma ve geliştirmeler yapılacaktır.

KAYNAKLAR

[1] F. Yavuz. Radar target detection with CNN. In *2021 29th European Signal Processing Conference (EUSIPCO)*, pages 1581–1585. **2021**.

[2] N. Anwar and M. Malik. Comparison of direction of arrival (DOA) estimation techniques for closely spaced targets. *International Journal of Future Computer and Communication*, pages 654–659, **2013**.

[3] M. Xiao, D. Lixia, and Yuping Z. Implementation of a digital down converter using graphics processing unit. In *2013 15th IEEE International Conference on Communication Technology*, pages 655–660. **2013**.

[4] J. Gunther, H. Gunther, and T.K. Moon. GPU acceleration of DSP for communication receivers. *GNU radio*, 2, **2017**.

[5] K. Pavel and David S. Algorithms for efficient computation of convolution. **2013**.

[6] A. V. Oppenheim, R. W. Schafer, and J. R. Buck. *Discrete-Time Signal Processing*. Prentice-Hall, **1999**.

[7] M. Renfors, J. Yli-Kaakinen, and F. J. Harris. Analysis and design of efficient and flexible fast-convolution based multirate filter banks. *IEEE Transactions on Signal Processing*, 62(15):3768–3783, **2014**.

[8] M. I. Jordan and T. M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, **2015**.

[9] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, **2015**.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., **2012**.

[11] M. A. Richards. *Fundamentals of Radar Signal Processing*. McGraw-Hill Professional, **2005**.

[12] F. Yavuz and M. Kalfa. Radar target detection via deep learning. In *2020 28th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4. **2020**.

[13] R. Nitzberg. Constant-false-alarm-rate signal processors for several types of interference. *IEEE Transactions on Aerospace and Electronic Systems*, AES-8(1):27–34, **1972**.

[14] R. Schmidt. Multiple emitter location and signal parameter estimation. *IEEE Transactions on Antennas and Propagation*, 34(3):276–280, **1986**.

[15] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, **1996**.

[16] M. H. Hayes. *Statistical Digital Signal Processing and Modeling*. John Wiley; Sons, Inc., **1996**.

[17] L.C. Godara. Application of antenna arrays to mobile communications. ii. beam-forming and direction-of-arrival considerations. *Proceedings of the IEEE*, 85(8):1195–1245, **1997**.

[18] M. Barlett. An introduction to stochastic processes. *Quarterly Journal of the Royal Meteorological Society*, 81(350):650–650, **1955**.

[19] D.L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, **2006**.

[20] A. Fisne, B. Kilic, A. Güngör, and A. Ozsoy. Efficient heterogeneous parallel programming for compressed sensing based direction of arrival estimation. *Concurrency and Computation: Practice and Experience*, **2021**.

[21] H. E. Güven, A. Güngör, and M. Çetin. An augmented lagrangian method for complex-valued compressed sar imaging. *IEEE Transactions on Computational Imaging*, 2(3):235–250, **2016**.

[22] J. Mitola. Software radios-survey, critical evaluation and future directions. In *Proceedings NTC-92: National Telesystems Conference*, pages 15–23. **1992**.

[23] GNU Radio Website. http://www.gnuradio.org, **Son Erişim 2022**.

[24] A. Agarwal, L. Boppana, and R.K. Kodali. A fractional sample rate conversion filter for a software radio receiver on FPGA. In *2014 IEEE Region 10 Conference*, pages 1–6. **2014**.

[25] P. K. Meher, J. Valls, T.B. Juang, K. Sridharan, and K. Maharatna. 50 years of cordic: Algorithms, architectures, and applications. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 56(9):1893–1907, **2009**.

[26] K. Jiang, Y. Wang, X. Lian, and H. Ma. An improved wideband digital baseband conversion method. In *2013 IEEE Third International Conference on Information Science and Technology*, pages 1522–1525. **2013**.

