BURSA TEKNİK ÜNİVERSİTESİ ❖ FEN BİLİMLERİ ENSTİTÜSÜ

ATALETSEL ÖLÇÜM BİRİMLERİ İLE ÜÇ BOYUTLU YÖRÜNGE TESPİT SİSTEMLERİNİN GELİŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

Sinan ÖZCAN

Elektrik - Elektronik Mühendisliği Anabilim Dalı

ŞUBAT 2021

BURSA TEKNİK ÜNİVERSİTESİ ❖ FEN BİLİMLERİ ENSTİTÜSÜ

ATALETSEL ÖLÇÜM BİRİMLERİ İLE ÜÇ BOYUTLU YÖRÜNGE TESPİT SİSTEMLERİNİN GELİŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

Sinan ÖZCAN
171082303

Elektrik - Elektronik Mühendisliği Anabilim Dalı

Tez Danışmanı: Doç. Dr. Gökhan GELEN

ŞUBAT 2021

BTÜ, Fen Bilimleri Enstitüsü'nün 171082303 numaralı Yüksek Lisans Öğrencisi Sinan ÖZCAN, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı "ATALETSEL ÖLÇÜM BİRİMLERİ İLE ÜÇ BOYUTLU YÖRÜNGE TESPİT SİSTEMLERİNİN GELİŞTİRİLMESİ" başlıklı tezini aşağıda imzaları olan jüri önünde başarı ile sunmuştur.

Tez Danışmanı : **Doç. Dr. Gökhan GELEN** .....
Bursa Teknik Üniversitesi

Jüri Üyeleri : **Doç. Dr. Cemal HANİLÇİ** .....
Bursa Teknik Üniversitesi

Dr. Öğr. Üyesi M. HADİ SUZER .....
Harran Üniversitesi

Teslim Tarihi :
Savunma Tarihi : 12 ŞUBAT 2021

20.04.2016 tarihli Resmi Gazete'de yayımlanan Lisansüstü Eğitim ve Öğretim Yönetmeliğinin 9/2 ve 22/2 maddeleri gereğince; Bu Lisansüstü teze, Bursa Teknik Üniversitesi'nin abonesi olduğu intihal yazılım programı kullanılarak Fen Bilimleri Enstitüsü'nün belirlemiş olduğu ölçütlere uygun rapor alınmıştır.

İNTİHAL BEYANI

Bu tezde görsel, işitsel ve yazılı biçimde sunulan tüm bilgi ve sonuçların akademik ve etik kurallara uyularak tarafımdan elde edildiğini, tez içinde yer alan ancak bu çalışmaya özgü olmayan tüm sonuç ve bilgileri tezde kaynak göstererek belgelediğimi, aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul ettiğimi beyan ederim.

Öğrencinin Adı Soyadı: Sinan ÖZCAN

İmzası :

ÖNSÖZ

Bu tezin gerçekleşmesi için her türlü desteğini esirgemeyen, karşılaştığım sorunlarda bilgi ve tecrübesiyle yol gösteren değerli danışmanım Doç. Dr. Gökhan GELEN'e sonsuz teşekkürlerimi sunarım. Ayrıca hiçbir zaman desteklerini esirgemeyen babam Şali ÖZCAN'a, annem Şükran ÖZCAN'a ve ablam Sena ÖZCAN'a sonsuz sevgilerimi ve saygılarımı sunarım.

Şubat 2021 Sinan ÖZCAN

İÇİNDEKİLER

Sayfa

ÖNSÖZ ..... **V**
İÇİNDEKİLER ..... **Vİ**
KISALTMALAR ..... **Vİİİ**
SEMBOLLER ..... **X**
ÇİZELGE LİSTESİ ..... **Xİ**
ŞEKİL LİSTESİ ..... **Xİİ**
ÖZET ..... **XVİİ**
SUMMARY ..... **XVİİİ**
1. GİRİŞ **1**
1.1 Literatür Araştırması 2
1.2 Tezin Amacı 5
2. TEMEL KAVRAMLAR VE KONULAR **6**
2.1 Deney Düzeneğinde Kullanılan Araçlar 6
2.1.1 Endütriyel robot 6
2.1.2 AÖB sensörleri 8
2.1.3 Arduino mega 2560 mikrodenetleyici kartı 12
2.1.4 Arduino pro mini mikrodenetleyici kartı 13
2.2 Deney Düzeneğinde Kullanılan Veri İyileştirme Algoritmaları 15
2.2.1 Kalman filtresi 15
2.2.2 Medyan filtresi 16
2.2.3 Ortalama filtresi 17
2.2.4 Makine öğrenmesi 17
2.2.4.1 Yapay sinir ağları 20
2.2.4.2 Derin sinir ağları 20
2.2.4.3 Kümeleme analizi 21
2.2.4.4 DNN regresyon analizi 22
3. VERİ SETİNİN OLUŞTURULMASI **23**
3.1 Sensör Kartları Montaj Aparatlarının Mekanik Tasarımı ve İmalatı 23
3.2 Sensörlerin ve Mikrodenetleyici Kartlarının Elektronik Bağlantıları 28
3.3 Endüstriyel Robottan ve Sensorlerden Veri Gönderilmesi 29
3.4 Endüstriyel Robottan ve Sensörlerden Gelen Verilerin Senkronize Edilerek Toplanması 33
4. İVME VERİSİNİN İYİLEŞTİRİLMESİ **40**
4.1 AÖB Sensörlerinden Ölçülen İvme Verilerine Filtre Uygulanması 40
4.1.1 Kalman filtresinin uygulanması 40
4.1.2 Medyan filtresinin uygulanması 50
4.1.3 Ortalama filtresinin uygulanması 53
4.2 Filtrelenmiş İvme Verilerinin Makine Öğrenmesi Tekniği İle Model Eğitilmesiyle İyileştirilmesi 58
4.2.1 Birinci model: tek girdili DNN regresyon modeli 58

4.2.2 İkinci model: geçmiş veriler ile birlikte 10 girdili DNN regresyon modeli . 64
4.2.3 Üçüncü model: geçmiş ve gelecek veriler ile birlikte 19 girdili DNN regresyon modeli 69
5. İYİLEŞTİRİLEN VERİNİN KULLANIMI 75
6. SONUÇLAR VE ÖNERİLER 84
KAYNAKLAR 86
ÖZGEÇMİŞ 89

KISALTMALAR

AÖB **:** Ataletsel Ölçüm Birimi (Inertial Measurement Unit)
CNC **:** Computer Numerical Control
DOF **:** Degrees of Freedom
I^2C **:** Inter-Integrated Circuit
RSI **:** Robot Sensor Interface
UDP **:** User Datagram Protocol
DNN **:** Deep Neural Network
IFR **:** International Federation of Robotics
HRI **:** Human-Robot Interaction
IP **:** Internet Protokol
UWB **:** Ultra-WideBand
GPS **:** Global Positioning System
KRL **:** Kuka Robot Language
ISO **:** International Organization for Standardization
ADR **:** Address
IDE **:** Integrated Development Environment
PWM **:** Pulse-Width Modulation
FTDI **:** Future Technology Devices International
UART **:** Universal Asynchronous Receiver-Transmitter
ANN **:** Artificial Neural Network
SDA **:** Serial Data
SCL **:** Serial Clock
GND **:** Ground
TX **:** Transmitted Data

RX **:** Received Data

SPS **:** Speicherprogrammierbare Steuerung

PLC **:** Programmable Logic Controller

TCP **:** Transmission Control Protocol

ADS.NET **:** Autonomous Decentralized System Network

OPC **:** Ole for Process Control

XML **:** Extensible Markup Language

SEMBOLLER

X	**:** Veri Seti
Y	**:** Filtre Fonksiyonu
$\overrightarrow{V_k}$	**:** *k* anındaki hız
$\overrightarrow{a_k}$	**:** *k* anındaki ivme
$\overrightarrow{X_k}$	**:** *k* anındaki konum
t_k	**:** *k* anındaki zaman
$\widehat{X}_k$	**:** *k* anındaki verinin kalman filtresi uygulanmış hai
K_k	**:** Kalman kazancı
Z_k	**:** Sensörden ölçülen değer
R	**:** Varyans
$\overline{X}$	**:** Aritmetik ortalama

ÇİZELGE LİSTESİ

Sayfa

Çizelge 2.1: Kuka KR 16 R1610 Endüstriyel Robotuna ait teknik özellikler **8**
Çizelge 2.2: Adafruit BNO055 AÖB kartının vektör olarak sağladığı veriler ve birimleri **9**
Çizelge 2.3: I^2C Adres Seçimi **11**
Çizelge 4.1: AÖB sensörlerine ait hesaplanan varyans değerleri. **41**
Çizelge 4.2: Hesaplanan Ortalama Mutlak Hata Değerleri. **50**
Çizelge 4.3: Medyan filtreli verilere ait ortalama mutlak hata değerleri. **53**
Çizelge 4.4: Koşullu ortalama filtreli verilere ait ortalama mutlak hata değerleri. ... **58**
Çizelge 5.1: Tüm Uygulamalara Ait Ortalama Mutlak Hata Değerleri. **75**
Çizelge 5.2: Hız verilerine ait ortalama mutlak hatalar ve son değerlerin mutlak hatası. **81**
Çizelge 5.3: Konum verilerine ait ortalama mutlak hatalar ve son değerlerin mutlak hatası. **81**

ŞEKİL LİSTESİ

Sayfa

Şekil 1.1: İlk endustriyel robot Unimate **1**
Şekil 2.1: Deney Düzeneği. **7**
Şekil 2.2: Deney Düzeneğine ait fotoğraflar. **7**
Şekil 2.3: Kuka KR 16 R1610 Endüstriyel Robot **8**
Şekil 2.4: Adafruit BNO055 9-DOF Mutlak Oryantasyon AÖB. **9**
Şekil 2.5: I^2C Bağlantı Şeması **10**
Şekil 2.6: Uart Bağlantı Şeması **10**
Şekil 2.7: HID- I^2C Bağlantı Şeması **11**
Şekil 2.8: Mekanik bir ivmeölçerin temsili iç kısmı **12**
Şekil 2.9: İvmeölçer entegresinin temsili iç kısmı **12**
Şekil 2.10: Arduino Mega 2560 Mikrodenetleyici Kartı **12**
Şekil 2.11: Arduino Pro Mini Mikrodenetleyici Kartı **14**
Şekil 2.12: FTDI Programlama Kartı **14**
Şekil 2.13: Kalman filtresinin temel adımları: öngörü ve güncelleme **15**
Şekil 2.14: Makine öğrenmesi algoritmaları kullanılarak yapılmış örnek bir nesne tespit uygulaması **18**
Şekil 2.15: Makine öğrenmesi algoritmaları kullanılarak yapılmış örnek bir borsa tahmin uygulaması **19**
Şekil 2.16: Temsili yapay sinir ağı modeli **20**
Şekil 2.17: a) Yapay sinir ağı(ANN) modeli b) Derin sinir ağı(DNN) modeli **21**
Şekil 2.18: Verilerin üç kümeye ayrıldığı bir kümeleme analizi örneği **21**
Şekil 2.19: a) Doğrusal regresyon b) DNN regresyon **22**
Şekil 3.1: Robot kolu flanşına ait bir teknik resim **23**
Şekil 3.2: AÖB Sensör kartına ait bir teknik resim **24**
Şekil 3.3: Arduino Mega 2560 Kartına ait bir teknik resim **24**
Şekil 3.4: Arduino Pro Mini Kartına ait bir teknik resim **25**
Şekil 3.5: Flanş aparatı. **25**
Şekil 3.6: Sabitleyici aparat. **26**
Şekil 3.7: Üst kutu aparatı. **26**
Şekil 3.8: Alt kutu aparatı. **27**
Şekil 3.9: Bağlantı elemanı. **27**
Şekil 3.10: Aparatların toplanıp robot kolu flanşına montajlanmış hali. **28**
Şekil 3.11: Bağlantı Şeması. **29**
Şekil 3.12: RSIVisualShell de hazırlanan blok diagram. **30**
Şekil 3.13: Arduino Pro Mini kartına yüklenen programın akış diyagramı. **31**
Şekil 3.14: Arduino Mega 2560 kartına yüklenen programın akış diyagramı. **32**
Şekil 3.15: Hız ve ivme hesaplama fonksiyonuna ait akış diyagramı. **33**
Şekil 3.16: Çoklu işlemci yöntemiyle hazırlanan veri kaydetme programına ait akış diyagramı **35**

Şekil 3.17: 1. AÖB sensörüne ait X yönlü ivme verisi ve robot kolu flanşına ait ivme verisi. **35**
Şekil 3.18: 1. AÖB sensörüne ait Y yönlü ivme verisi ve robot kolu flanşına ait ivme verisi. **36**
Şekil 3.19: 1. AÖB sensörüne ait Z yönlü ivme verisi ve robot kolu flanşına ait ivme verisi. **36**
Şekil 3.20: 2. AÖB sensörüne ait X yönlü ivme verisi ve robot kolu flanşına ait ivme verisi. **37**
Şekil 3.21: 2. AÖB sensörüne ait Y yönlü ivme verisi ve robot kolu flanşına ait ivme verisi. **37**
Şekil 3.22: 2. AÖB sensörüne ait Z yönlü ivme verisi ve robot kolu flanşına ait ivme verisi. **38**
Şekil 3.23: 3. AÖB sensörüne ait X yönlü ivme verisi ve robot kolu flanşına ait ivme verisi. **38**
Şekil 3.24: 3. AÖB sensörüne ait Y yönlü ivme verisi ve robot kolu flanşına ait ivme verisi. **39**
Şekil 3.25: 3. AÖB sensörüne ait Z yönlü ivme verisi ve robot kolu flanşına ait ivme verisi. **39**
Şekil 4.1: İvme verisi iyileştirme akış diyagramı. **40**
Şekil 4.2: Kalman filtresi fonksiyonunun akış diyagramı. **42**
Şekil 4.3: a) 1. AÖB sensörüne ait X yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi..... **42**
Şekil 4.4: a) 1. AÖB sensörüne ait Y yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi..... **43**
Şekil 4.5: a) 1. AÖB sensörüne ait Z yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi..... **43**
Şekil 4.6: a) 2. AÖB sensörüne ait X yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi..... **44**
Şekil 4.7: a) 2. AÖB sensörüne ait Y yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi..... **44**
Şekil 4.8: a) 2. AÖB sensörüne ait Z yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi..... **45**
Şekil 4.9: a) 3. AÖB sensörüne ait X yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi..... **45**
Şekil 4.10: a) 3. AÖB sensörüne ait Y yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi..... **46**
Şekil 4.11: a) 3. AÖB sensörüne ait Z yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi..... **46**
Şekil 4.12: a) 1. AÖB sensörüne ait X yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği..... **47**
Şekil 4.13: a) 1. AÖB sensörüne ait Y yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği..... **47**
Şekil 4.14: a) 1. AÖB sensörüne ait Z yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği..... **47**
Şekil 4.15: a) 2. AÖB sensörüne ait X yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği..... **48**
Şekil 4.16: a) 2. AÖB sensörüne ait Y yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği..... **48**
Şekil 4.17: a) 2. AÖB sensörüne ait Z yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği..... **48**

Şekil 4.18: a) 3. AÖB sensörüne ait X yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği..... **49**
Şekil 4.19: a) 3. AÖB sensörüne ait Y yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği..... **49**
Şekil 4.20: a) 3. AÖB sensörüne ait Z yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği..... **49**
Şekil 4.21: a) X yönlü ivme verilerine, medyan filtresi uygulanmış veri grafiği, 1. AÖB, 2.AÖB ve 3.AÖB sensörlerine kalman filtresi uygulanmış veri grafiği. b) X yönlü ivme verilerine, medyan filtresi uygulanmış veri grafiği ve robot ivme verisi grafiği. **50**
Şekil 4.22: a) Y yönlü ivme verilerine, medyan filtresi uygulanmış veri grafiği, 1. AÖB, 2.AÖB ve 3.AÖB sensörlerine kalman filtresi uygulanmış veri grafiği. b) Y yönlü ivme verilerine, medyan filtresi uygulanmış veri grafiği ve robot ivme verisi grafiği. **51**
Şekil 4.23: a) Z yönlü ivme verilerine, medyan filtresi uygulanmış veri grafiği, 1. AÖB, 2.AÖB ve 3.AÖB sensörlerine kalman filtresi uygulanmış veri grafiği. b) Z yönlü ivme verilerine, medyan filtresi uygulanmış veri grafiği ve robot ivme verisi grafiği. **51**
Şekil 4.24: X yönlü ivme verilerine, medyan filtresi uygulanmış sonuçlarının hata dağılım grafiği. **52**
Şekil 4.25: Y yönlü ivme verilerine, medyan filtresi uygulanmış sonuçlarının hata dağılım grafiği. **52**
Şekil 4.26: Z yönlü ivme verilerine, medyan filtresi uygulanmış sonuçlarının hata dağılım grafiği. **53**
Şekil 4.27: Koşullu ortalama filtresinin akış diyagramı. **54**
Şekil 4.28: a) X yönlü koşullu ortalama filtre ivme grafiği ve medyan filtresi ivme grafiği b) X yönlü koşullu ortalama filtre ivme grafiği ve robot ivme grafiği. **55**
Şekil 4.29: a) Y yönlü koşullu ortalama filtre ivme grafiği ve medyan filtresi ivme grafiği b) Y yönlü koşullu ortalama filtre ivme grafiği ve robot ivme grafiği. **55**
Şekil 4.30: a) Z yönlü koşullu ortalama filtre ivme grafiği ve medyan filtresi ivme grafiği b) Z yönlü koşullu ortalama filtre ivme grafiği ve robot ivme grafiği. **56**
Şekil 4.31: X yönlü ivme verilerine, koşullu ortalama filtresi uygulanmış sonuçlarının hata dağılım grafiği. **56**
Şekil 4.32: Y yönlü ivme verilerine, koşullu ortalama filtresi uygulanmış sonuçlarının hata dağılım grafiği. **57**
Şekil 4.33: Z yönlü ivme verilerine, koşullu ortalama filtresi uygulanmış sonuçlarının hata dağılım grafiği. **57**
Şekil 4.34: Tek girdili derin sinir ağı modeli. **59**
Şekil 4.35: X ivmesi modeli için çevrim – ortalama mutlak hata grafiği. **59**
Şekil 4.36: a) X yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği. **60**
Şekil 4.37: X yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği. **60**
Şekil 4.38: Y ivmesi modeli için çevrim – ortalama mutlak hata grafiği. **61**
Şekil 4.39: a) Y yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği. **61**
Şekil 4.40: Y yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği. **62**
Şekil 4.41: Z ivmesi modeli için çevrim – ortalama mutlak hata grafiği. **62**