[27] G. Shao, X. Chen, and L. Yang. Research and implementation of a high performance parallel computing digital down converter on graphics processing unit. *Concurrency and Computation: Practice and Experience*, 29(8):e4042, **2017**.

[28] K. George and C-In H. Chen. Design and performance evaluation of a digital wideband receiver on a hybrid computing platform. In *2011 IEEE International Instrumentation and Measurement Technology Conference*, pages 1–5. **2011**.

[29] V. Adhinarayanan and W.C. Feng. Wideband channelization for software-defined radio via mobile graphics processors. In *2013 International Conference on Parallel and Distributed Systems*, pages 86–93. **2013**.

[30] S. C. Kim and S. S. Bhattacharyya. Implementation of a high-throughput low-latency polyphase channelizer on GPUs. *EURASIP Journal on Advances in Signal Processing*, 2014:141, **2014**.

[31] S. C. Kim, W. L. Plishker, and S. S. Bhattacharyya. An efficient GPU implementation of an arbitrary resampling polyphase channelizer. In *2013 Conference on Design and Architectures for Signal and Image Processing*, pages 231–238. **2013**.

[32] S. C. Kim and S. S. Bhattacharyya. A wideband front-end receiver implementation on GPUs. *IEEE Transactions on Signal Processing*, 64(10):2602–2612, **2016**.

[33] M. Borgerding. Turning overlap-save into a multiband mixing, downsampling filter bank. *IEEE Signal Processing Magazine*, 23(2):158–161, **2006**.

[34] M. Rupniewski, G. Mazurek, J. Gambrych, M. Nałęcz, and R. Karolewski. A real-time embedded heterogeneous GPU/FPGA parallel system for radar signal processing. In *2016 Intl IEEE Conferences on Ubiquitous Intelligence Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress*, pages 1189–1197. **2016**.

[35] F. Meinl, E. Schubert, M. Kunert, and H. Blume. Realtime FPGA-based processing unit for a high-resolution automotive MIMO radar platform. In *2015 European Radar Conference (EuRAD)*, pages 213–216. **2015**.

[36] F. Meinl, E. Schubert, M. Kunert, and H. Blume. An experimental high performance radar system for highly automated driving. In *2017 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*, pages 71–74. **2017**.

[37] M. R. Bales, T. Benson, R. Dickerson, D. Campbell, R. Hersey, and E. Culpepper. Real-time implementations of ordered-statistic cfar. In *2012 IEEE Radar Conference*, pages 0896–0901. **2012**.

[38] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, **2016**.

[39] Y. LeCun and C. Cortes. MNIST handwritten digit database. **2010**.

[40] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, **1998**.

[41] N. Aloysius and M. Geetha. A review on deep convolutional neural networks. In *2017 International Conference on Communication and Signal Processing (ICCSP)*, pages 0588–0592. **2017**.

[42] Y. LeCun, B. Boser, J. S. Denker, R. E. Howard, W. Habbard, L. D. Jackel, and D. Henderson. *Handwritten Digit Recognition with a Back-Propagation Network*, page 396–404. Morgan Kaufmann Publishers Inc., **1990**.

[43] C. Szegedy, L. Wei, J. Yangqing, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9. **2015**.

[44] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. **2015**.

[45] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778. **2016**.

[46] N. Ammour, H. S. Alhichri, Y. Bazi, B. Benjdira, N. A. Alajlan, and M. A. Al Zuair. Deep learning approach for car detection in uav imagery. *Remote. Sens.*, 9:312, **2017**.

[47] L. Wang, J. Tang, and Q. Liao. A study on radar target detection based on deep neural networks. *IEEE Sensors Letters*, 3(3):1–4, **2019**.

[48] R. K. Raney. Synthetic aperture imaging radar and moving targets. *IEEE Transactions on Aerospace and Electronic Systems*, AES-7(3):499–505, **1971**.

[49] E. Mason, B. Yonel, and B. Yazici. Deep learning for radar. In *2017 IEEE Radar Conference (RadarConf)*, pages 1703–1708. **2017**.

[50] S. Chen, H. Wang, F. Xu, and Y.-Q. Jin. Target classification using the deep convolutional networks for SAR images. *IEEE Transactions on Geoscience and Remote Sensing*, 54(8):4806–4817, **2016**.

[51] T. D. Ross, S. W. Worrell, V. J. Velten, J. C. Mossing, and M. L. Bryant. Standard SAR ATR evaluation experiments using the MSTAR public release data set. In *Algorithms for Synthetic Aperture Radar Imagery V*, volume 3370, pages 566 – 573. International Society for Optics and Photonics, SPIE, **1998**.

[52] R. Ravindran, M. J. Santora, and M. M. Jamali. Multi-object detection and tracking, based on DNN, for autonomous vehicles: A review. *IEEE Sensors Journal*, 21(5):5668–5677, **2021**.

[53] J. E. Ball, D. T. Anderson, and Chan C. S. S. Comprehensive survey of deep learning in remote sensing: theories, tools, and challenges for the community. *Journal of Applied Remote Sensing*, 11(4):1 – 54, **2017**.

[54] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, **2014**.

[55] F. Chollet. Keras, github, https://github.com/fchollet/keras, **2015**.

[56] M. Abadi. Tensorflow: Large-scale machine learning on heterogeneous systems, https://www.tensorflow.org/, **2015**.

[57] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, **2016**.

[58] A. Paszke. Automatic differentiation in PyTorch. **2017**.

[59] G. Golcarenarenji, I. Martinez-Alpiste, Q. Wang, and J. Alcaraz Calero. Efficient real-time human detection using unmanned aerial vehicles optical imagery. *International Journal of Remote Sensing*, 42:2440–2462, **2021**.

[60] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia. Path aggregation network for instance segmentation. *CoRR*, abs/1803.01534, **2018**.

[61] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao. YOLOv4: Optimal speed and accuracy of object detection. *CoRR*, abs/2004.10934, **2020**.

[62] M. Capra, B. Bussolino, A. Marchisio, G. Masera, M. Martina, and M. Shafique. Hardware and software optimizations for accelerating deep neural networks: Survey of current trends, challenges, and the road ahead. *IEEE Access*, 8:225134–225180, **2020**.

[63] S. Mittal. A survey on optimized implementation of deep learning models on the NVIDIA Jetson platform. *Journal of Systems Architecture*, 97:428–442, **2019**.

[64] JetsonTx1:supercomputer on-module drives next wave of autonomous machines, https://devblogs.nvidia.com/nvidia-jetson-tx1-supercomputer-on-module-drives-next-wave-of-autonomous-machines, **Son Erişim : 2022**.

[65] JetsonTx2:https://developer.nvidia.com/blog/jetson-tx2-delivers-twice-intelligence-edge/, **Son Erişim : 2022**.

[66] JetsonAGXXavier:https://www.nvidia.com/tr-tr/autonomous-machines/embedded-systems/jetson-agx-xavier/, **Son Erişim : 2022**.

[67] Raspberry Pi:https://www.raspberrypi.com/, **Son Erişim : 2022**.

[68] Intel NUC,https://www.intel.com.tr/content/www/tr/tr/products/details/nuc/mini-pcs.html, **Son Erişim : 2022**.

[69] M. Ibrahim, F. Roemer, and G. Del Galdo. On the design of the measurement matrix for compressed sensing based doa estimation. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3631–3635. **2015**.

[70] B. Kılıç. *Adaptive Techniques in Compressed Sensing based Direction of Arrival Estimation*. Master's thesis, Bilkent University, **2021**.

[71] E.J. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, **2006**.

[72] R. G. Baraniuk. Compressive sensing [lecture notes]. *IEEE Signal Processing Magazine*, 24(4):118–121, **2007**.