Şekil 4.42: a) Z yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği. **63**
Şekil 4.43: Z yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği. **63**
Şekil 4.44: On girdili derin sinir ağı modeli. **64**
Şekil 4.45: X ivmesi modeli için çevrim – ortalama mutlak hata grafiği. **65**
Şekil 4.46: a) X yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği. **65**
Şekil 4.47: X yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği. **66**
Şekil 4.48: Y ivmesi modeli için çevrim – ortalama mutlak hata grafiği. **66**
Şekil 4.49: a) Y yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği. **67**
Şekil 4.50: Y yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği. **67**
Şekil 4.51: Z ivmesi modeli için çevrim – ortalama mutlak hata grafiği. **68**
Şekil 4.52: a) Z yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği. **68**
Şekil 4.53: Z yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği. **69**
Şekil 4.54: On dokuz girdili derin sinir ağı modeli. **70**
Şekil 4.55: X ivmesi modeli için çevrim – ortalama mutlak hata grafiği. **70**
Şekil 4.56: a) X yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği. **71**
Şekil 4.57: X yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği. **71**
Şekil 4.58: Y ivmesi modeli için çevrim – ortalama mutlak hata grafiği. **72**
Şekil 4.59: a) Y yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği. **72**
Şekil 4.60: Y yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği. **73**
Şekil 4.61: Z ivmesi modeli için çevrim – ortalama mutlak hata grafiği. **73**
Şekil 4.62: a) Z yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği. **74**
Şekil 4.63: Z yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği. **74**
Şekil 5.1: a) 1.AÖB sensörünün X yönlü ivme verileri kullanılarak hesaplanan hız grafiği. b) 1.AÖB sensörünün X yönlü ivme verileri kullanılarak hesaplanan konum grafiği. **76**
Şekil 5.2: a) 1.AÖB sensörünün Y yönlü ivme verileri kullanılarak hesaplanan hız grafiği. b) 1.AÖB sensörünün Y yönlü ivme verileri kullanılarak hesaplanan konum grafiği. **77**
Şekil 5.3: a) 1.AÖB sensörünün Z yönlü ivme verileri kullanılarak hesaplanan hız grafiği. b) 1.AÖB sensörünün Z yönlü ivme verileri kullanılarak hesaplanan konum grafiği. **77**
Şekil 5.4: a) X yönlü filtrelenmiş ivme verileri kullanılarak hesaplanan hız grafiği. **78**
Şekil 5.5: a) Y yönlü filtrelenmiş ivme verileri kullanılarak hesaplanan hız grafiği. **78**
Şekil 5.6: a) Z yönlü filtrelenmiş ivme verileri kullanılarak hesaplanan hız grafiği. **79**
Şekil 5.7: a) X yönlü 3. DNN modeliyle iyileştirilmiş ivme verileri kullanılarak hesaplanan hız grafiği. b) X yönlü 3. DNN modeliyle iyileştirilmiş ivme verileri kullanılarak hesaplanan konum grafiği. **79**
Şekil 5.8: a) Y yönlü 3. DNN modeliyle iyileştirilmiş ivme verileri kullanılarak hesaplanan hız grafiği. b) Y yönlü 3. DNN modeliyle iyileştirilmiş ivme verileri kullanılarak hesaplanan konum grafiği. **80**
Şekil 5.9: a) Z yönlü 3. DNN modeliyle iyileştirilmiş ivme verileri kullanılarak hesaplanan hız grafiği. b) Z yönlü 3. DNN modeliyle iyileştirilmiş ivme verileri kullanılarak hesaplanan konum grafiği. **80**

Şekil 5.10: X yönlü konum değerlerine ait karşılaştırma grafiği. **82**
Şekil 5.11: Y yönlü konum değerlerine ait karşılaştırma grafiği. **82**
Şekil 5.12: Z yönlü konum değerlerine ait karşılaştırma grafiği. **83**

ATALETSEL ÖLÇÜM BİRİMLERİ İLE ÜÇ BOYUTLU YÖRÜNGE TESPİT SİSTEMLERİNİN GELİŞTİRİLMESİ

ÖZET

Ataletsel ölçüm birimleri (AÖB), robotik, tüketici elektroniği ve sağlık gibi çok geniş uygulama alanlarında konum, hız veya yönelimin belirlenmesi için kullanılmaktadır. AÖB'ler içerdiği ivme ölçer, jiroskop ve manyetometre sensörlerinden elde edilen bilgilerin birleştirilmesi ile hassas ölçümler sunabilmektedir. AÖB'lerin en önemli dezavantajı değişimleri her zaman kendilerine göre ölçmeleri ve gürültü kaynaklı hataya yatkın olmalarıdır. Bu tezin amacı, AÖB sensörlerine ait ivme verilerinin bazı filtreler ve makine öğrenmesi yaklaşımları kullanılarak iyileştirilmesidir. Bu iyileştirme çalışması iki aşamalı olarak gerçekleştirilmiştir. Birinci aşamada veriye bazı filtreler uygulanarak gürültü azaltılmaya çalışılmıştır. İkinci aşamada ise veri işaretlenerek makine öğrenmesi yaklaşımı ile iyileştirme modeli geliştirilmiştir. Veri setinin oluşturulması için çok hassas hareket edebilen KUKA marka bir endüstriyel robot kullanılmıştır. AÖB sensörlerinin ve gerekli elektronik kartların endüstriyel robot flanşına sabitlenmesi için bir montaj parçası tasarlanmıştır. Çalışmada, endüstriyel robotun uç noktasının ivme verileri referans ivme değeri olarak kullanılmıştır. Sensörlerin ölçtüğü ivme verileri ve robot denetleyicisinden alınan referans ivme verisini senkron olarak toplamak için Python programlama dilinde çoklu işlemci yaklaşımı ile program oluşturulmuştur.İlk filtreleme olarak her IMU sensörü çıkışına ayrı ayrı Kalman filtresi uygulanmıştır. Daha sonra, bir medyan filtresi ve bir koşullu ortalama filtresi, Kalman filtresinin çıktısına art arda uygulanmıştır. Bu filtrelenmiş veriler ile endüstriyel robottan gelen referans ivme verileri eşleştirilerek veri seti oluşturulmuştur. Bu veri setleri ile üç farklı DNN regrasyon modeli oluşturulup test edilmiştir. Karşılaştırmalı sonuçlara göre, uygulanan filtreleme ve regrasyon modeli ivme verisinde iyileşme sağlamıştır.

Anahtar kelimeler: Ataletsel Ölçüm Birimi (AÖB), ivmeölçer, makine öğrenmesi, robot, DNN, Kalman Filtresi.

IMPROVEMENT OF THREE DIMENSIONAL PATH FINDING SYSTEMS USING INERTIA BASED MEASUREMENT UNITS

SUMMARY

Inertial measurement units (IMU) are used to determine position, speed, or orientation in a wide range of applications such as robotics, consumer electronics, and healthcare. IMUs can offer precise measurements by combining the information obtained from the accelerometer, gyroscope, and magnetometer sensors. The major disadvantage of IMUs is that they always measure changes relative to themselves and are prone to noise-induced error. This thesis aims to improve the acceleration data of IMU sensors using some filters and machine learning approaches. The improvement is carried out in two stages. In the first stage, some filters are applied to the data to decrease the noise. In the second stage, the data is marked and an improvement model is implemented using the machine learning approach. A high precision industrial robot from KUKA is used to create the data set. A mounting apparatus is designed to fix the IMU sensors and necessary electronic cards to the industrial robot flange. In the study, the acceleration data of the endpoint of the industrial robot is used as the reference acceleration value. A program is created in the Python programming language using a multiprocessing approach to synchronously collect the acceleration data measured by the sensors and the reference acceleration data from the robot controller. As the first filtering, the Kalman filter has been applied to each IMU sensor's output separately. Then, a median filter and a conditional mean filter is applied consecutively to the output of Kalman. The data set is obtained by matching this filtered data with the reference acceleration data from the industrial robot. Three different DNN regression models are implemented and tested for these data sets. According to the comparative results, the applied filtering and regression model improved the acceleration data.

Keywords: Inertial Measurement Unit (IMU), accelerometer, machine learning, robot, DNN, Kalman Filter.

1. GİRİŞ

Robot kelimesi, ilk defa 1920 yılında, Çek yazar Karel Capek tarafından "Rossum'un Evrensel Robotları" adlı oyununda kullanılmıştır [1]. İlk tasvirlerine de o yıllarda yazılan bilim kurgu romanlarında rastlanmaktadır. Romanlarda tasvir edilen bu robotlar birer ev hizmetçisi, uzay savaşçısı ya da bir süper kahraman gibi gösterilse de, günümüz teknolojisinde geliştirilen robotlar, o tasvirlerden farklı olmaktadır. Robotların hayatlarımıza ilk etkileri sanayi sektöründe olmuştur. İlk endüstriyel robot General Motors tarafından 1962 yılında üretilen Unimate isimli robottur [2].



Şekil 1.1: İlk endustriyel robot Unimate [2].

Endüstriyel robotlar hassasiyet, hız gerektiren ve tekrara dayalı işlerde başarılı sonuçlar verdiği için kullanım alanları zamanla genişlemiştir. Uluslararası Robotik Federasyonu'nun (IFR) 2020 Eylülde yayınladığı rapora göre 2020 yılı içerisinde dünyada 2.7 milyon yeni robot kurulmuştur [3].

Endüstriyel robotlar hassasiyet ve hız gerektiren işlerde başarılı olmalarına rağmen karmaşık ve karara dayalı işlerde yeterli değillerdir. Bu sebeple karmaşık ve karara bağlı işlerin insanlara yaptırılırken, insanın yetersiz kaldığı hassasiyet, güç ve hız

gerektiren işlerin ise robotlara yaptırıldığı yeni çalışma modeli oluşturulmuştur. Bu yeni çalışma modeli, insan-robot etkileşimi (Human-Robot Interaction)(HRI) olarak anılmaktadır [4]. HRI çalışma modelinde robotlar güç gerektiren ve bir insan için tehlikeli olabilecek işleri yerine getirirken, insanlar da zeka gerektirecek işleri yerine getirebilmektedir [5]. HRI sistemlerinin verimliliğe katkılarının çok yüksek olacağı düşünüldüğünden, akademinin ve sanayi sektörünün ilgisini çekmektedir. Bu sebeple HRI sistemleri üzerine birçok akademik ve sektörel çalışma yapılmıştır.

1.1 Literatür Araştırması

İnsan – Robot etkileşimi konusundaki ilk çalışmalar, 1996 yılında Nortwhestern Üniversitesi'nde yapılmıştır [6]. İlk patentler ise 1997 yılında alınmıştır. Sonraki yıllarda yapılan çalışmaların sayısı hızlı bir artış göstermiştir. Yapılan ilk çalışmalarda, insanların robotlar ile algılayıcılar kullanılarak etkileşime girmeleri sağlanmıştır.

İlk çalışmalarda kullanılan algılayıcılar ile daha çok insan sesi, mimikleri ve hareketleri algılanmaya çalışılmıştır. Ses için mikrofon kullanılmış ve algılanan verinin sınıflandırılması için ses işleme algoritmaları geliştirilmiştir. J. Norberto Pires'in yaptığı çalışma insanların seslerini kullanarak robotlarla etkileşime girmesine güzel bir örnektir [7]. J. Norberto Pires çalışmasında mikrofonlu kulaklık kullanarak robota sesiyle talimatlar vermiş ve bu talitmatları ses işleme motorunda işleyerek robot için anlamlı hale getirmiştir. Robot bu talimatlar doğrultusunda daha önce programlanan görevleri gerçekleştirmiştir. Ses ile etkileşime bir diğer örnek ise Jovica Tasevski ve arkadaşlarının yaptığı çalışmadır [8]. Yaptıkları çalışmada IP kamera ve mikrofon kullanarak görüntü ve ses işlemesi birlikte gerçekleştirilmiştir ama görüntü verileri insanla etkileşime geçmek için değil, belli bir alana giren parçaların şekillerine, renklerine ve ebatlarına göre ayırt edilebilmeleri amaçlıdır. İnsanla etkileşime girebilmek için sadece mikrofondan gelen ses verileri kullanılmıştır. Ayrıca sisteme hoparlör de eklenerek robotunda insanla etkileşime girmesi sağlanmıştır. Sundukları örnek çalışma modelinde robot kamera ile önündeki parçaların konumlarını hesaplamış ve parçaları ebatlarına, şekillerine ve renklerine göre sınıflara ayırmıştır. İnsan sesini kullanarak parçaya ait renk, ebat ve şekil bilgisini girerek robotun almasını istediği parçayı belirtmiştir. Robotun görevi hakkında hoparlörden bilgi vermesiyle beraber insan ile etkileşime girmesi sağlanmıştır.

İnsan hareketlerini algılamak için kullanılan algılayıcılardan biri de kameralardır. Kameralardan elde edilen görüntüler, görüntü işleme algoritmaları ile işlenerek anlamlı hale getirilmiştir. Bu yöntemle yapılan çalışmalardan biri Claus Lenz ve arkadaşlarının yaptığı çalışmadır [9]. Bu çalışmada insan ve robotun ortak çalışma alanı üç kamera ile üç farklı açıdan algılanmış, insan ve robotun bu alandaki hareketleri kontrol edilmiştir. Gerçekleştirdikleri örnek çalışma modelinde insan ve robot aynı iş parçası üzerinde çalışmaktadır. Kamera ile çalışma yapılan alanda insan hareketleri algılandığı an robot, insan için güvenli noktaya çekilmektedir. İnsan işini bitirip o alandan çekilince kamera ile iş parçası üzerinden, işin hangi aşamada olduğu algılanıp robota kalınan yerden devam etmesi sağlanmaktadır. Miguel Simão ve arkadaşları yaptıkları çalışmada insan el hareketlerini ve mimiklerini algılamak için farklı bir algılayıcı ve yöntem denemişlerdir [10]. Çalışmalarında bilek hareketi için manyetik bir izleyici ve el hareketi için bir veri eldiveni kullanmışlardır. Kullanılan sensörlerden gelen veriler kararlı olmadığı için yapay sinir ağı modeli eğiterek 25 tane el mimiği, 10 tane de el hareketi öğretmişlerdir. Eğitilen bu model sayesinde insanların el mimikleri ve hareketlerini kullanarak robotları kontrol etmeleri ve görevlere yönlendirilmeleri sağlanmıştır. İnsan el hareketlerinin algılandığı bir başka çalışma da el hareketini algılamak için atalet ölçüm birim (AÖB) sensörlerinden biri olan jiroskoplar kullanılmıştır [11]. Jiroskoplar bir parçanın X, Y ve Z eksenindeki açılarını ölçecek şekilde konumlandırılıp üçlü paketler halinde montajlamıştır. Jiroskop paketleri insan pazusuna, ön koluna ve eline giydirilmiştir. Böylelikle bu üç uzuva ait rotasyon açıları sensörlerden alınıp, her rotasyon açısı robotun bir ekseni ile eşleştirilerek robotlarda el ile açısal kontrol gerçekleştirilmiştir. AÖB sensörlerinin kullanıldığı bir diğer çalışma da ise jiroskop, ivmeölçer ve manyetometre sensörlerinin füzyon edilerek parçaya ait rotasyonları hesaplayan BNO055 sensör kartları kullanılmıştır [12]. Bu sensör kartları insanın omuzuna, pazusuna ve ön koluna giydirilmiştir. Böylelikle o uzuvlara ait rotasyon bilgileri elde edilmiştir. O rotasyon bilgileri ve uzuv uzunlukları kullanılarak rotasyon kinematiği ve hareket kinematiği matrisleri ile elin vücut bel hizasına göre kartezyen konumu hesaplanmıştır. Hesaplanan kartezyen konum robota iletilerek el ile robotun kartezyen konum kontrolü sağlanmıştır.

Bir diğer makalede tek çeşit algılama ile yapılan etkileşimlerin sorunlarından bahsetmişlerdir [13]. Örnek olarak el mimiklerinin ya da el hareketinin algılandığı

sistemlerde, insanın zorunlu olarak iki el ile çalışma durumlarında kontrolcüleri sıkıntıya sokabileceğini, aynı şekilde ses ile çalışma durumlarında ortamda oluşabilecek ani bir gürültünün sıkıntı oluşturabileceğini, bu tarz durumların güvenlik açığı oluşturduğu öne sürülmüştür. Öneri olarak da birden fazla algılama tekniğinin kullanılması ve bunlar ile hibrit bir sistem yapılması önerilmiştir. Oluşturulan çalışma modelinde de göz hareketlerinin, sesin ve el hareketlerinin ortak algılanarak robot ile etkileşime girmesi sağlanmıştır.

Juan Antonio Corrales Ramon ve arkadaşları da tek çeşit algılamanın yeterli olmadığını ve sıkıntılarını yaptıkları çalışmada bahsetmişlerdir [5]. Onlara göre vücudu referans alarak yapılan el hareketi algılamaları yetersizdir. Vücudun konumunun veya yöneliminin değişmesinde, hesaplamalarda hata oluşabileceği gibi robotla insanın vücut konumunun belirsizliğinden dolayı çarpışma riski de vardır. Çalışmalarında insan vücudunun tüm uzuvlarına AÖB sensörleri yerleştirmişlerdir. Ayrıca vücudun ortamdaki konumunu algılamak için ultra geniş band (Ultra-WideBand) (UWB) teknolojisi kullanmışlar, bu amaçla alanın çeşitli yerlerine erişim noktaları yerleştirmişlerdir. Böylelikle insan vücudunun tüm uzuvlarının robota göre konumlarını hassas bir şekilde algılamayı başarmışlardır.

İnsan – robot etkileşimli çalışmalarda algılayıcı çeşitliliği arttıkça klasik robot kontrolcüleri yeterli gelmemeye başlamıştır. Konuyla ilgili bir makalede HRI sistemlerinde makine öğrenmesi ile geliştirilen kontrolcülerin kullanılması önerilmiştir [4]. Klasik kontrolcü modelleri, aynı anda birçok veriyi işlemek için güçlü ve esnek değildir. Makine öğrenmesi yaklaşımı ile eğitilecek yeni kontrolcü modelleri birçok çeşit verinin aynı anda işlenebileceği kadar güçlü ve esnek olacaktır. Hongyi Liu ve arkadaşları bu yaklaşımla bir HRI uygulaması gerçekleştirmişlerdir [14]. Çalışmalarında vücut duruşunu, el hareketini ve sesi aynı anda algılayarak verilerin derin öğrenme yaklaşımı ile geliştirdikleri kontrolcüde işlenerek robot ile etkileşime girmeleri sağlanmıştır. Bu yaklaşımın kullanılarak yapıldığı bir başka çalışma ise, Madeline Gannon'a ait Manus ismini verdiği çalışmadır [15]. Çalışmasında 10 tane endüstiyel robotun birbirleriyle ve kameralar aracılığı ile insanlarla etkileşime girmelerini sağlamıştır.

Yapılan tüm çalışmalar incelendiğinde, kontrolcülerin doğru kararlar vermelerinin algılayıcıların doğruluğu ile direkt ilişkili olduğu görülmüştür. İncelenen hiçbir çalışmada AÖB sensörlerinden, ivmeölçerin hiçbir endüstriyel uygulamada tek başına

kullanılmadığı görülmüştür. Bunun sebebi ivmeölçerlerin fazla gürültülü çalışmalarından dolayı direk hareket algılayıcısı olarak kullanılamamalarıdır. Genelde başka uygulamalarda, navigasyon yada drone sistemlerinde, GPS (Global Positioning System) verileri ile birlikte füzyon edilerek kullanılmışlardır. Bu sıkıntıya çözüm olarak çeşitli filtreler ile sensörlerin doğruluğu arttırılmaya çalışılmıştır. Örnek olarak, Yeşim Günhan'ın çalışması gösterilebilir [16]. Çalışmasında ivmeölçerlere ait veri hatalarını, deterministik hata ve olasılıksal hata olarak ikiye ayırmıştır. Olasılıksal hataları çözmek için dalgacık dönüşümünü ve kalman filtresinin beraber kullanıldığı yeni bir yöntem önermiş olduğu da görülmektedir. AÖB sensörlerini iyileştirmeye yönelik yapılan bir diğer çalışma ise Joseph M. Mahoney ve Matthew B. Rhudy'nin çalışmasıdır [17]. Yaptıkları çalışmada 14 sağlıklı yetişkinin ayak bileklerine AÖB sensörlerinden, ivmeölçer ve jiroskop takılmış ve bu kişilerin koşu bandında 90 saniye boyunca 3 farklı hızda (1.21, 2.01, 2.68 m / s) hareket etmeleri sağlanmıştır. Toplanan veriler sabit hızlarla işaretlenerek, yapay sinir ağı yaklaşımı ile hızı ayırt edebilecek bir model geliştirilmiştir.

1.2 Tezin Amacı

Sunulan tez kapsamında, endüstriyel robot flanşına bağlı AÖB sensörlerinden elde edilen ivme verilerinin iyileştirilmesi amaçlanmıştır. Sensörlerdeki gürültülerin azaltılması için sensör verilerine kalman filtre uygulanmıştır ve sensör sayısı arttırılarak ölçüm hassasiyeti arttırılmaya çalışılmıştır. Sensörlerden gelen veriler medyan filtreden ve koşullu ortalama filtreden geçirilerek verilerde iyileştirme yapılmıştır. İyileştirilen veriler endüstriyel robot flanşının ivme değerleri ile işaretlenerek makine öğrenmesi yaklaşımı ile iyileştirme modellerinin eğitilmesi sağlanmıştır.