[73] D. Malioutov, M. Cetin, and A.S. Willsky. A sparse signal reconstruction perspective for source localization with sensor arrays. *IEEE Transactions on Signal Processing*, 53(8):3010–3022, **2005**.

[74] S.G. Mallat and Zhifeng Z. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, **1993**.

[75] M. Ibrahim, V. Ramireddy, A. Lavrenko, J. König, F. Römer, M. Landmann, M.and Grossmann, G. Del Galdo, and R. S. Thoma. Design and analysis of compressive antenna arrays for direction of arrival estimation. *Signal Processing*, 138:35–47, **2017**.

[76] H. Cramer. Mathematical methods of statistics. **1946**.

[77] C. R. Rao. *Information and the Accuracy Attainable in the Estimation of Statistical Parameters*, pages 235–247. Springer New York, **1992**.

[78] Y. Nardi and A. Rinaldo. Autoregressive process modeling via the lasso procedure. *Journal of Multivariate Analysis*, 102(3):528–549, **2011**.

[79] R. Tibshirani, J. Bien, J. Friedman, T. Hastie, N. Simon, J. Taylor, and R. J. Tibshirani. Strong rules for discarding predictors in lasso-type problems. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74(2):245–266, **2012**.

[80] T. Hastie, R. Tibshirani, and M. Wainwright. *Statistical Learning with Sparsity: The Lasso and Generalizations*. Chapman & Hall/CRC, **2015**.

[81] Q. Shen, W. Liu, W. Cui, and S. Wu. Underdetermined DOA estimation under the compressive sensing framework: A review. *IEEE Access*, 4:8865–8878, **2016**.

[82] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. **2011**.

[83] B. Wahlberg, S. Boyd, M. Annergren, and Y. Wang. An admm algorithm for a class of total variation regularized estimation problems*. *IFAC Proceedings Volumes*, 45(16):83–88, **2012**.

[84] K. C. Toh, R. H. Tütüncü, and M. J. Todd. SDPT3 : A matlab software package for semidefinite programming, version 1.3. *Optimization Methods and Software*, 11(1-4):545–581, **1999**.

[85] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson. Optimal parameter selection for the alternating direction method of multipliers (ADMM): Quadratic problems. *IEEE Transactions on Automatic Control*, 60(3):644–658, **2015**.

[86] A. Güngör. *Compressive sensing methods for multi-contrast magnetic resonance imaging*. Master's thesis, Middle East Technical University, **2017**.

[87] Z. Zhou, J. Feng, Z. Chang, and X. Shen. Energy-efficient edge computing service provisioning for vehicular networks: A consensus ADMM approach. *IEEE Transactions on Vehicular Technology*, 68(5):5087–5099, **2019**.

[88] C. Liang, F. R. Yu, H. Yao, and Z. Han. Virtual resource allocation in information-centric wireless networks with virtualization. *IEEE Transactions on Vehicular Technology*, 65(12):9902–9914, **2016**.

[89] J. Weizenecker, B. Gleich, J. Rahmer, H. Dahnke, and J. Borgert. Three-dimensional real-time in vivomagnetic particle imaging. 54(5):L1–L10, **2009**.

[90] T. Knopp and A. Weber. Sparse reconstruction of the magnetic particle imaging system matrix. *IEEE Transactions on Medical Imaging*, 32(8):1473–1480, **2013**.

[91] S. İlbey, A. Güngör, C. B. Top, E. Ü. Sarıtaş, and H. E. Güven. Real-time three-dimensional image reconstruction using alternating direction method of multipliers for magnetic particle imaging. In *2018 26th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4. **2018**.

[92] Q. Wang, Z. Zhao, and Z. Chen. Fast compressive sensing doa estimation via admm solver. In *2017 IEEE International Conference on Information and Automation (ICIA)*, pages 53–57. **2017**.

[93] B. P. Lathi. *Modern Digital and Analog Communication Systems 3e Osece*. Oxford University Press, Inc., USA, 3rd edition, **1998**.