2. TEMEL KAVRAMLAR VE KONULAR

Tez kapsamında, AÖB sensörlerinden gelen ivme bilgilerinin doğruluğunu ölçmek ve işaretli veri oluşturmak için referans ivme verilerini toplamamız gerekmektedir. Referans ivme verisi toplamak için tekrarlanabilirlik hassasiyeti 0.04 mm olan bir endüstiryel robotun kullanılmasına karar verilmiştir. Şekil 2.1'de gösterilen deney düzeneği kurulmuştur.

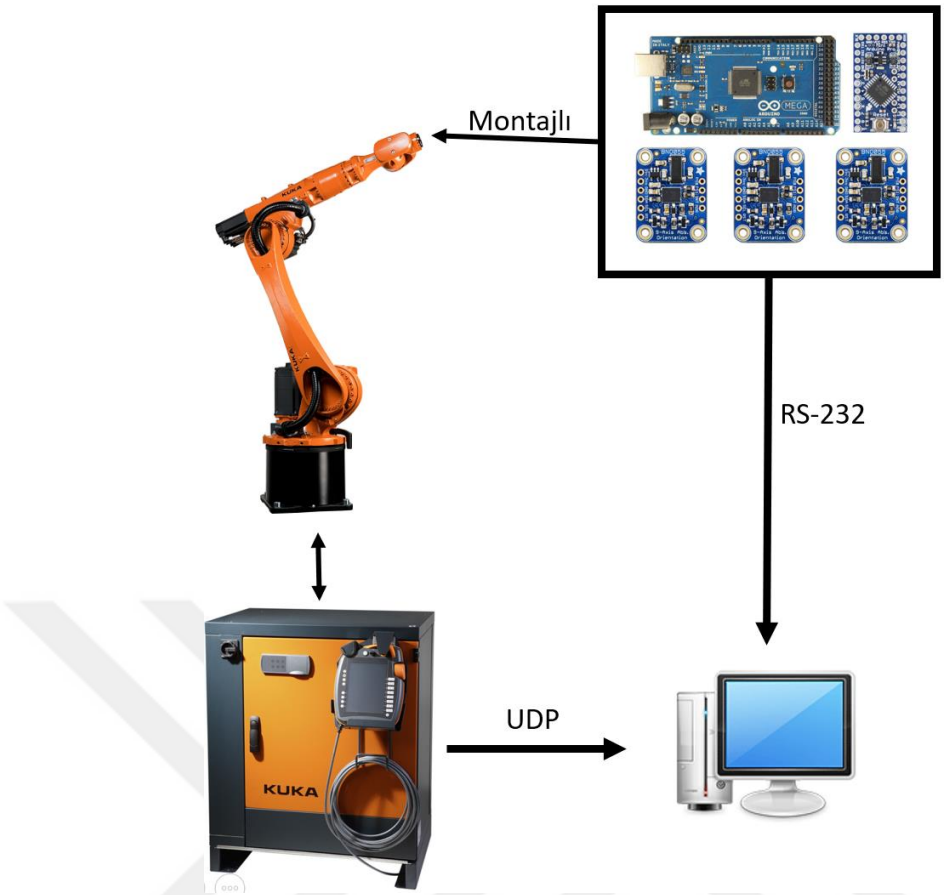
Deney düzeneğinde iki adet BNO055 AÖB sensörü arduino megaya I^2C protokolü ile veri akışı sağlarken bir adet BNO055 AÖB sensörü önce I^2C protokolü ile arduino pro mini kartına, bu karttan da RS-232 protokolü ile arduino megaya veri akışı sağlanmıştır. Kullandığımız Kuka KR 16 R1610 model 6 eksen endüstriyel robot ise RSI (Robot sensör interface) opsiyonu ile UDP protokolünü kullanarak robot flanşının hassas konum bilgisini 4 ms aralıkla tek yönlü olarak aktarmıştır. Arduino, mega sensörlerden aldığı verileri, yaklaşık 15 ms aralıkla seri haberleşme kullanarak, tek yönlü olarak iletmiştir. Böylelikle iki taraflı veriler, ana bilgisayarda senkron edilerek toplanmış ve işaretli verilerin oluştulurması sağlanmıştır.

Deney düzeneğinde üç adet AÖB sensörü doğrultuları aynı olacak şekilde, Arduino pro mini ve Arduino mega kartları ile birlikte kestamit malzemeden CNC'de işlenen kutuya montaj edilmiştir. Kestamit kutuda endüstriyel robotun flaşına montelenmiştir. Böylelikle robotta sağlanacak hareketin ivmesinin doğrudan sensör tarafından ölçülmesi sağlanmıştır. Montajların tamamlanmış halini Şekil 2.2'de görülebilmektedir.

2.1 Deney Düzeneğinde Kullanılan Araçlar

2.1.1 Endütriyel robot

Tez kapsamında, Kuka markasının Şekil 2.3'te görülen KR 16 R1610 model endüstriyel robotu kullanılmıştır. Bu robot KRC4 kontrolcüsü üzerinden KRL programlama diliyle programlanmaktadır.



Şekil 2.1 : Deney Düzeneği.



Şekil 2.2: Deney Düzeneğine ait fotoğraflar.

Kuka firmasına ait ücretsiz yayınlanan WorkVisual programıyla robotun programlanması ve konfigurasyon işlemleri yapılmıştır. Robota ait teknik özellikler Çizelge 2.1'de görülebilmektedir.



Şekil 2.3: Kuka KR 16 R1610 Endüstriyel Robot [18].

Çizelge 2.1: Kuka KR 16 R1610 Endüstriyel Robotuna ait teknik özellikler [18].

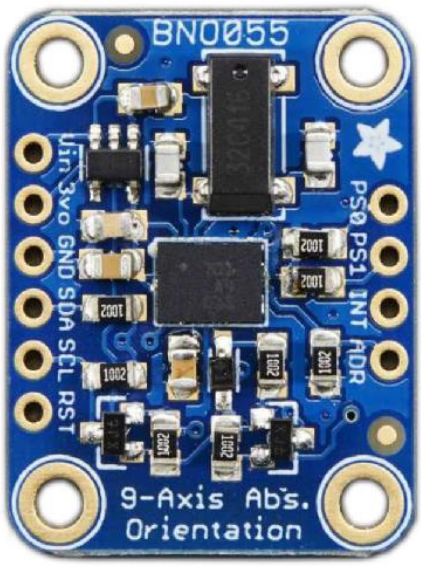
Maksimum Erişim	1612 mm
Maksimum Taşıma Kapasitesi	20 kg
Nominal Taşıma Kapasitesi	16 kg
Tekrarlanabilir Pozisyon hassasiyeti (ISO 9283)	± 0.04 mm
Eksen sayısı	6
Ağırlık	245 kg

Endüstriyel robottan pozisyon bilgilerini hızlı ve hassas çekebilmek için Kuka firmasının kendi robotları için geliştirdiği RSI (Robot Sensor Interface) [19] opsiyonu kullanılmıştır. Bu opsiyon sayesinde robot flanşına ait konum bilgileri 4 ms aralıklarla UDP protolü ile aktarılmıştır.

2.1.2 AÖB sensörleri

Deney düzeneğinde, Bosch firmasının geliştirdiği Şekil 2.4'te görülen BNO055 ivmeölçer entegreli AÖB kartları kullanılmıştır. Bu kartlar adafruit firması tarafından imal edilip satışı yapılmaktadır. Kartların üzerinde ivmeölçerin yanında manyetometre ve jiroskop entegreleri de bulunmaktadır. Üç entegreden 3 boyutlu veriler bastığı için sensör kartı 9-DOF AÖB olarak geçmektedir. Kartın üzerinde ayrıca, Arm Cortex M0 işlemcisi bulunmaktadır. Böylelikle kart entegrelerden çektiği ham verileri direkt

aktarabildiği gibi, işleyip iyileştirerek de basabilmektedir. Adafruit BNO055 AÖB kartının vektör olarak sağladığı verileri ve birimleri Çizelge 2.2'de görülebilmektedir.



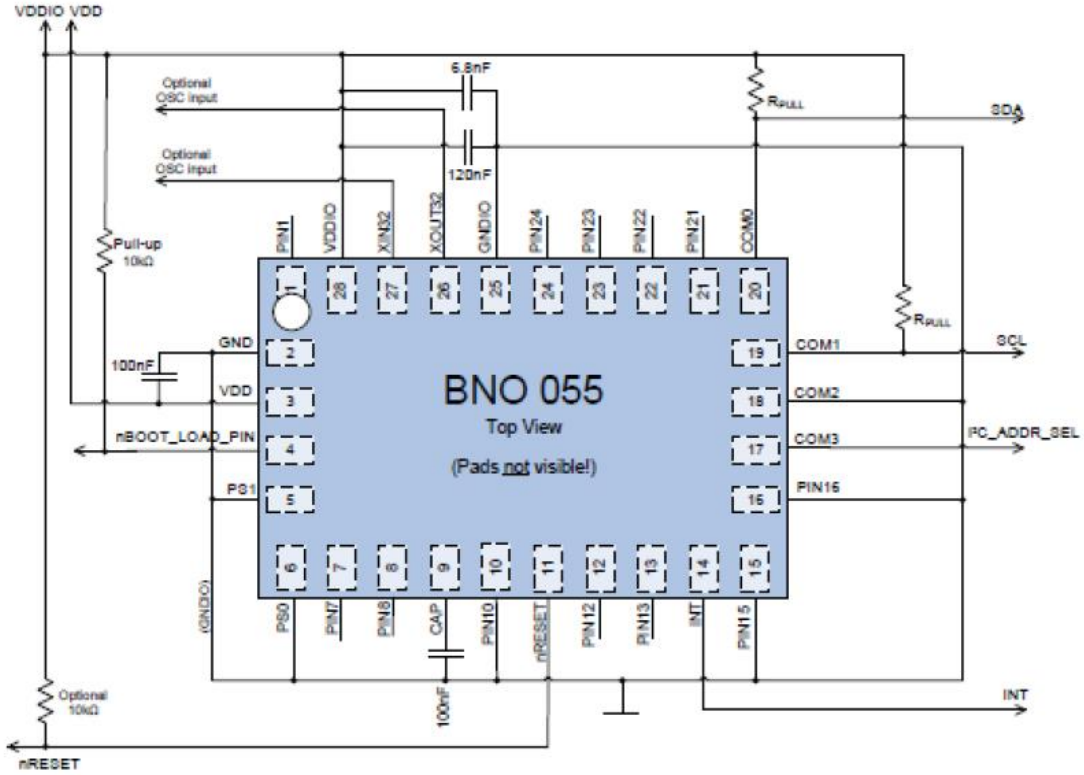
Şekil 2.4: Adafruit BNO055 9-DOF Mutlak Oryantasyon AÖB.

Çizelge 2.2 : Adafruit BNO055 AÖB kartının vektör olarak sağladığı veriler ve birimleri [20]

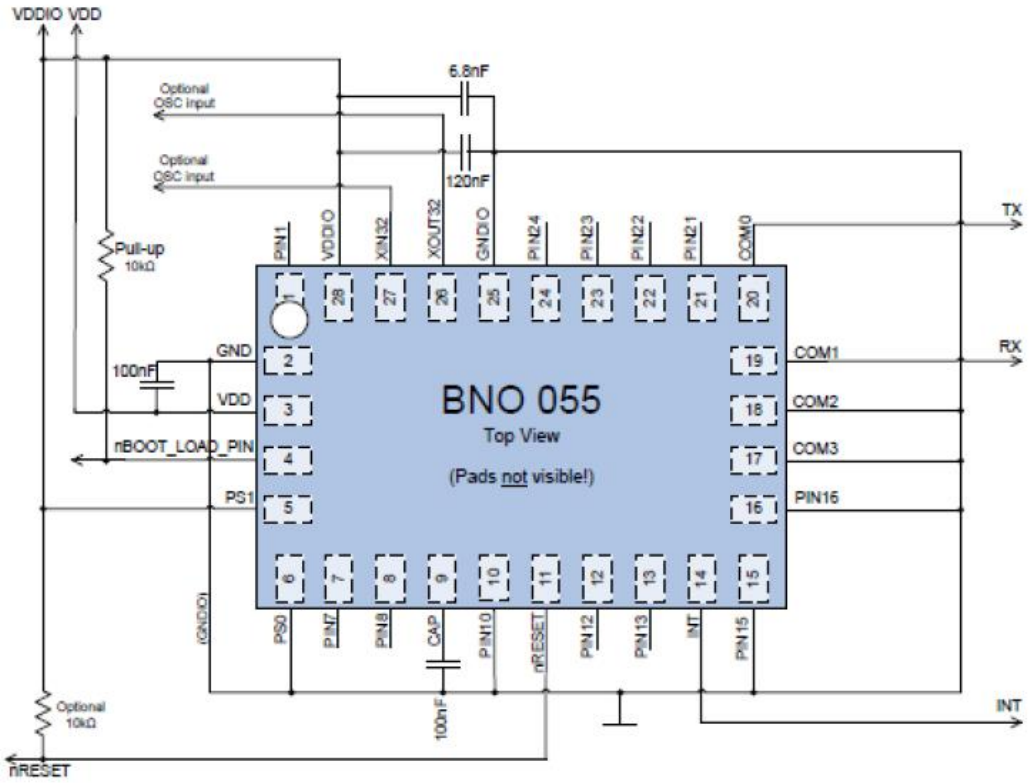
Veri Tipi	Birimi
Manyetometre	uT, micro Teslas
Jiroskop	rps, radyan / saniye
Euler rotasyon	Derece (0..359)
İvmeölçer	m/s^2
Doğrusal İvme	m/s^2
Yerçekimi İvmesi	m/s^2

Bu AÖB kartı, I^2C ,Uart ve HID-I^2C haberleşme protokollerini desteklemektedir [20]. Bu haberleşme protokollerine ait bağlantı şemaları Şekil 2.5, Şekil 2.6 ve Şekil 2.7'da görülmektedir.

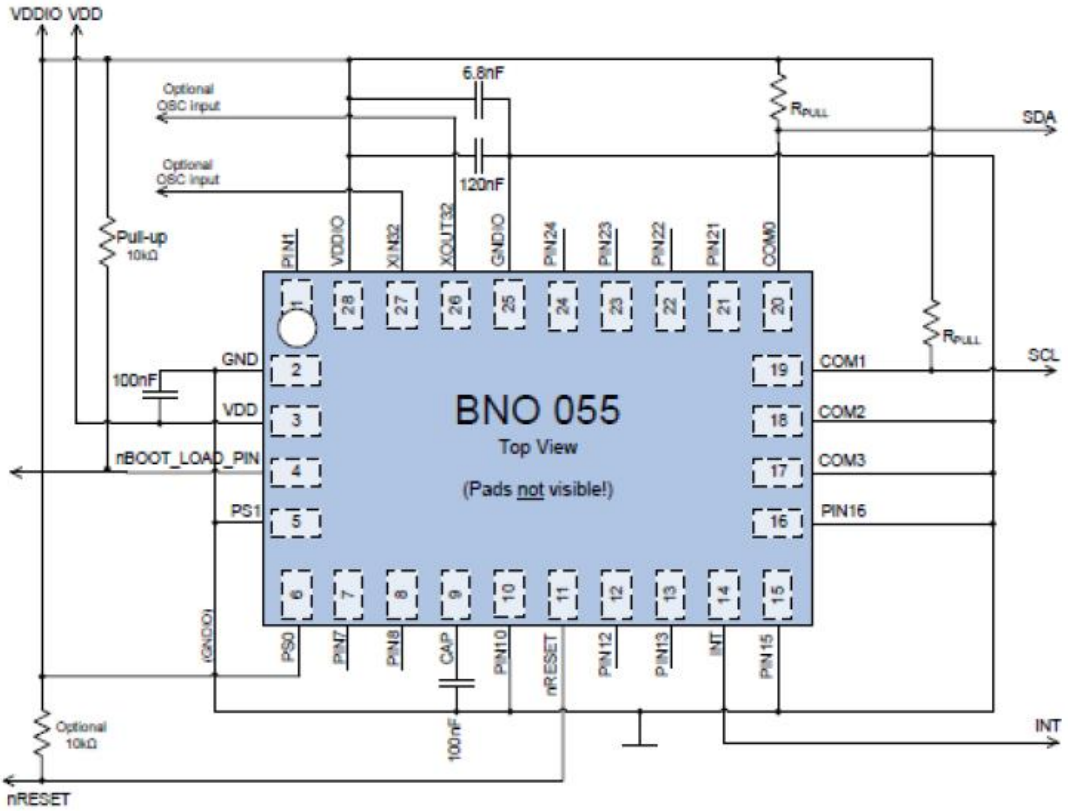
I^2C haberleşme protokolü, master-slave yapısında bir seri haberleşme protokolüdür. Bu haberleşme protokolü ile tek hat üzerinden birden fazla cihaz haberleşebilmektedir. Her master cihaz okuma ve yazma işi yapabilirken, slave cihazlar sadece yazma işi yapabilmektedir. Adafruit firması tarafından AÖB kartları birer slave cihaz olarak düşünülerek tasarlanmıştır ve sadece iki farklı adres alabilmektedirler. Bu sebeple bir hat üzerinden en fazla iki farklı AÖB kartı ile iletişim sağlanabilmektedir. Kartlara ait adresler ADR pinine verilen 0 yada 5 volta göre değişiklik göstermektedir. Çizelge 2.3'te I^2C adres seçimi görülebilmektedir.



Şekil 2.5: I²C Bağlantı Şeması [20].



Şekil 2.6: Uart Bağlantı Şeması [20].



Şekil 2.7: HID- I^{2}C Bağlantı Şeması [20].

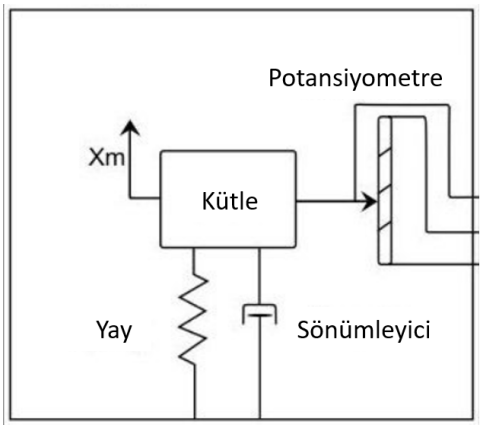
Çizelge 2.3: I^{2}C Adres Seçimi [20].

I^{2}C Konfigürasyonu	ADR Pin	I^{2}C Adresi
Slave	0	0x28
Slave	1	0x29
HID- I^{2}C	0	0x40

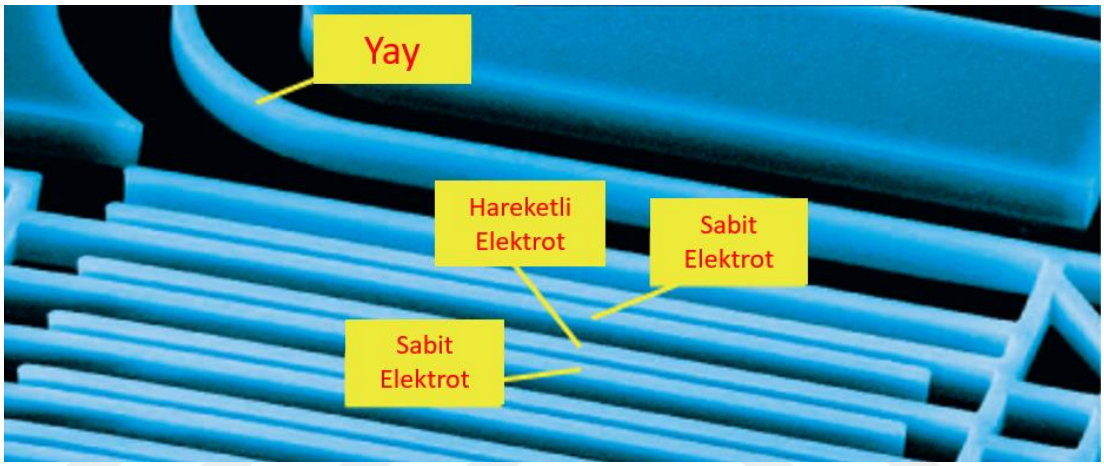
İvmeölçerler bir kütleye uygulanan ivmeyi ölçen cihazlardır [21]. Bu cihazlar kütlesi bilinen bir cihaza uygulanan ivmeyi cihaza uygulanan kuvvetten yola çıkarak hesaplamaktadırlar. Mekanik olarak yaylı ölçüm cihazları olduğu gibi, elektronik entegre olarak kullanılan modelleri de bulunmaktadır.

Mekanik ivmeölçerler, Şekil 2.8'te gösterildiği gibi temelde bir yaya ve dampere bağlı bir kütlenin hareketine bağlı olarak potansiyometrede direnç değişimine sebep olmaktadırlar. Bu direnç değişimiyle ilişkili olarak ivme hesaplanabilmektedir.

Elektronik entegre olarak satılan ivme ölçerler ise Şekil 2.9'ta gösterildiği gibi içlerinde bir adet sabit elektrot ve bir adet yaya bağlı hareketli elektrot barındırmaktadır [23]. Entegre bir ivmeye maruz kalırsa hareketli elektrot salınım yaparak kapasitansta değişime sebep olur ve kapasitanstaki değişim ölçülerek ivme hesaplanabilmektedir.



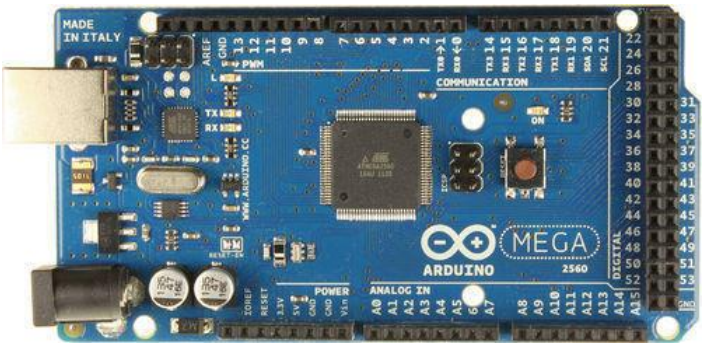
Şekil 2.8: Mekanik bir ivmeölçerin temsili iç kısmı [22].



Şekil 2.9: İvmeölçer entegresinin temsili iç kısmı [23].

2.1.3 Arduino mega 2560 mikrodenetleyici kartı

Arduino Mega 2560, üzerinde Atmel firmasına ait Atmega2560 mikro denetleyicisi olan Şekil 2.10'da gösterilen bir fiziksel programlama kartıdır. Üzerinde hazır bulunan programlayıcı devresi ile Arduino IDE'si üzerinden kolaylıkla programlanabilmektedir.



Şekil 2.10: Arduino Mega 2560 Mikrodenetleyici Kartı [24].

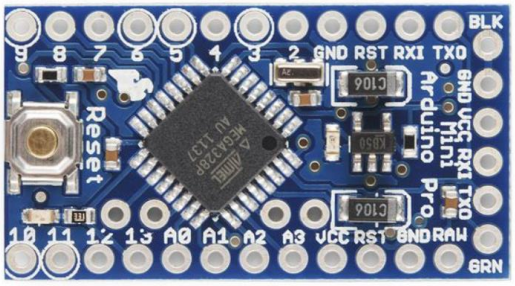
Arduino Mega 2560 kartının seçilmesinin sebebi, I^2C haberleşme protokolünü desteklemesidir. Ayrıca dört farklı cihazla haberleşebilecek şekilde RS-232 pinlerine sahiptir. Bunların haricinde Arduino mega kartına ait teknik özellikler aşağıda sıralanmıştır [24].

- Mikrodenetleyici Atmega2560
- Çalışma Gerilimi 5V
- Giriş Gerilimi (önerilen) 7-12V
- Giriş Gerilimi (limit) 6-20V
- Dijital I/O Pinleri 54 (15 tanesi PWM çıkışı)
- Analog Giriş Pinleri 16
- Her I/O için Akım 40 mA
- 3.3V Çıkış için Akım 50 mA
- Flash Hafıza 256 KB (Atmega2560) 8 KB kadarı bootloader tarafından kullanılmaktadır
- SRAM 8 KB (ATmega2560)
- EEPROM 4 KB (ATmega2560)
- Saat Hızı 16 MHz
- Uzunluk 101.6 mm
- Genişlik 53.4 mm
- Ağırlık 36 g

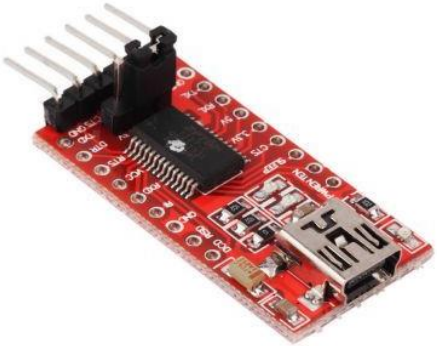
Arduino Mega 2560 kartını kullanarak, iki adet AÖB sensörü ile I^2C haberleşme protokolünü kullanarak direkt haberleşme, bir adet AÖB sensörü ile de arduino pro mini kartı aracılığıyla RS-232 haberleşme protokolünü kullanarak dolaylı haberleşme sağlanmıştır. Son olarak sensörlerden gelen veriler seri port üzerinden ana bilgisayara gönderilmiştir.

2.1.4 Arduino pro mini mikrodenetleyici kartı

Arduino Pro Mini kartı, üzerinde atmel firmasına ait atmega328 mikrodenetleyicisi bulunan Şekil 2.11'de gösterilen bir fiziksel programlama kartıdır. Diğer arduino kartlarına göre daha küçük boyutta olmasından dolayı tercih sebebidir. Diğer kartların aksine üzerinde programlayıcı entegresi bulunmamaktadır. Bu sebeple Arduino IDE'si üzerinden program atmak için Şekil 2.12'de gösterilen bir FTDI programcıya ihtiyaç duymaktadır. Arduino Pro Mini ye ait diğer teknik özellikler aşağıda sıralanmıştır [25].



Şekil 2.11: Arduino Pro Mini Mikrodenetleyici Kartı [25].



Şekil 2.12: FTDI Programlama Kartı [26].

- 16 MHz'de çalışan ATmega328
- Auto-reset desteği
- 5V regülatör devresi
- Max 150mA çıkış
- Aşırı akım koruması
- 5-12V arası DC giriş
- Analog Pin sayısı: 8
- Digital I/O pini sayısı: 14
- Uzunluk: 33 mm
- Genişlik: 18 mm
- Ağırlık: <2 gram

Bu kartın tercih edilmesinin sebebi, 3. AÖB sensör kartının haberleşmesini sağlamaktır. Arduino Pro Mini kartını kullanarak sensörden I^2C protokolü ile dataların çekilip RS-232 seri haberleşme protokolü ile Arduino Mega 2560 kartına yollanması sağlanmıştır.

2.2 Deney Düzeneğinde Kullanılan Veri İyileştirme Algoritmaları

Tez kapsamında, ham veriyi iyileştirmek için Kalman, Medyan ve Ortalama filtreleri kullanılmıştır. Filtrelerden geçen veriler ile bir model eğitmek için makine öğrenmesi tekniklerinden DNN regresyon tekniği kullanılmıştır.

2.2.1 Kalman filtresi

Kalman filtresi, bir durumun çıktısını, geçmiş verileri ve oluşturulan matematiksel modeli kullanarak tahmin etmeye yarayan algoritmadır [27]. Kalman filtresinin temel adımları Şekil 2.13'te gösterilmiştir. Özellikle gürültülü sistemlerde ya da matematiksel modeli bilinen sistemlerde ölçümleri iyileştirmek için kullanılmaktadır. Bir diğer kullanım alanı, uzun ölçüm aralıklı sistemlerin iki ölçüm aralığındaki değerleri zamana bağlı olarak tahmin etmesidir.



Şekil 2.13: Kalman filtresinin temel adımları: öngörü ve güncelleme [27].

Kalman filtresinin ilk olarak, Macar asıllı Amerikan matematikçi Rudolf Kalman tarafından geliştirildiği kabul edilse de, daha önce Thorvald Nicolai Thiele ve Peter Swerling tarafından da benzer modeller geliştirildiği görülmektedir. Stanley F. Schmidt, genellikle bir Kalman filtresinin ilk uygulamasını geliştiren kişi olarak bilinmektedir. Nasa'nın Apollo programında navigasyon sistemlerinde doğrusal olmayan yörünge tahmin uygulamalarında kullanılmıştır [27].

Kalman fitresinde gözlem çıktısı kalman kazancı denilen bir katsayı ile ölçeklendirilerek modele girdi olarak verilmektedir. Kazanç katsayısı değiştirilerek, model iyileştirme performansı ayarlanabilmektedir. Örneğin, kazanç katsayısının yüksek olması gürültü miktarını azaltabileceği gibi, sistemin tepki süresini uzatıp ölçümlerde gecikmeye de sebep olabilmektedir. Aynı şekilde kazanç katsayısının

azaltılması da tepki süresini azaltacağı gibi, sistemdeki gürültünün artmasına sebep de olabilmektedir. Kazanç katsayısının incelenip en optimum seviyede tutulması gerekmektedir.

2.2.2 Medyan filtresi

Medyan filtresi, doğrusal olmayan sayısal filtreleme tekniğidir. Genellikle görüntü işlemede ve sinyal işlemede kullanılmaktadır. Filtrenin amacı sinyaldeki ya da görüntüdeki gürültüleri azaltmaktır [28].

Çalışma mantığı, belirtilen sayıdaki geçmiş veriyi tutarak her zaman çıkış olarak ortanca veriyi vermesidir. Örneğin 2.1'de verilen bir veri kümesinin 3 boyutlu bir medyan filtreden adım adım çıktısı incelenmektedir:

$$X = (2,12,36,22,14,20,30) \tag{2.1}$$

Y fonksiyonu filtre fonksiyonu olsun.

$$Y_1 = median(2,12,36) = 12 \tag{2.2}$$

$$Y_2 = median(12,36,22) = 22 \tag{2.3}$$

$$Y_3 = median(36,22,14) = 22 \tag{2.4}$$

$$Y_4 = median(22,14,20) = 20 \tag{2.5}$$

$$Y_5 = median(14,20,30) = 20 \tag{2.6}$$

Görüldüğü üzere veriler kendilerinden önceki üç veri ile birlikte değerlendirilerek veri setindeki çok büyük ve çok küçük değerlerden temizlenmiştir. Böylelikle girdideki gürültü miktarı azaltılmıştır.

Medyan filtrenin sorunu ise, sistemin tepki süresinde gecikmeye sebep olmasıdır. Bundan dolayı deney düzeneğinde kendinden önceki veriler ile işlemek yerine, eş zamanlı olarak aynı ölçümü yapan üç sensör kullanılarak bu üç sensörden gelen veri filtrelenmiştir. Örnek olarak, 2.7 ,2.8 ve 2.9'daki veri setlerinin ayrı sensörlerden gelen veri setleri olduğu varsayılmaktadır.

$$X_1 = (2,12,36,22,14,20,30) \tag{2.7}$$

$$X_2 = (5,6,4,10,33,15,8) \tag{2.8}$$

$$X_3 = (9,2,1,40,32,16,28) \tag{2.9}$$

Y fonksiyonu filtre fonksiyonu olsun.

$$Y_1 = median(2,5,9) = 5 \tag{2.10}$$

$$Y_2 = median(12,6,2) = 2 \tag{2.11}$$

$$Y_3 = median(36,4,1) = 4 \tag{2.12}$$

$$Y_4 = median(22,10,40) = 22 \quad (2.13)$$

$$Y_5 = median(14,33,32) = 32 \quad (2.14)$$

$$Y_6 = median(20,15,16) = 16 \quad (2.15)$$

$$Y_5 = median(30,8,28) = 28 \quad (2.16)$$

2.2.3 Ortalama filtresi

Ortalama filtresi, doğrusal olmayan bir sayısal fitreleme tekniğidir. Veri setindeki gürültüleri azaltmak için kullanılmaktadır. Genellikle görüntü işlemede ve sinyal işlemede tercih edilmektedir. Medyan filtreden farkı veri setini belirli boyutta kümelere ayırarak kümenin ortanca değeri yerine kümenin ortalamasını çıktı olarak vermesidir.

Bu tez kapsamında, üç ayrı sensörden gelen verileri ortalama filtreye sokarak gürültüyü azaltma yoluna gidilmiştir. Örnek olarak 2.17-2.19 arasındaki veri setleri, üç ayrı sensörden gelen veri setleri olarak varsayılmıştır.

$$X_1 = (2,12,36) \quad (2.17)$$

$$X_2 = (5,6,4) \quad (2.18)$$

$$X_3 = (9,2,1) \quad (2.19)$$

Y fonksiyonu filtre fonksiyonu olsun.

$$Y_1 = \left(\frac{2+5+9}{3}\right) = 5.334 \quad (2.20)$$

$$Y_2 = \left(\frac{12+6+2}{3}\right) = 6.667 \quad (2.21)$$

$$Y_3 = \left(\frac{36+4+1}{3}\right) = 13.667 \quad (2.22)$$

2.2.4 Makine öğrenmesi

Makine öğrenmesi, çeşitli algılayıcılarla ya da insan gözlemleri sonucu oluşturulan veri kümelerini, çeşitli algoritmalar ile kullanarak, tahmin etme modelleri oluşturmayı amaçlayan bir bilim dalıdır. İlk örneklerine 1959 yılında rastlanmıştır [29]. Temelinde insan beyninden ilham almıştır. Nasıl insanlar doğumundan ölümüne kadar olan süreçte kendi gözlemlerine dayalı olarak bilgiler inşa ediyorsa, makine öğrenmesinin de aynı felsefede çalışması amaçlanmaktadır.

Makine öğrenmesinin, insanın öğrenmesine karşı dezavantajları olduğu gibi avantajları da vardır. Dezavantaj olarak insan beyninin aynı anda birçok konuya ve alana karşı eğitilebiliyor oluşu, birçok konuya odaklı olarak çalışmasını kısıtlayacak

sınırları bulunmaması örnek verilebilmektedir. Makinelerin ise birden fazla konuya odaklanmaları işlemci güçlerine ve geliştirilen modelin esnekliğine bağlıdır. Günümüz teknolojisinde halen insan beyni kadar stabil ve çok odaklı çalışabilen bir işlemci bulunmamaktadır. Avantajlı olduğu konular ise, insan beyni bazı zor konularda yıllarca çalıştıktan ve gözlem yaptıktan sonra tecrübe kazanır ve doğru karar verme ihtimalini arttırır. Makine öğrenmesi ise, zor konularda veri kümelerinin büyüklüğü ile orantılı olarak doğru karar ihtimalini arttırmaktadır. İnsan yıllarca yaptığı bu gözlemi, beyni dışında bir yerde depolayamadığı için ölmesiyle birlikte tüm tecrübesi de onunla birlikte yok olmaktadır. Başka bir insana ya da ortama aktarması ise yine aylar ya da yıllar sürebilmekte ve bu aktarım kendi beynindeki gözlem sonuçları kadar ayrıntılı olmamaktadır. Makineler ise edindikleri gözlemleri ve sonuçları birbirleriyle hızlıca ve ayrıntılı bir biçimde paylaşabilmektedir. Yıllarca çalışan bir makinenin edindiği gözlemler, makinenin yıpranmasına ve içindeki teknolojinin geri kalmasına bağlı olarak, yeni ve daha ileri bir teknolojiye sahip bir makineye aktarılıp öğrenmenin daha ileriye taşınması da sağlanabilmektedir.

Günümüzde makine öğrenmesi dijital reklamcılık, otopilot uygulamaları, borsa, dijital yayın servisleri, insansız hava araçaları, görüntü işleme uygulamaları, endüstri 4.0'a dayalı otomasyon uygulamaları gibi ve benzeri bir çok alanda kullanılmaktadır. Şekil 2.14'te makine öğrenmesi ile yapılan örnek bir nesne tespit uygulaması görülmektedir. Gösterilen uygulamada insanlar, arabalar, motorsikletler ayrı ayrı tespit edilmektedir. Şekil 2.15'te ise geçmiş hisse rakamları kullanılarak yapılmış örnek bir borsa tahmin uygulaması görülmektedir.



Şekil 2.14: Makine öğrenmesi algoritmaları kullanılarak yapılmış örnek bir nesne tespit uygulaması [30].



Şekil 2.15: Makine öğrenmesi algoritmaları kullanılarak yapılmış örnek bir borsa tahmin uygulaması [31].

Makine öğrenme algoritmaları amaçladıkları hedef doğrultusunda beş kategoriye ayrılmıştır. Kategoriler aşağıda sıralanmıştır [32];

- Gözetimli Öğrenme
- Gözetimsiz Öğrenme
- Yarı Gözetimli Öğrenme
- Pekiştirmeli Öğrenme
- Öğrenmeyi Öğrenme

Gözetimli öğrenme, giriş ve çıkış verilerinin birlikte kullanılarak yapıldığı öğrenme yöntemidir. Her girdi verisine karşı hedeflenen çıktı verisinin kaydedildiği verilere işaretli veri denilmektedir. Girdi verileri, genellikle bir algılayıcıyla ya da insan gözlemine dayalı olarak işaretlenmektedir.

Gözetimsiz öğrenmede işaretsiz veriler kullanılmaktadır. Bu öğrenme çeşidi genellikle, işaretli verilerin pahalı olduğu durumlarda ya da işaretleme imkanının olmadığı durumlarda kullanılmaktadır. Genellikle verileri işleyerek aralarında ilişki kurmaya ve kurulan ilişkiler sayesinde verileri kümelemeye çalışmaktadır.

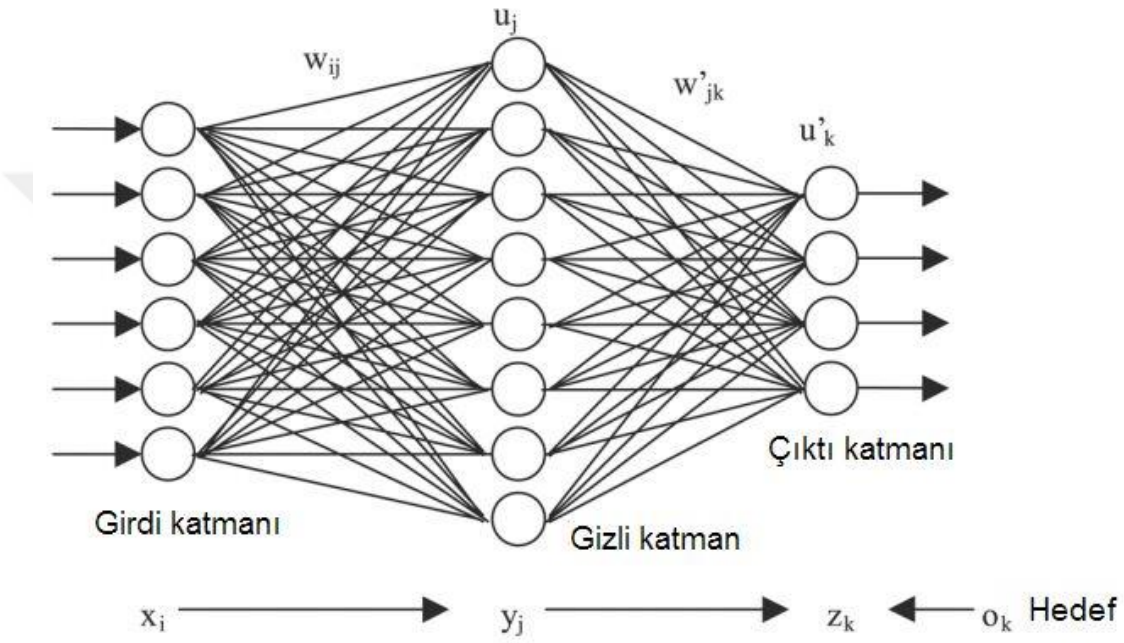
Yarı gözetimli öğrenme, işaretli ve işaretsiz verilerin birlikte kullanıldıkları öğrenme çeşididir. Verinin bir kısmı işaretlenerek tahmin algoritması güçlendirilmektedir.