[94] J. Doweck. Inside intel® core microarchitecture. In *2006 IEEE Hot Chips 18 Symposium (HCS)*, pages 1–35. **2006**.

[95] M. Frigo and S.G. Johnson. Fftw: an adaptive software architecture for the fft. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98*, volume 3, pages 1381–1384 vol.3. **1998**.

[96] M. Frigo and S.G. Johnson. The design and implementation of fftw3. *Proceedings of the IEEE*, 93(2):216–231, **2005**.

[97] Intel W-2123 : https://ark.intel.com/content/www/tr/tr/ark/products/125036/intel-xeon-w-2123-processor-8-25m-cache-3-60-ghz.html/, **Son Erişim : 2022**.

[98] M. J. Flynn. Some computer organizations and their effectiveness. *IEEE Transactions on Computers*, C-21(9):948–960, **1972**.

[99] Intel SIMD : https://www.intel.com/content/www/us/en/docs/intrinsics-guide/index.html/, **Son Erişim : 2022**.

[100] Arms SIMD : https://developer.arm.com/architectures/instruction-sets/simd-isas/neon, **Son Erişim : 2022**.

[101] J. L. Hennessy and D. A. Patterson. *Computer Architecture, Fifth Edition: A Quantitative Approach*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 5th edition, **2011**.

[102] R. Chandra, L. Dagum, D. Kohr, R. Menon, D. Maydan, and J. McDonald. *Parallel programming in OpenMP*. Morgan Kaufmann, **2001**.

[103] Intel FFT Kütüphaneleri, https://www.intel.com/content/www/us/en/developer/articles/technical/ onemkl-ipp-choosing-an-fft.html, **Son Erişim : 2022**.

[104] P. Steinbach and M. Werner. Gearshifft – the FFT benchmark suite for heterogeneous platforms. In *High Performance Computing*, pages 199–216. Springer International Publishing, **2017**.

[105] Nvidia CUDA cuFFT Kütüphanesi, https://docs.nvidia.com/cuda/cufft/index.html, **Son Erişim : 2022**.

[106] S. Faulkner, S. D. Elton, T. A. Lamahewa, and D. Roberts. A reconfigurable wideband streaming channeliser for rf sensing applications: A multiple gpu-based implementation. In *2017 11th International Conference on Signal Processing and Communication Systems (ICSPCS)*, pages 1–8. **2017**.

[107] L. R. Rabiner, B. Gold, and C. K. Yuen. Theory and application of digital signal processing. *IEEE Transactions on Systems, Man, and Cybernetics*, 8(2):146–146, **1978**.

[108] W. Ma and J Lu. An equivalence of fully connected layer and convolutional layer. *CoRR*, abs/1712.01252, **2017**.

[109] Intel oneAPI Programlama Mimarisi, https://www.intel.com/content/www/us/en/developer/tools/oneapi/overview.html, **Son Erişim : 2022**.

[110] Intel oneDNN Kütüphanesi, https://spec.oneapi.io/versions/latest/elements/oneDNN/source/index.html, **Son Erişim : 2022**.

[111] Intel oneAPI Heterojen Hesaplama Mimarisi, https://www.oneapi.io/, **Son Erişim : 2022**.

[112] Intel AVX-512 Komut Seti, https://www.intel.com/content/www/us/en/developer/articles/technical/intel-avx-512-instructions.html, **Son Erişim : 2022**.

[113] J. Nickolls and W. J. Dally. The gpu computing era. *IEEE Micro*, 30(2):56–69, **2010**.

[114] NVIDIA CUDA cuDNN Kütüphanesi, https://developer.nvidia.com/cudnn, **Son Erişim : 2022**.

[115] M. Jorda, P. Valero-Lara, and A. J. Pena. Performance evaluation of cudnn convolution algorithms on nvidia volta gpus. *IEEE Access*, 7:70461–70473, **2019**.