Pekiştirmeli öğrenmede, eğitilen modele ayrı bir algılayıcı ile sürekli geri bildirim yapılmakta ve ödül ceza yöntemi kullanılarak model her zaman güçlendirilmektedir.

Öğrenmeyi öğrenme algoritmaları, daha önce eğitilen modeller üzerinden daha stabil öğrenme algoritmaları geliştirme yöntemidir.

2.2.4.1 Yapay sinir ağları

Yapay sinir ağları (Artificial Neural Network) (ANN), insan beyninin bilgiyi işleme tekniğinden ilham alınarak geliştirilmiş bir makine öğrenme algoritmasıdır. İnsan beyninin içindeki nöronların birbirleri ile kurdukları sinaptik bağların benzerleri dijital olarak oluşturularak algoritmalar modellenmektedir [33].

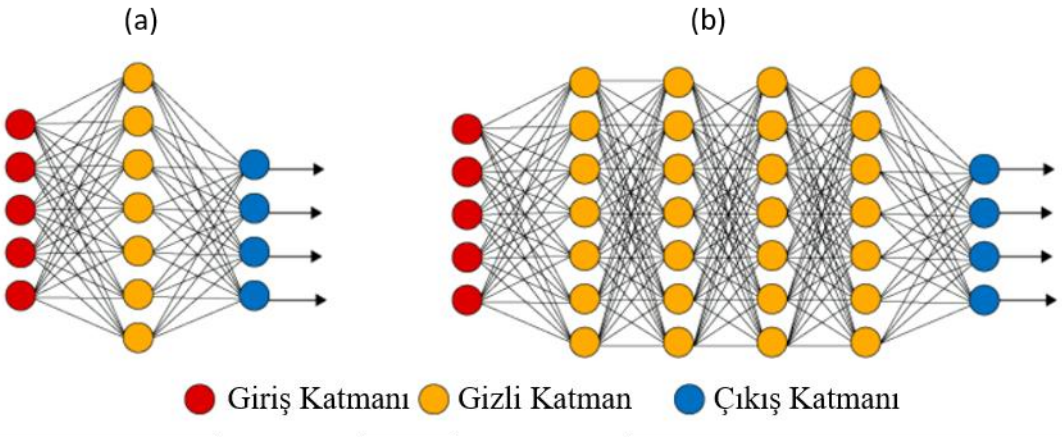


Şekil 2.16: Temsili yapay sinir ağı modeli [34].

Bir yapay sinir ağı modeli Şekil 2.16'da gösterildiği gibi girdi katmanı, gizli katmanlar ve çıktı katmanından meydana gelmektedir. Gizli katmanların eğitilmesi ile model meydana getirilmektedir. Yapay sinir ağları tek giriş-tek çıkışlı, tek giriş-çoklu çıkış, çoklu giriş- tek çıkış ve çoklu giriş-çoklu çıkış olmak üzere dört farklı modelde olabilmektedir.

2.2.4.2 Derin sinir ağları

Yapay sinir ağlarında gizli katman sayısının birden fazla olduğu modellere derin sinir ağı (Deep Neural Network) (DNN) modeli denir [35]. Aralarındaki fark Şekil 2.17'de gösterilmektedir.

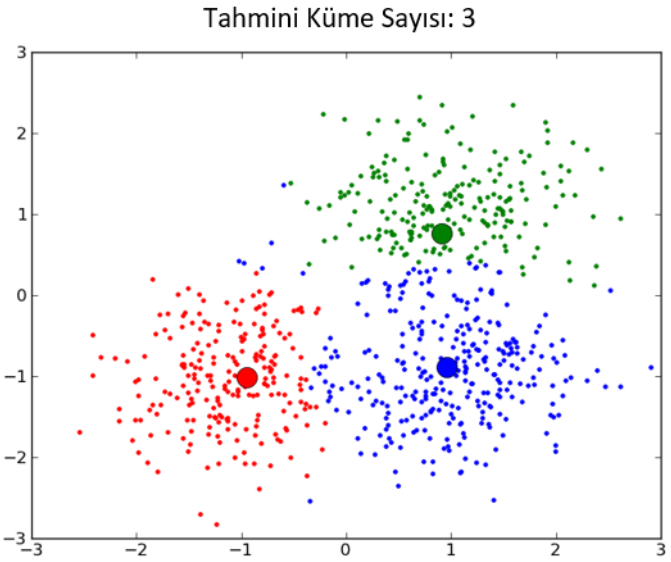


Şekil 2.17: a) Yapay sinir ağı(ANN) modeli b) Derin sinir ağı(DNN) modeli [35].

Derin sinir ağları, diğer yapay sinir ağı modellerine kıyasla eğitilmeleri daha uzun sürmektedir. Ayrıca daha fazla işlemci gücüne ihtiyaç duymaktadırlar. Bu dezavantajlarına karşılık diğer yapay sinir ağı modellerinden daha hassas ve kesin sonuçlar verebilmektedirler.

2.2.4.3 Kümeleme analizi

Kümeleme analizi, verilerin birbirleriyle benzerlikleri dikkate alınarak ayrıldıkları bir öğrenme yöntemidir [36]. Şekil 2.18'de görüldüğü gibi veriler birbirleri ile yakınlıklarına göre gruplanmaktadır. Bu öğrenme çeşidinde çıktının sayısal bir çıktıdan çok tanım içermesi beklenmektedir. En çok kullanıldıkları uygulamalar nesne tespit uygulamalarıdır.

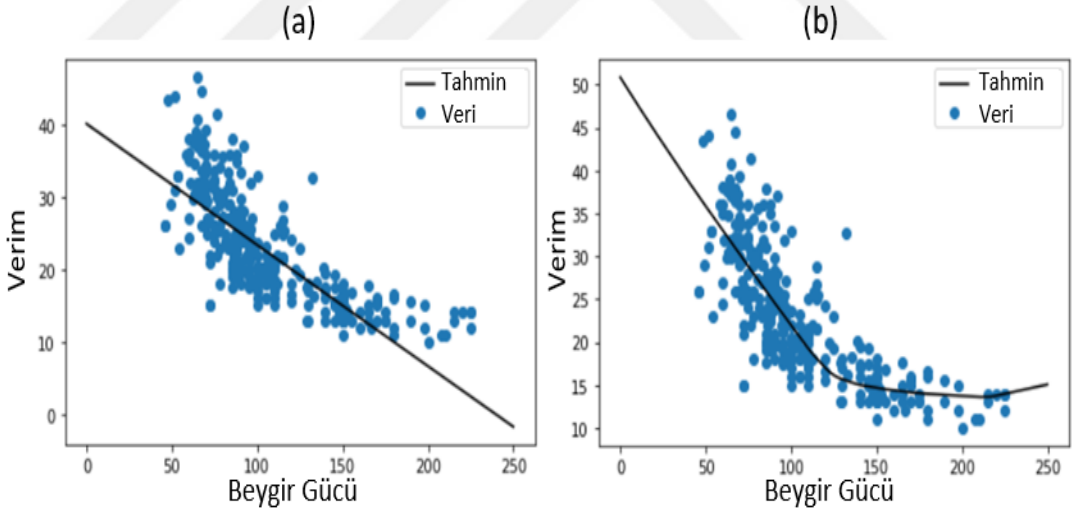


Şekil 2.18: Verilerin üç kümeye ayrıldığı bir kümeleme analizi örneği [36].

Örnek olarak elde kedi, köpek ve insana ait olarak ayrı ayrı işaretlenmiş verilerin olduğu bir veri seti olduğu varsayılmaktadır. Bu veri setini kullanarak görseldekinin insan mı, kedi mi yoksa köpek mi olduğunu tanımlayabilecek bir model inşa etmeye karar verilmiştir. Bu veri setini kullanarak geliştirilen yapay sinir ağı modeli, üç çıkıştan 0 ile 1 arasında çıkış vererek görselin kümelere ait benzerlik oranlarını belirtmektedir. Benzerlik oranının en fazla olduğu küme tanım olarak kullanılmaktadır.

2.2.4.4 DNN regresyon analizi

DNN regresyon analizi, girdiler ve çıktılar arası ilişki kurarak sayısal bir değer döndüren modellerin oluşmasını sağlayan bir öğrenme algoritmasıdır. Klasik regresyonda analizleri doğrusal, üstel, polinomal ya da exponansiyel gibi bir yaklaşımla eğri fit etmeye çalışırken, DNN regresyon daha esnek bir modelin eğitilmesine olanak vermektedir. Örnek olarak, beygir gücüne karşılık, yakıt verimliliğinin tahmin edildiği bir doğrusal regresyon ile DNN regresyon arasındaki fark Şekil 2.19'te görülebilmektedir.



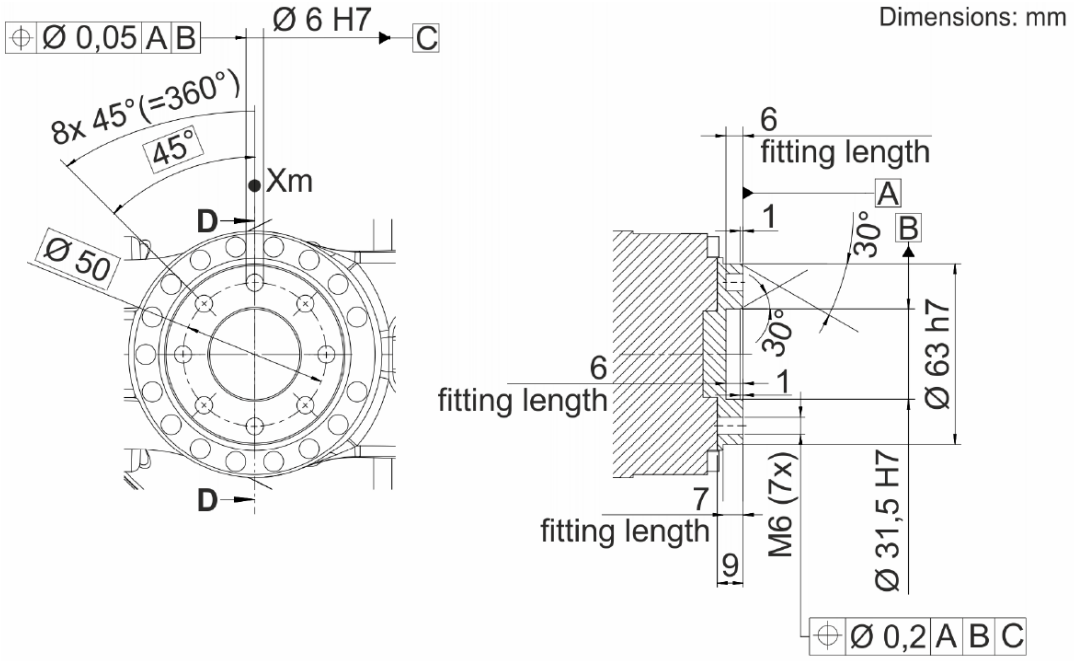
Şekil 2.19: a) Doğrusal regresyon b) DNN regresyon [37].

Yapılan çalışmalarda, sayısal değer döndürmesi ve eğri fit etme esnekliğinden dolayı DNN regresyon yöntemi kullanılmıştır.

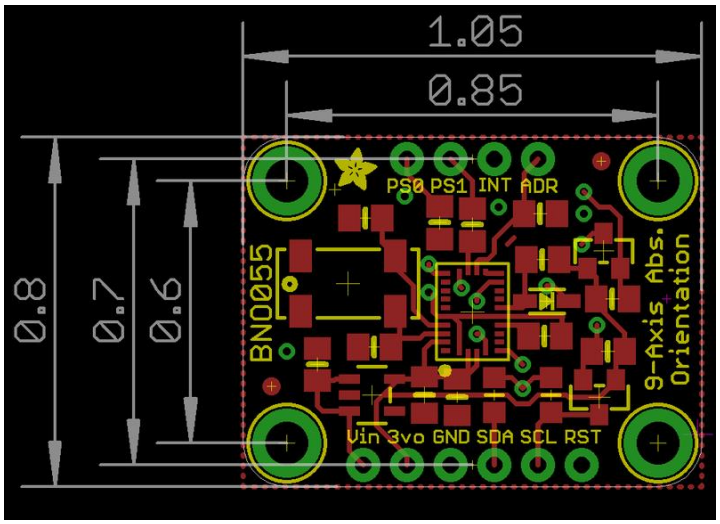
3. VERİ SETİNİN OLUŞTURULMASI

3.1 Sensör Kartları Montaj Aparatlarının Mekanik Tasarımı ve İmalatı

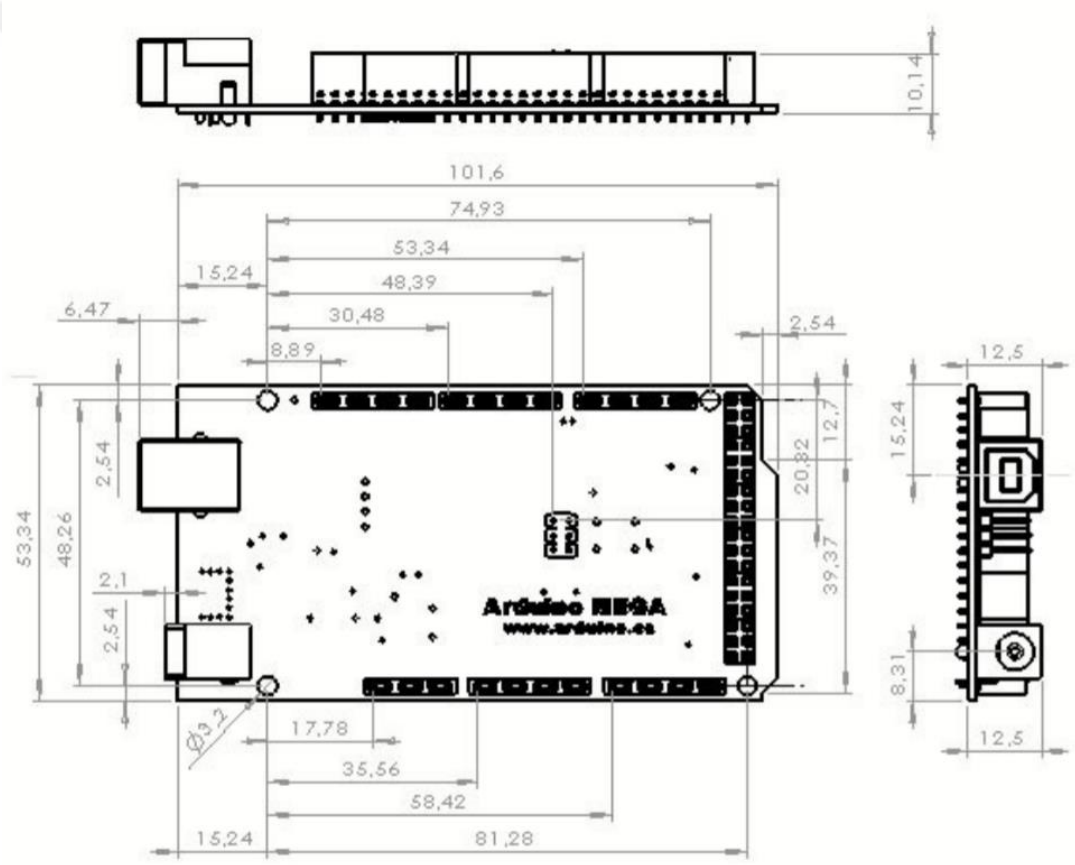
Tez kapsamında, işaretli veri toplamak için AÖB sensörlerinin robot kolunun flanşına sabitlenmesi gerekmektedir. Bu amaçla, AÖB sensörleri ve mikrodenetleyici kartlar için üç boyutlu modelleme programı kullanarak montaj aparatı tasarlanmıştır. Bu aparatı tasarlarken, robot kolu flanşına ait bir teknik resim, AÖB sensörlerine ait bir teknik resim, Arduino Mega 2560 kartına ait bir teknik resim ve Arduino Pro Mini kartına ait bir teknik resim referans alınmıştır ve o teknik resimlerdeki ölçüler kullanılmıştır. Teknik resimleri Şekil 3.1 - 3.4 arasında görmek mümkündür.



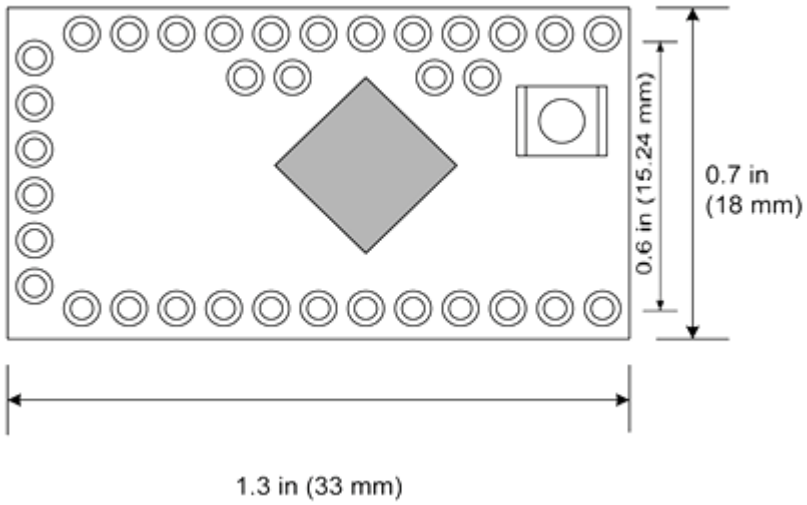
Şekil 3.1: Robot kolu flanşına ait bir teknik resim [18].



Şekil 3.2: AÖB Sensör kartına ait bir teknik resim [38].

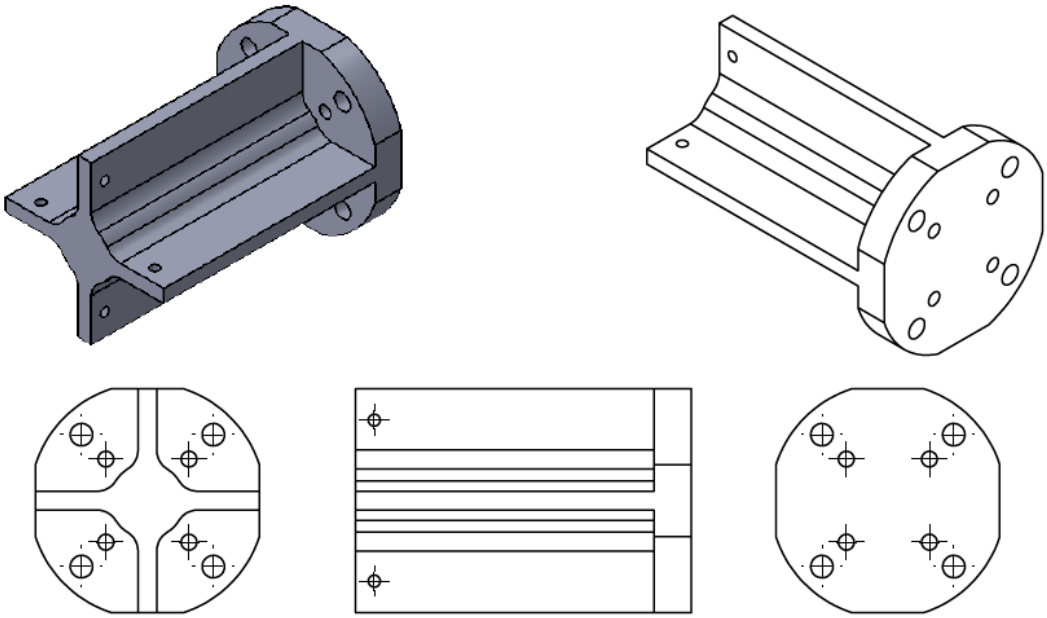


Şekil 3.3: Arduino Mega 2560 Kartına ait bir teknik resim [39].

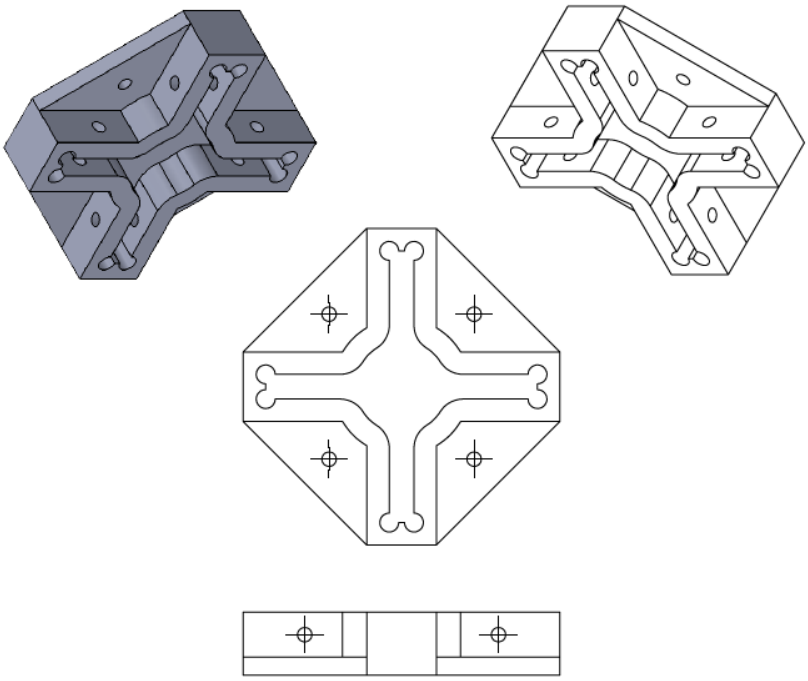


Şekil 3.4: Arduino Pro Mini Kartına ait bir teknik resim [40].

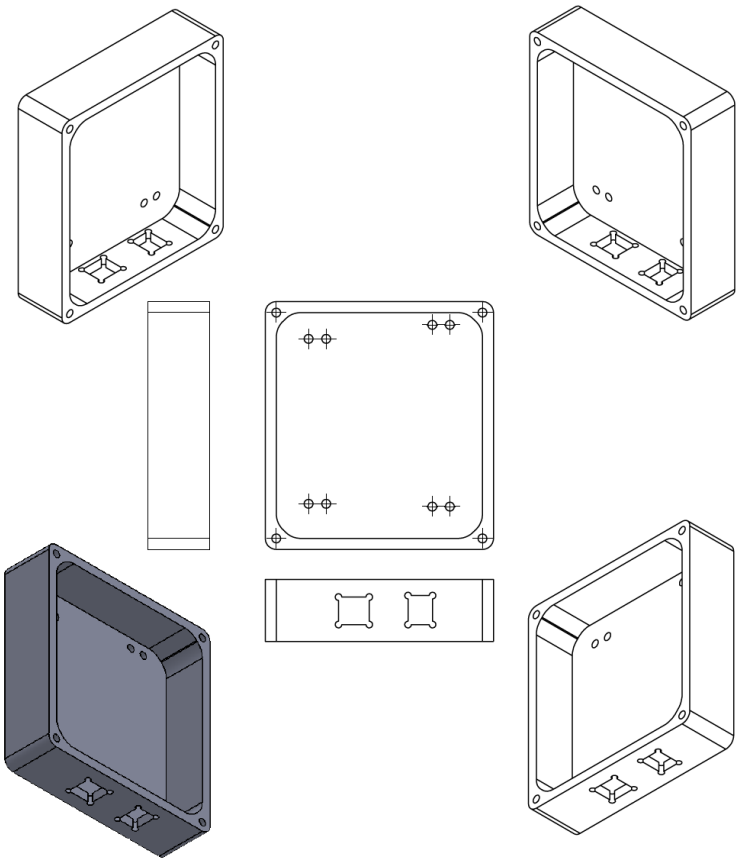
Tasarım yapılırken, aparatların montajlarının ve imalatlarının kolay olmasına dikkat edilmiştir. Bu sebeple montaj aparatı dört farklı parça olacak şekilde tasarlanmıştır. Bunlar flanş aparatı, sabitleyici aparat, üst kutu aparatı ve alt kutu aparatıdır. Bu aparatlara ait görselleri Şekil 3.5 - 3.8 arasında, aparatların montajlı hali de Şekil 3.9'da incelenebilmektedir.



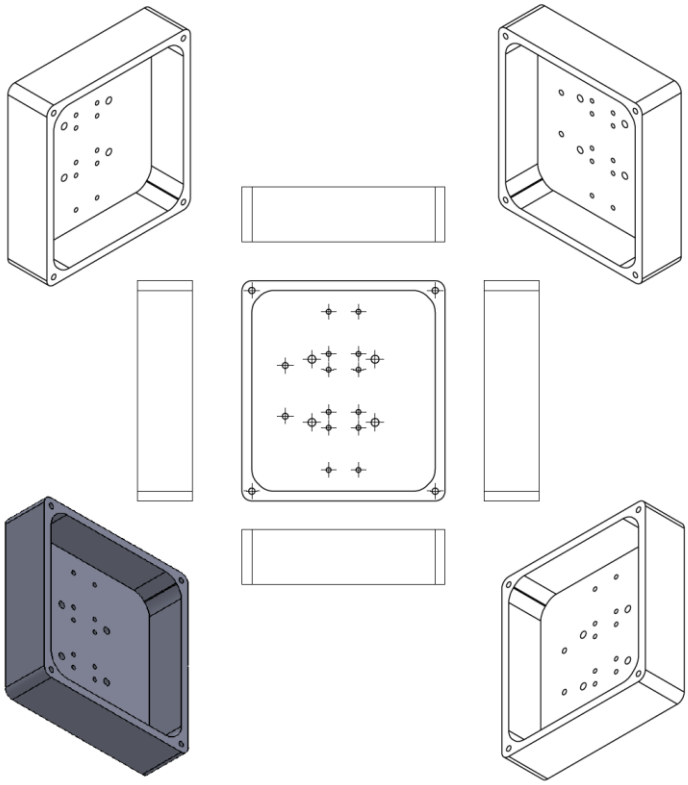
Şekil 3.5: Flanş aparatı.



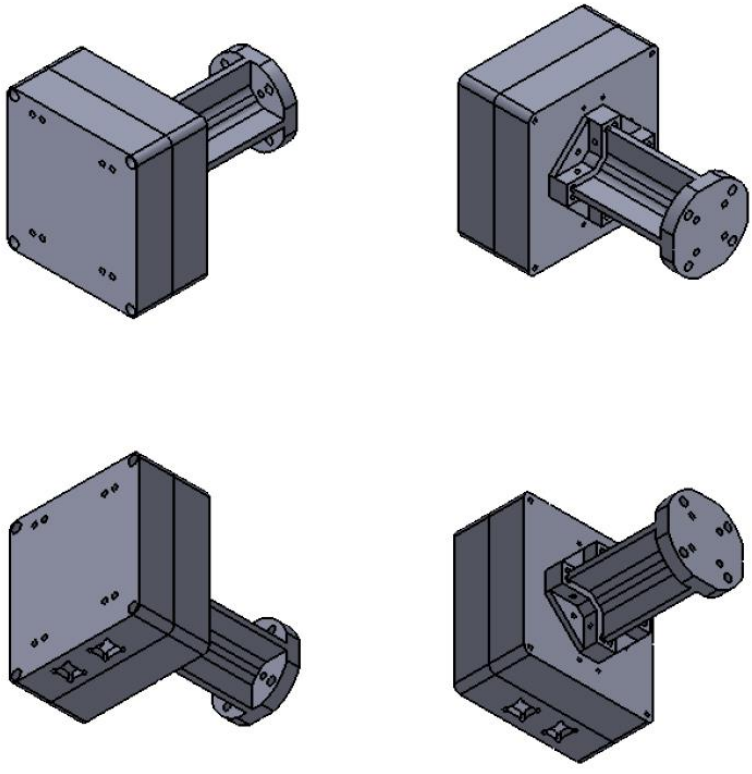
Şekil 3.6: Sabitleyici aparat.



Şekil 3.7: Üst kutu aparatı.

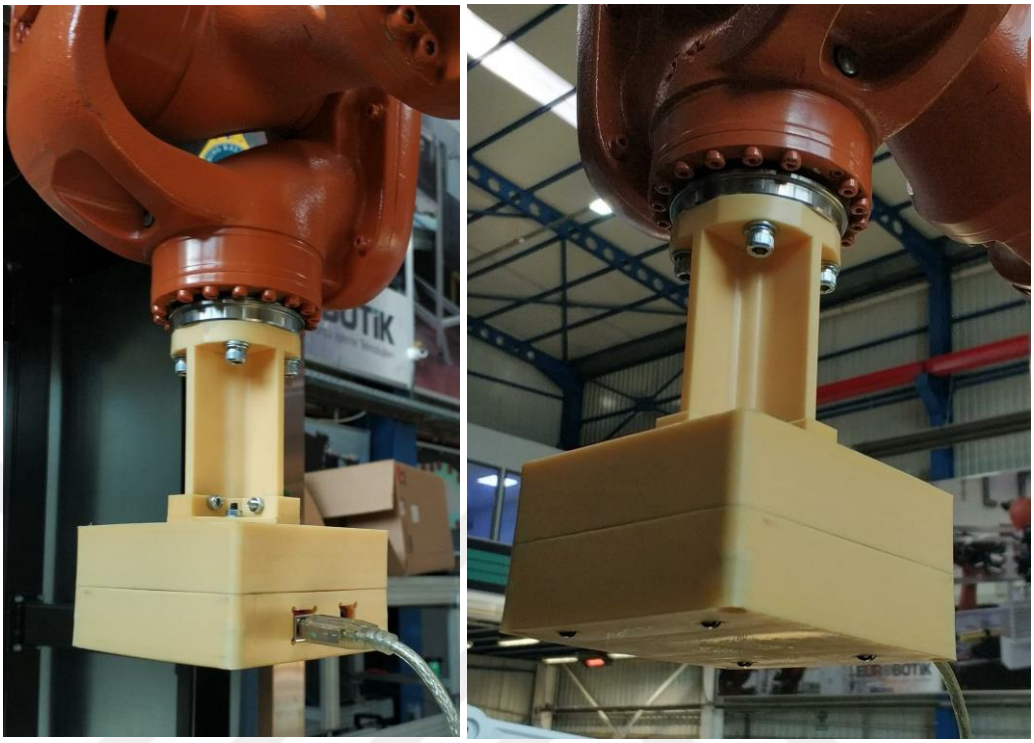


Şekil 3.8: Alt kutu aparatı.



Şekil 3.9: Bağlantı elemanı.

Bağlantı elemanı, CNC makinede talaşlı imalat ile kestamit malzeme kullanılarak imal edilmiştir. Bağlantı elemanının toplanıp robot flanşına montajlanmış hali Şekil 3.10'da görülebilmektedir.

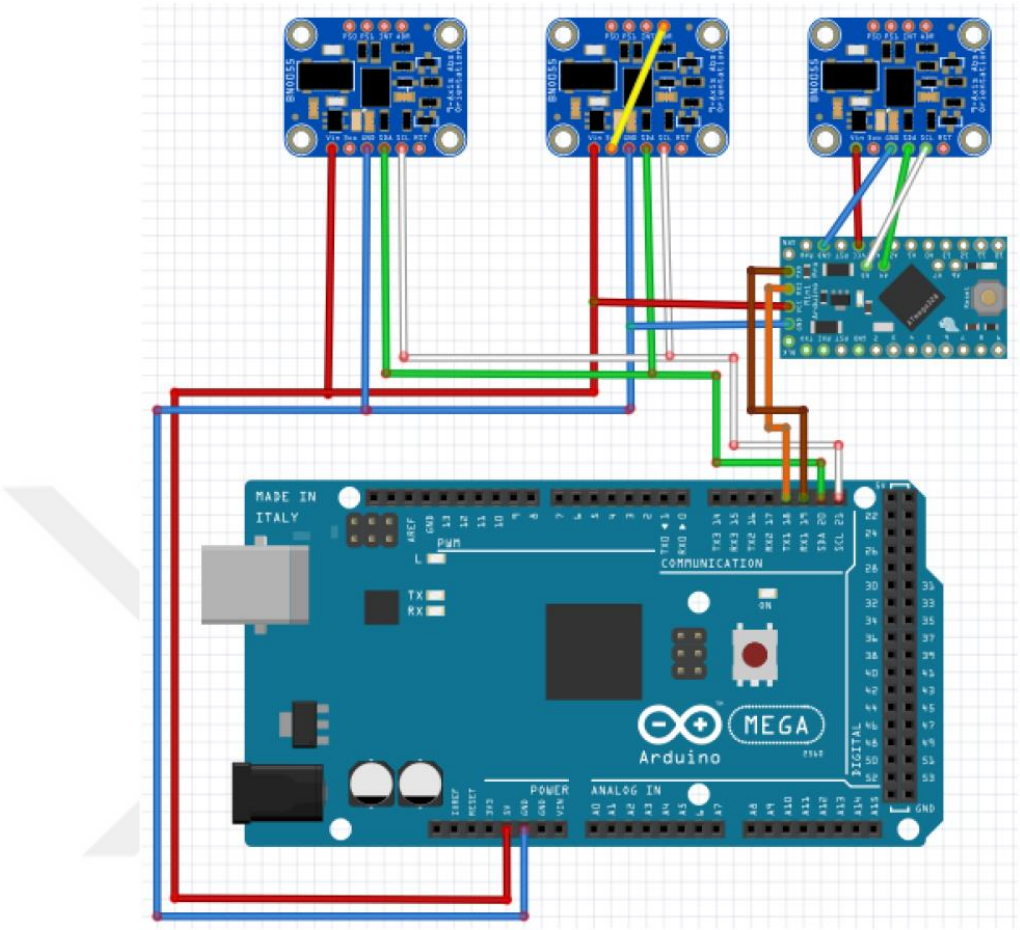


Şekil 3.10: Aparatların toplanıp robot kolu flanşına montajlanmış hali.

3.2 Sensörlerin ve Mikrodenetleyici Kartlarının Elektronik Bağlantıları

AÖB sensörlerinden veri çekilebilmesi için bir kontrolcüye bağlanmaları gerekmektedir. Bu sebeple AÖB kartlarının iki tanesi arduino mega 2560 kartına bir tanesi de arduino pro mini kartına balğanmıştır. Bir kartın ayrı bağlanmasının sebebi, AÖB kartlarının I^2C haberleşme protokolünün sadece iki farklı adresten destekleniyor olmasından kaynaklanmaktadır. Arduino mega 2560 kartının SDA SCL hattı, iki AÖB sensörünün SDA SCL hattına bağlanmıştır. Beslemeleri de Arduino Mega 2560 kartının 5V,GND çıkışlarından sağlanmıştır. Arduino Mega 2560 kartının beslemesi usb kablosu üzerinden yapılmıştır. Arduino Pro Mini kartının SDA SCL hattı ise, 3. AÖB sensörünün SDA SCL hattına bağlanmıştır ve beslemesi yine aynı kartın 5V , GND çıkışından verilmiştir. Arduino pro mini kartının 3. AÖB sensöründen aldığı verileri Arduino mega 2560 kartına aktarabilmesi için iki kartın TX,RX hatları çapraz olarak bağlanmıştır. Arduino pro mini kartının beslemesi arduino mega 2560 kartı üzerinden verilmiştir. Son olarak 2. AÖB kartının I^2C adresi 1. AÖB kartndan farklı

olması için ADR pinine 3.3V bağlanmıştır. Tüm bağlantıların şemasını Şekil 3.11'de incelemek mümkündür.



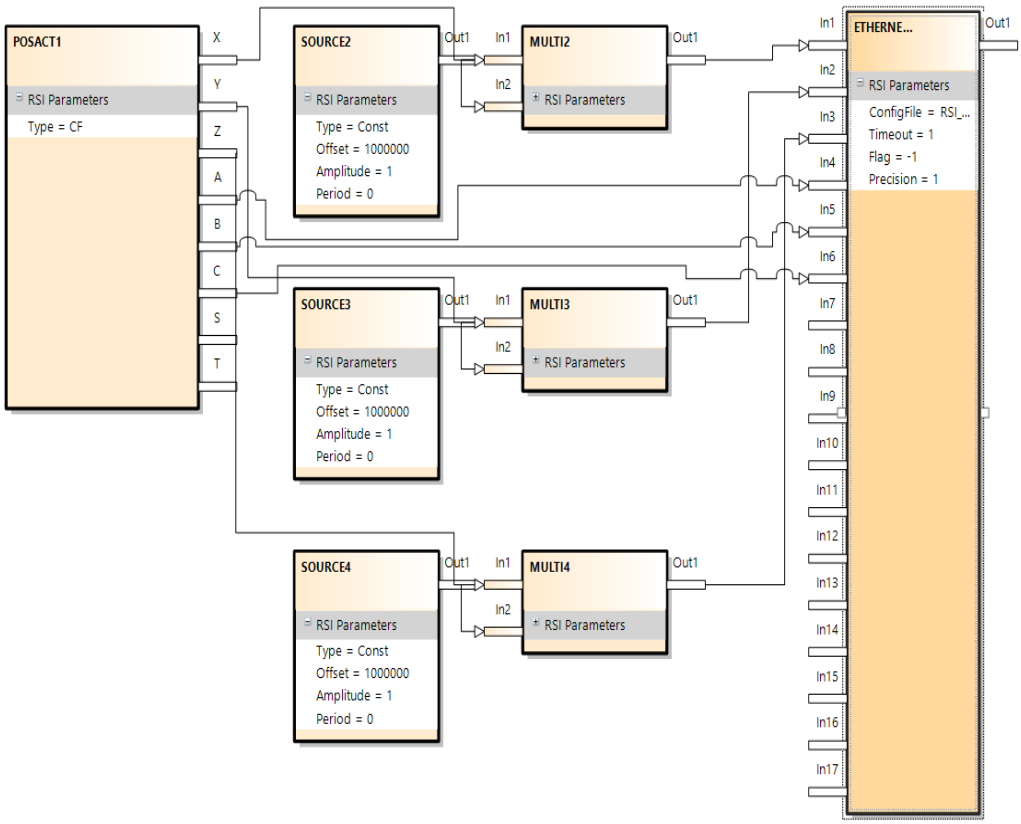
Şekil 3.11: Bağlantı Şeması.

3.3 Endüstriyel Robottan ve Sensorlerden Veri Gönderilmesi

Kuka Kr 16 R1610 model robot kolundan dışarıya, flanş konumu bilgisini aktarmanın birden fazla yolu bulunmaktadır. İlk akla gelenler, EthernetKRL opsiyonunu kullanarak robot kolunun kontrolcüsünden SPS (PLC kısaltmanın almanca karşılığı) programı ile TCP-IP protokolü ile sürekli yollamaktır. Bir diğer yöntem de, endüstriyel haberleşme protokolü olan ethercat protokolü ile yine SPS programı üzerinden ara bir kontrolcüye yani PLC'ye yollamak, karşı bilgisayarın da bu, PLC'ye gelen veriyi ADS.NET yada OPC server ile sürekli çekmesidir. Bu iki aktarma yolu da SPS'ye bağlı oldukları için veri gönderim hızı yavaş olan tekniklerdir. Sebebi Kuka robotlarında SPS çevirim süresi 12 ms'dir. Bu iki senaryoda ihtiyaçlara göre kullanılabilse de, bu çalışma için yeterli değildir. Bu sıkıntıya çözüm olarak Kuka firması RSI (Robot Sensör Interface) opsiyonunu geliştirmiştir. Bu opsiyonun özelliği

çevrim süresini 4 ms'ye kadar indirebilmesidir. Yörünge düzeltme uygulamalarında da bu teknik kullanılmaktadır. Bu tez kapsamında, RSI opsiyonu kullanılarak robot flanşına ait konum (X,Y,Z) bilgileri ve rotasyon (A,B,C) bilgileri UDP protokolü ile 4 ms aralıkla gönderirmiştir.