[116] NVIDIA Profil Aracı, https://developer.nvidia.com/nvidia-visual-profiler, **Son Erişim : 2022**.

[117] Z. Cui, H. Quan, Z. Cao, S. Xu, C. Ding, and J. Wu. Sar target cfar detection via gpu parallel operation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 11(12):4884–4894, **2018**.

[118] M. Elad. Optimized projections for compressed sensing. *IEEE Transactions on Signal Processing*, 55(12):5695–5702, **2007**.

[119] D. Carvajalino, J. Martin, and G. Sapiro. Learning to sense sparse signals: Simultaneous sensing matrix and sparsifying dictionary optimization. *IEEE Transactions on Image Processing*, 18(7):1395–1408, **2009**.

[120] D. Kincaid and E. W Cheney. *Numerical analysis: Mathematics of scientific computing*. Pacific Grove, CA:Brooks/Cole, USA, 6th edition, **2002**.

[121] Intel VTune Amplifier Analiz Aracı, https://www.intel.com/content/www/us/en/developer/articles/release-notes/vtune-profiler-release-notes.html, **Son Erişim : 2022**.

[122] M.E. Wall, A. Rechtsteiner, and L.M. Rocha. Singular value decomposition and principal component analysis. *A Practical Approach to Microarray Data Analysis*, pages 91–109, **2003**.

[123] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, **2010**.

[124] Intel Math Kernel Library, https://www.intel.com/content/www/us/en/developer/articles/release-notes/onemkl-release-notes.html, **Son Erişim : 2022**.

[125] NVIDIA CUDA BLAS Kütüphanesi, https://docs.nvidia.com/cuda/cublas/index.html, **Son Erişim : 2022**.

[126] NVIDIA CUDA SOLVER Kütüphanesi, https://docs.nvidia.com/cuda/cusolver/index.html, **Son Erişim : 2022**.

[127] S. Lahabar and P. J. Narayanan. Singular value decomposition on gpu using cuda. In *2009 IEEE International Symposium on Parallel Distributed Processing*, pages 1–10. **2009**.

[128] A Woodbury Max. Inverting modified matrices. In *Memorandum Rept. 42, Statistical Research Group*, page 4. Princeton Univ., **1950**.

[129] Begnaud Francis Hildebrand. *Introduction to Numerical Analysis: 2nd Edition*. Dover Publications, Inc., USA, **1987**.

[130] M. Schubiger, G. Banjac, and J. Lygeros. Gpu acceleration of admm for large-scale quadratic programming. *Journal of Parallel and Distributed Computing*, 144:55–67, **2020**.

[131] S. Pisa, E. Pittella, and E. Piuzzi. A survey of radar systems for medical applications. *IEEE Aerospace and Electronic Systems Magazine*, 31(11):64–81, **2016**.

[132] A. Moreira, P. Prats-Iraola, M. Younis, G. Krieger, I. Hajnsek, and K. P. Papathanassiou. A tutorial on synthetic aperture radar. *IEEE Geoscience and Remote Sensing Magazine*, 1(1):6–43, **2013**.

[133] G.W. Stimson. *Introduction to Airborne Radar*. IEE radar, sonar, navigation, and avionics series. SciTech Pub., **1998**.

[134] M.A. Richards, J.A. Scheer, J. Scheer, and W.A. Holm. *Principles of Modern Radar: Basic Principles, Volume 1*. Electromagnetics and Radar. Institution of Engineering and Technology, **2010**.

[135] Avrupa Amatör Radyo Organizasyonu, https://www.eurao.org/en/node, **Son Erişim : 2022**.

[136] D. Adamy. *EW 103: Tactical Battlefield Communications Electronic Warfare*. Artech House radar library. Artech House radar library, **2009**.

[137] S.J. Roome. Digital radio frequency memory. *Electronics & Communication Engineering Journal*, 2:147–153, **1990**.