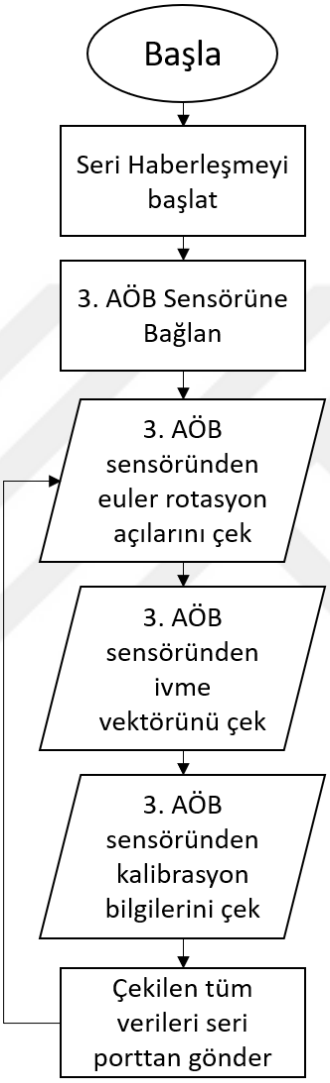
RSI opsiyonundan veri aktarmak için birkaç konfigurasyon yapılması gerekmektedir. Bunlardan biri RSIVisualShell programını kullanarak, Blok diagramla veri gönderim şeklini tanımlamak gerekmektedir. Şekil 3.12'de görülebileceği gibi X,Y ve Z verileri 1000000 sayısı ile çarpılarak yollanmıştır. Sebebi ise, gelen verinin virgülden sonra sadece iki hane olmasıdır. 1000000 ile çarpılarak gelen verinin hassasiyeti arttırılmıştır sonra hazırlanan bu diagrama ait XML çıktısı robota yüklenerek konfigurasyon işlemi tamamlanmıştır. Diğer konfigurasyon işlemi ise robot kontrolcüsünde sanal bir UDP soketi oluşturmaktır.



Şekil 3.12: RSIVisualShell de hazırlanan blok diagram.

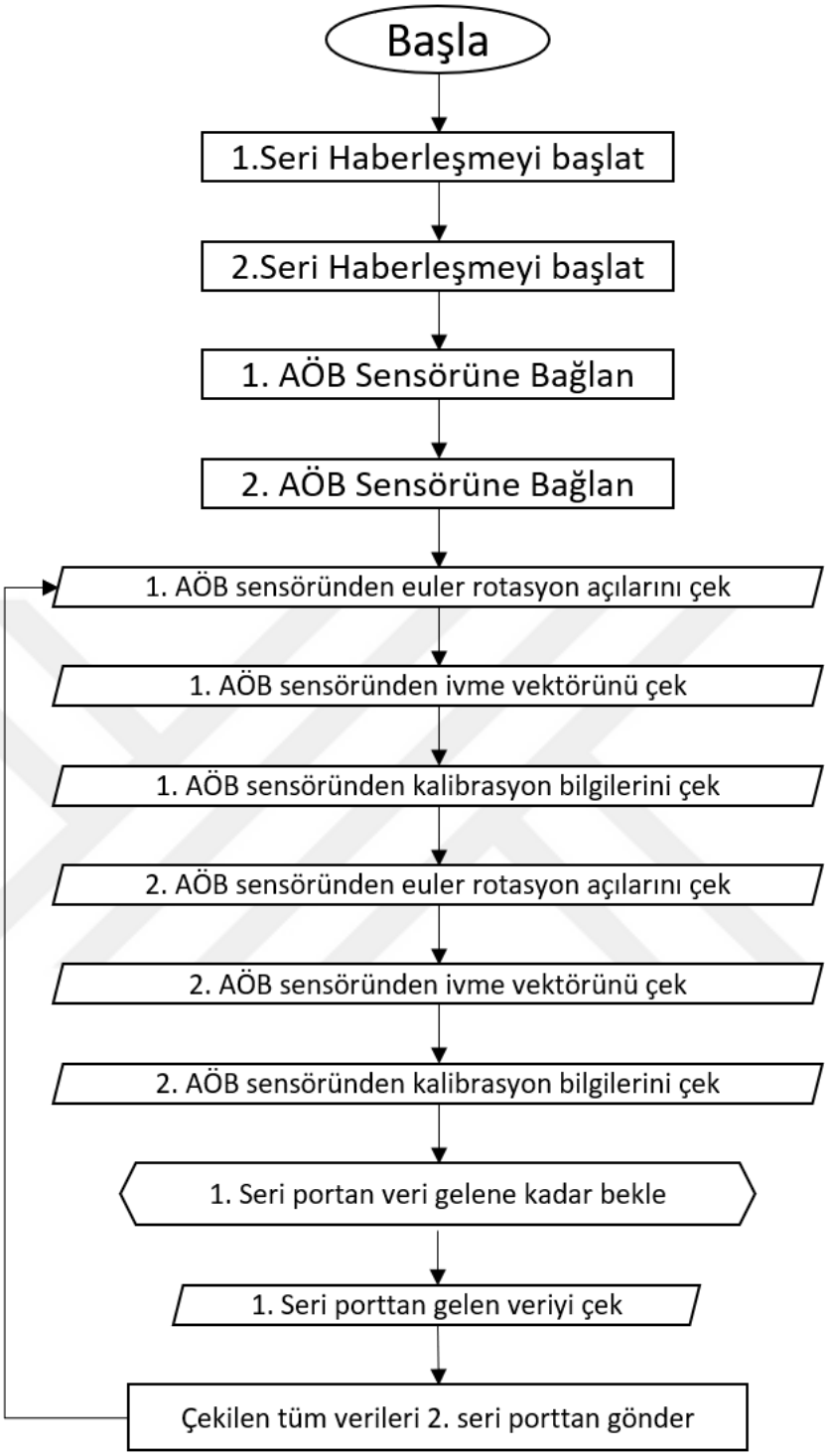
Adafruit BNO055 AÖB sensörlerinden veri çekebilmek için, Arduino Mega 2560 kartı ve Arduino Pro Mini kartları kullanılmıştır. Bu kartlara veriyi çekme ve aktarma

programları Arduino firmasının ücretsiz yayınladığı Arduino IDE programı kullanılarak oluşturulmuştur. Öncelikle Arduino Pro Mini kartına 3.AÖB sensöründen veriyi I^2C protokolü ile çekip RS-232 protokolü ile yollaması için bir kod yazılmıştır ve FTDI programlayıcısı ile bu kod Arduino Pro Mini kartına yüklenmiştir. Arduino Pro Mini kartına yüklenen kodun akış diyagramı Şekil 3.13'te gösterilmiştir.



Şekil 3.13: Arduino Pro Mini kartına yüklenen programın akış diyagramı.

Arduino Mega 2560 kartına 1. ve 2. AÖB sensörlerinden verilerin I^2C protokolü ile çekilmesi ve RS-232 protokolü ile arduino pro mini kartından 3. AÖB sensörüne ait verilerin çekilmesi ve bu çekilen tüm verilerin başka bir RS-232 portundan gönderilmesi için kod yazılmıştır. Kodun akış diyagramını Şekil 3.14'te görülebilmektedir.



Şekil 3.14: Arduino Mega 2560 kartına yüklenen programın akış diyagramı.

Yazılan bu kod ile birlikte AÖB sensörlerinden veri çekme ortamı hazır hale gelmiştir. Sensörlerin 100 Hz frekansında çalışması ve haberleşmelerde yaşanan gecikmeler ile birlikte ortalama 15 ms'lik bir çevrim süresinde sensörlerin veri yollanması da sağlanmıştır.

3.4 Endüstriyel Robottan ve Sensörlerden Gelen Verilerin Senkronize Edilerek Toplanması

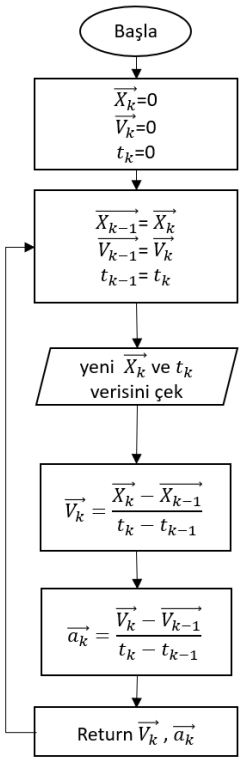
Endüstriyel robot kolundan çekilen konum verilerinden anlık hız ve ivme verilerini hesaplamakla işe başlanmıştır. Hesaplamalarda 3.1'de belirtilen hız formülü ve 3.2'de belirtilen ivme formülü kullanılmıştır.

$$\overrightarrow{V_k} = \frac{\overrightarrow{X_k} - \overrightarrow{X_{k-1}}}{t_k - t_{k-1}} \tag{3.1}$$

$$\overrightarrow{a_k} = \frac{\overrightarrow{V_k} - \overrightarrow{V_{k-1}}}{t_k - t_{k-1}} \tag{3.2}$$

$\overrightarrow{X_k}$, t_k zamanında gelen konum bilgisi, $\overrightarrow{X_{k-1}}$ ise bir önceki t_{k-1} zamanında gelen konum bilgisini belirtmektedir. Konumdaki değişimin zamana bölümü ise t_k anındaki hız bilgisini ($\overrightarrow{V_k}$) vermektedir. Aynı şekilde hızdaki değişimin zamana bölümü de ivme ($\overrightarrow{a_k}$) bilgisini vermektedir. Ölçüm aralıkları 4 ms olduğu için hız ve ivme bilgileri hassas olarak hesaplanmıştır.

Verileri çekme ve kaydetme programı Python programlama diliyle yazılmıştır. Konum verilerinden hız ve ivme verisini hesaplamak için hazırladığımız fonksiyonun akış diyagramını Şekil 3.15'te görülebilmektedir.



Şekil 3.15: Hız ve ivme hesaplama fonksiyonuna ait akış diyagramı.

Bu tez için öncelikle doğru işaretli veri seti oluşturulması çok önemlidir. Tezin amacı, AÖB sensörlerinden gelen ivme verilerini iyileştirmek olduğu için, endüstriyel robot kolundan gelen konum verileri ile hesaplanan ivme verileri doğru kabul edilmektedir ve AÖB sensörlerinden gelen ivme verileri doğru kabul edilen robot verileri ile eşleştirilerek kaydedilmiştir. Bu eşleştirme işleminin doğru olması için veriler arası senkronizasyon çok önemlidir. Bu tez kapsamında, robot kolu konum verisi 4 ms aralıkla, AÖB sensörü ivme verisi 15 ms aralıkla gelmektedir. Senkronizasyon için üç farklı yöntem denenmiş ve sadece birinde başarılı olunmuştur.

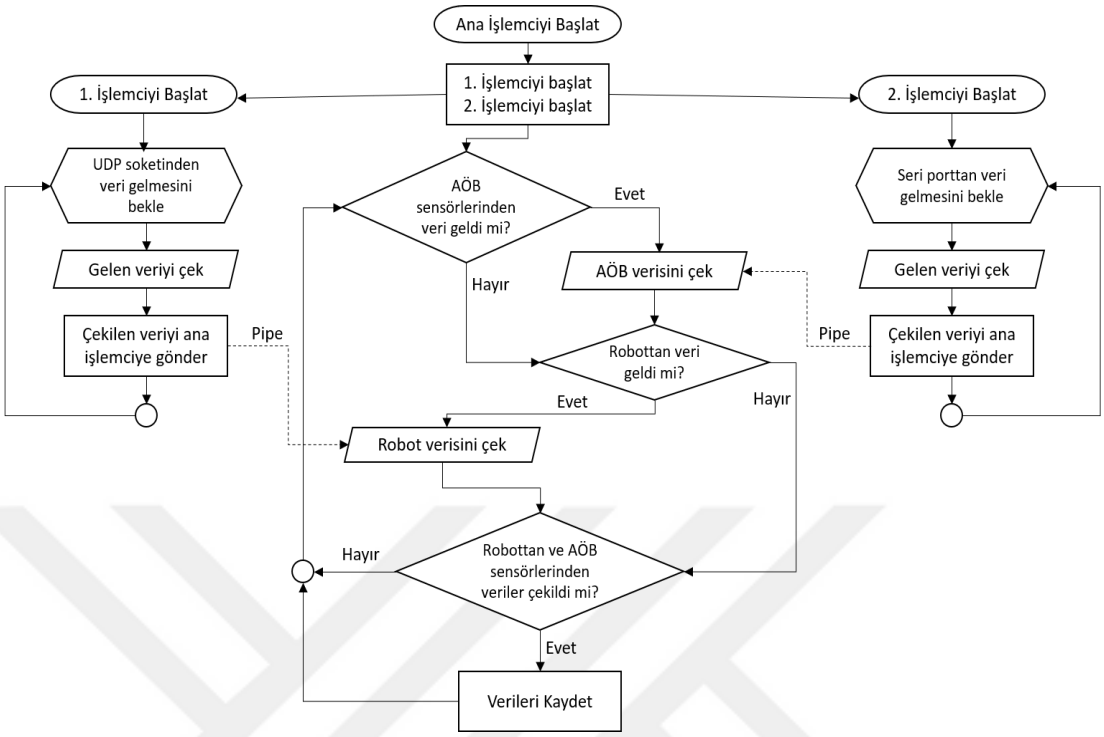
İlk denenen yöntemde, program yazılırken programın AÖB verisi gelene kadar beklemesi sağlanmış, geldikten sonra robot verisini çekmesi amaçlanmıştır. Bu yöntemin başarısız olmasının sebebi, robot kontrolcüsü bufferdaki paketlerin tek tek yollamıştır. Bufferda biriken tüm veri tek seferde çekilememiş bundan dolayı robot verilerinde gecikmeye sebep olmuştur.

Denemesi yapılan ikinci yöntemde ise, AÖB verilerinin çekildiği seri portun timeout parametresinin 4 ms'ye indirilmesidir. Bu sayede robot verileri düzgün bir şekilde çekilirken, AÖB verileri parça parça çekilip birleştilirerek anlamlı hale getirilmesi planlanmış ama başarısız olunmuştur. Sebebi, veri paketinin parçalanmasının tamamen rastgele bir şekilde olmasından kaynaklanmaktadır. Daha sonra birleştirip anlamlı hale getirecek bir fonksiyon yazılamamıştır.

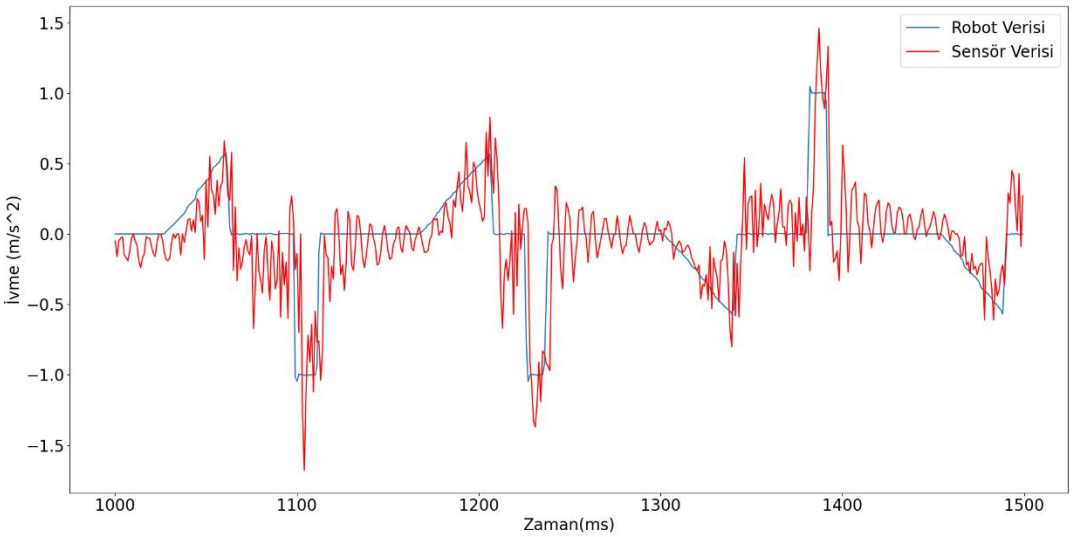
Son denemede çoklu işleme (multiprocessing) yöntemi kullanılmıştır. Çoklu işleme yöntemi, iki ya da daha fazla işlemcinin bir araya getirilerek bir görevi daha hızlı yerine getirmesi düşünülerek tasarlanmıştır. Veri çekmek için bir ana işlemci, iki tane de yan işlemci olmak üzere üç işlemcili bir program yazılmıştır. Ana işlemci, iki yan işlemciyi başlatıp onlardan veri geldikçe eşleştirerek kaydetmektedir. 1. yan işlemci durmaksızın robot kontrolcüsünden veri çekmekte, veri geldikçe ana işlemciye aktarmaktadır. 2. yan işlemci ise durmaksızın AÖB verilerini çekmekte ve veri geldikçe ana işlemciye aktarmaktadır. Bu denemenin veri senkronizasyonunda başarılı olduğu görülmüştür ve tez kapsamında bu denemede yazılan program kullanılmıştır. Programa ait akış şeması Şekil 3.16'da görülebilmektedir.

Çoklu işlemci tekniği ile yazılan program ile kaydedilen verileri grafik olarak Şekil 3.17 - 3.25 arası incelenebilmektedir. Mavi çizgi ile çizilen grafikler robottan gelen konum verisine göre hesaplanan ivme grafikleridir, kırmızı ile çizilenler ise AÖB sensörlerinden gelen ivme verilerini göstermektedir. Grafikler incelendiği zaman

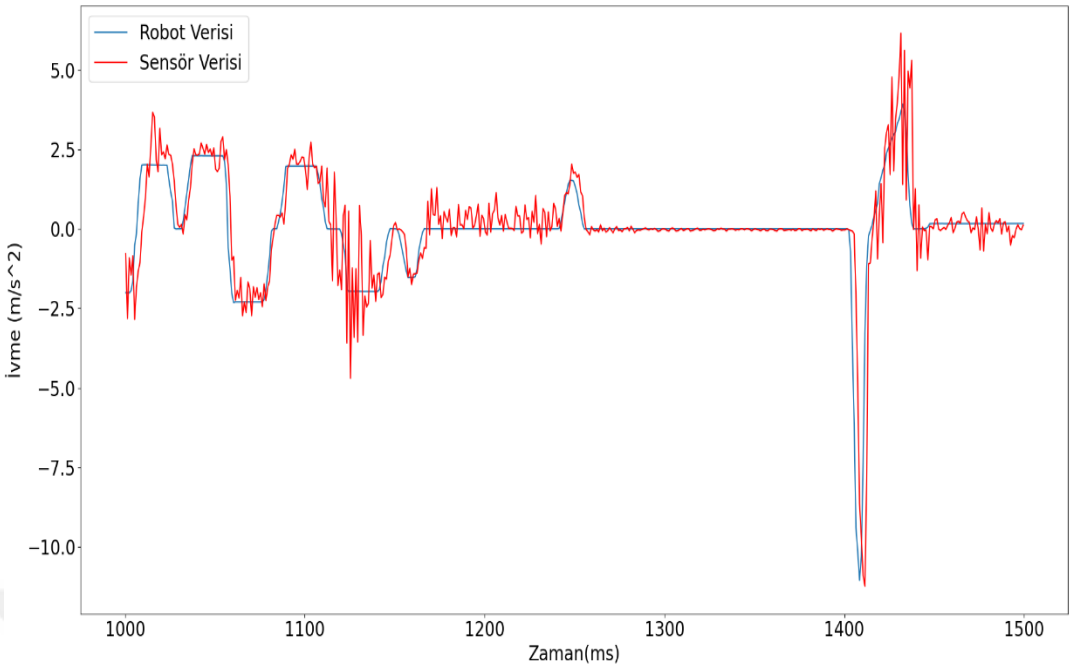
belirgin ivme artışlarının eş zamanlı olduğu görülmektedir. Bunun yanında sensörlerde gürültü olduğu da anlaşılmaktadır.



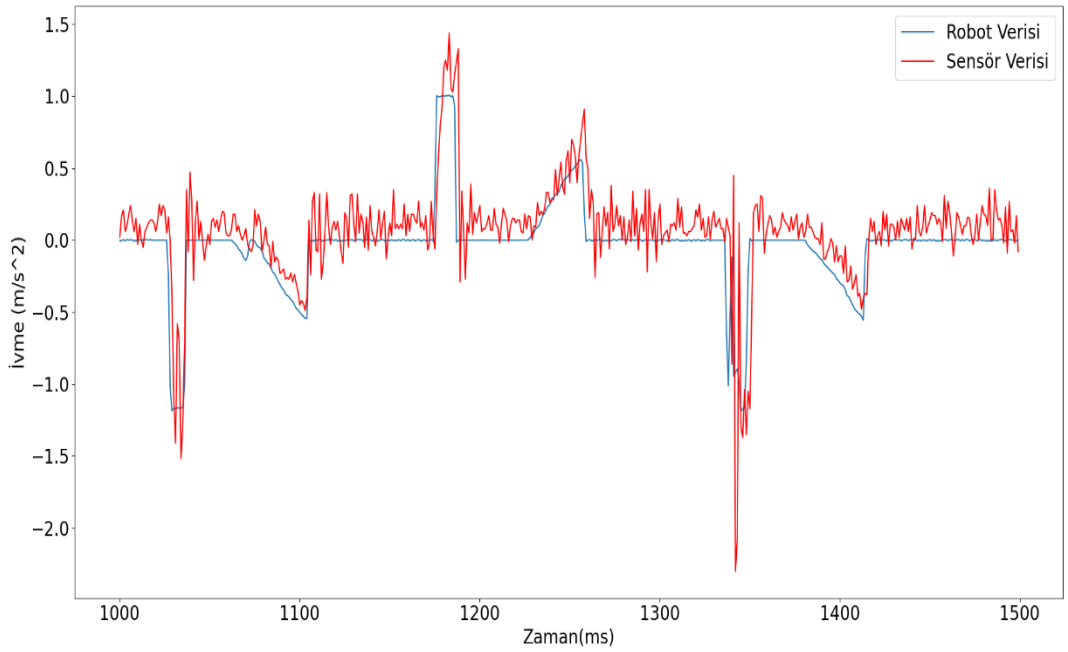
Şekil 3.16: Çoklu işlemci yöntemiyle hazırlanan veri kaydetme programına ait akış diyagramı.



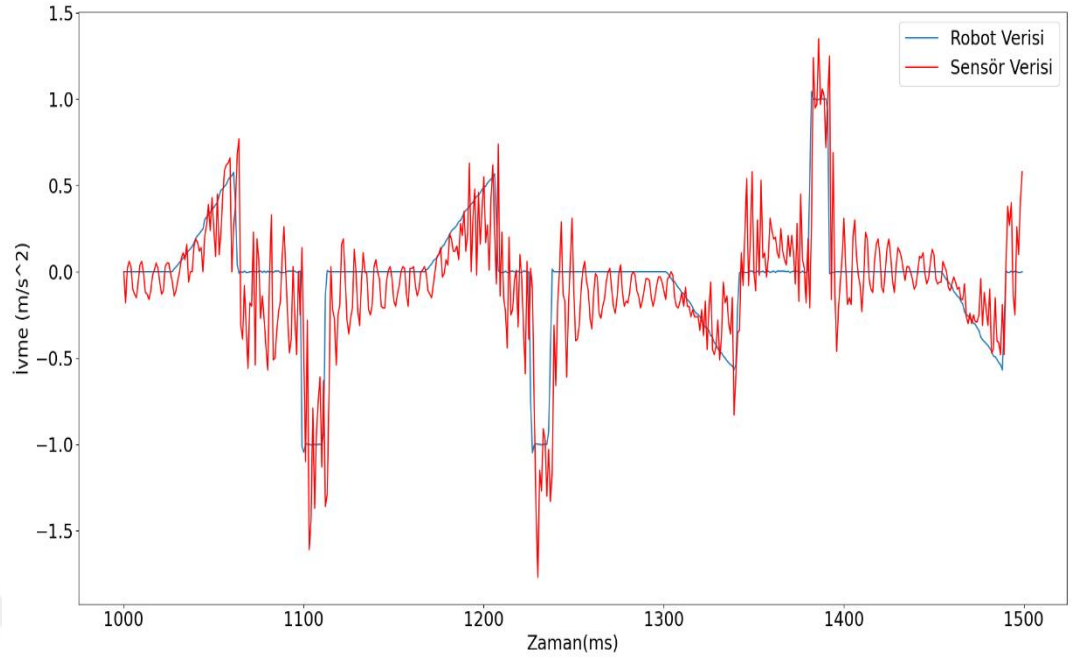
Şekil 3.17: 1. AÖB sensörüne ait X yönlü ivme verisi ve robot kolu flanşına ait ivme verisi.



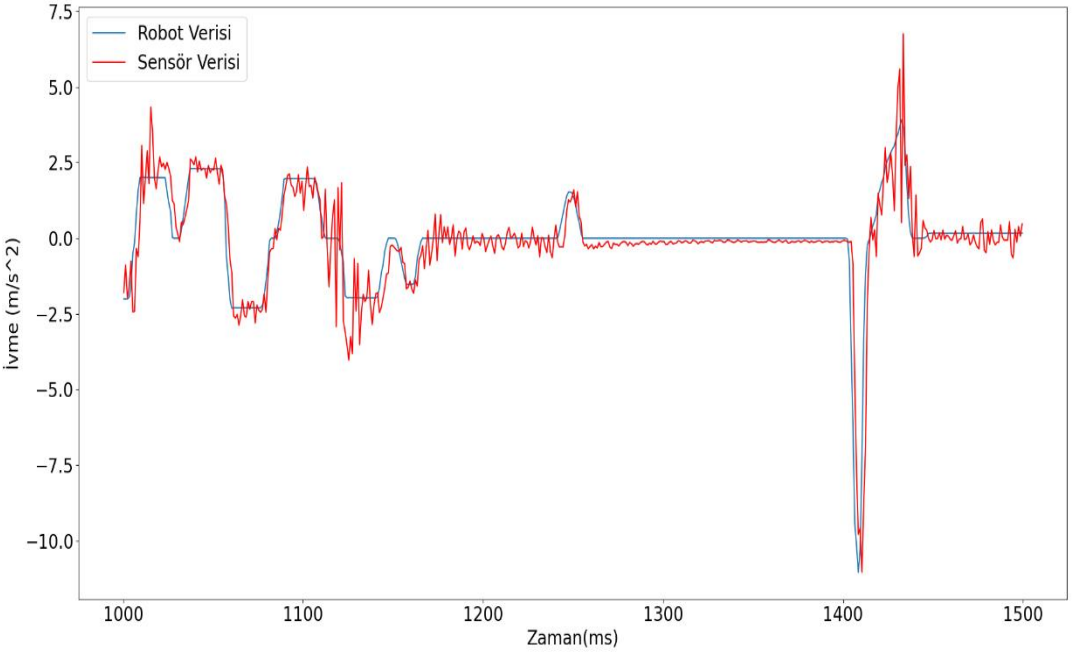
Şekil 3.18: 1. AÖB sensörüne ait Y yönlü ivme verisi ve robot kolu flanşına ait ivme verisi.



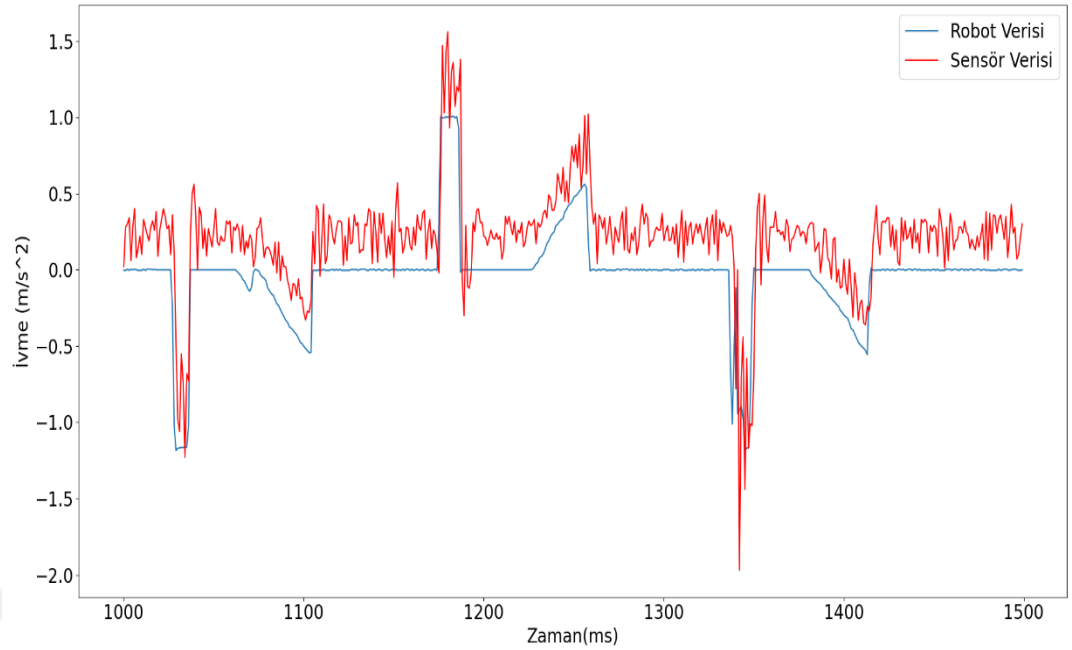
Şekil 3.19: 1. AÖB sensörüne ait Z yönlü ivme verisi ve robot kolu flanşına ait ivme verisi.



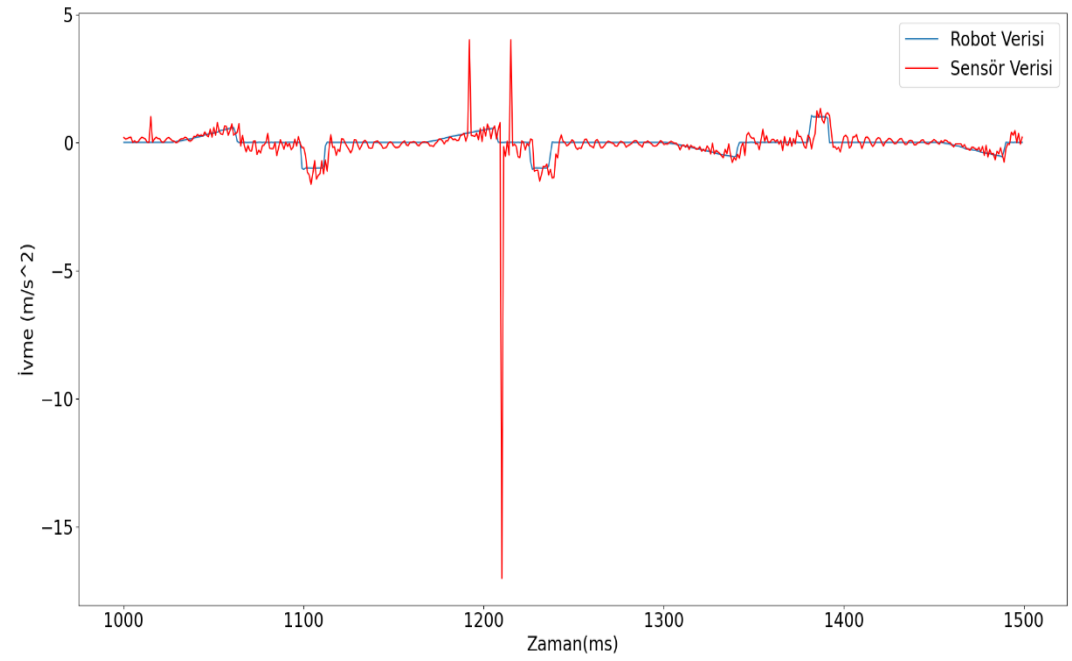
Şekil 3.20: 2. AÖB sensörüne ait X yönlü ivme verisi ve robot kolu flanşına ait ivme verisi.



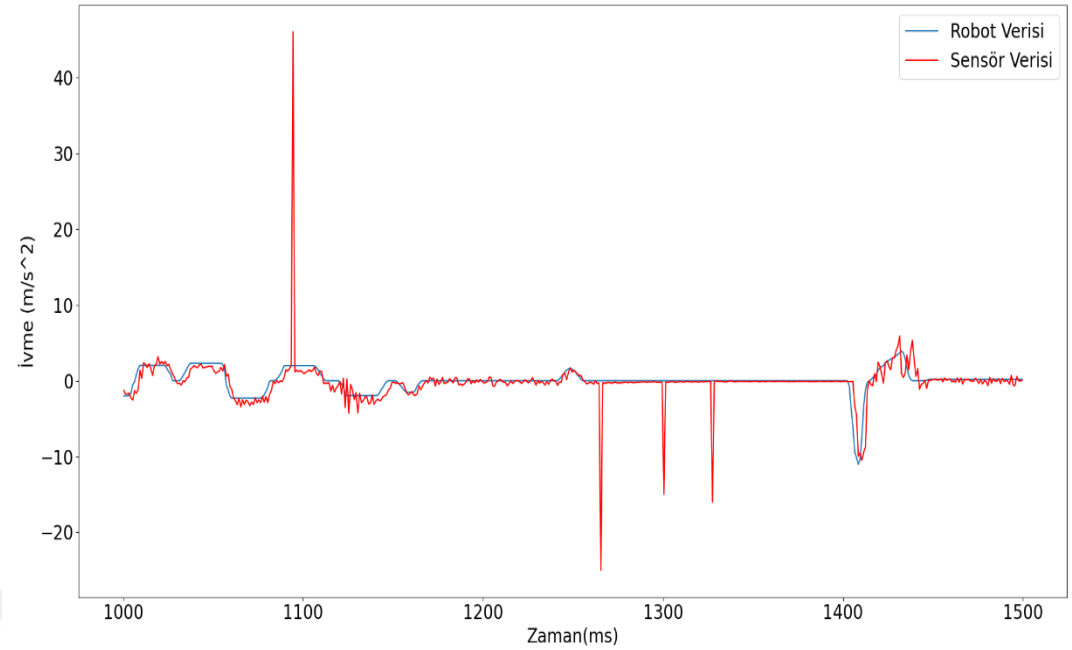
Şekil 3.21: 2. AÖB sensörüne ait Y yönlü ivme verisi ve robot kolu flanşına ait ivme verisi.



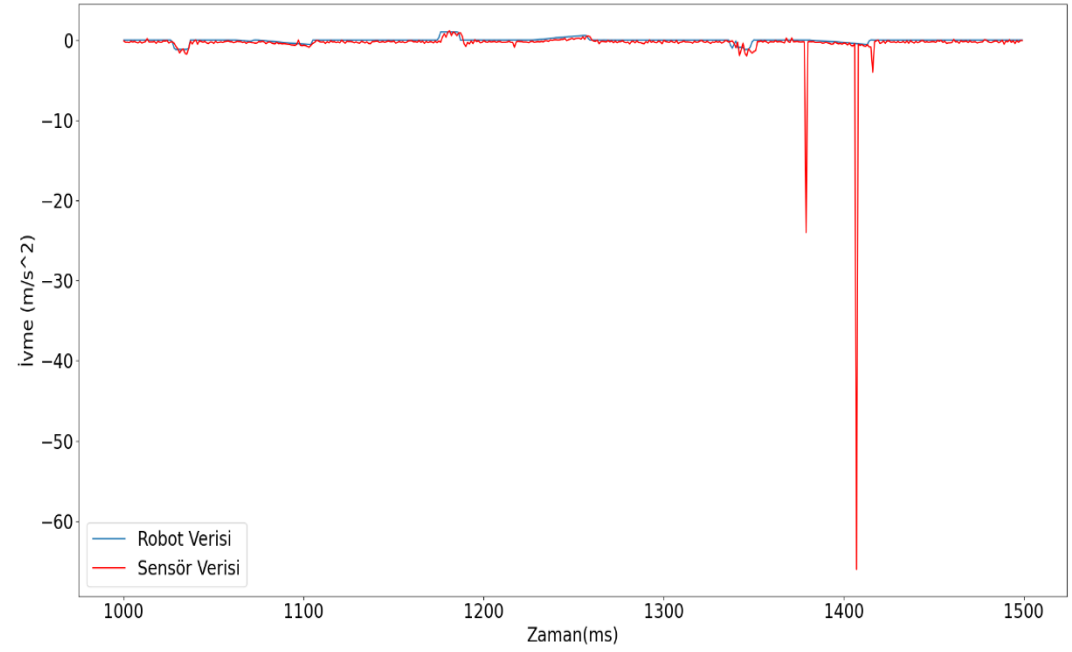
Şekil 3.22: 2. AÖB sensörüne ait Z yönlü ivme verisi ve robot kolu flanşına ait ivme verisi.



Şekil 3.23: 3. AÖB sensörüne ait X yönlü ivme verisi ve robot kolu flanşına ait ivme verisi.



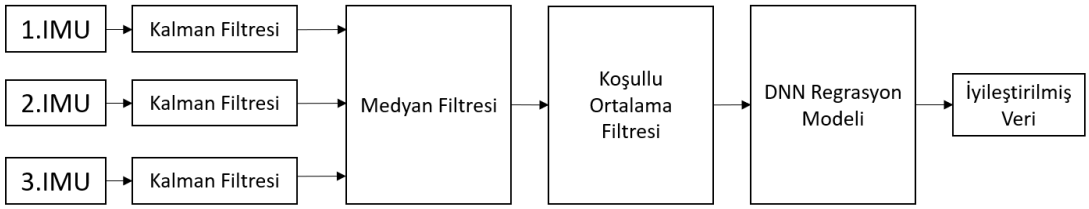
Şekil 3.24: 3. AÖB sensörüne ait Y yönlü ivme verisi ve robot kolu flanşına ait ivme verisi.



Şekil 3.25: 3. AÖB sensörüne ait Z yönlü ivme verisi ve robot kolu flanşına ait ivme verisi.

4. İVME VERİSİNİN İYİLEŞTİRİLMESİ

İvme verilerinin iyileştirilmesinin iki aşamalı olarak yapılması planlanmıştır. Şekil 4.1'deki akış diyagramında gösterildiği gibi önce AÖB sensörlerinden gelen ham verilere kalman filtresi uygulanmış daha sonra sırayla üç sensörden gelen veri önce medyan filtreden daha sonra koşullu ortalama filtreden geçirilmiştir, sonra ikinci aşamada filtrelenmiş bu verilerle makine öğrenmesi yaklaşımıyla iyileştirme modelleri eğitilmiştir.



Şekil 4.1: İvme verisi iyileştirme akış diyagramı.

4.1 AÖB Sensörlerinden Ölçülen İvme Verilerine Filtre Uygulanması

AÖB sensöründen ölçülen ivme verileri, robot kolundan gelen ivme verileri ile senkronize edilerek kaydedildikten sonra, AÖB sensöründen gelen ivme verilerinde istenmeyen gürültüler gözlenmiştir. Bu gürültüleri, Şekil 3.17-3.25'te görmek mümkündür. Bu gürültüleri temizlemek için Kalman filtresi, Medyan filtresi ve Ortalama filtresi kullanılmıştır.

4.1.1 Kalman filtresinin uygulanması

Kalman filtresi, sistem çıktısının belirli bir Kalman kazancı katsayısı ile çarpılarak tekrar sisteme girdi olarak veren bir sayısal filtreleme yöntemidir. Kalman filtresi, matematiksel modeli bilinen bir sistemde iyileştirme, sinyaldeki gürültüleri temizleme ve ölçüm aralıkları uzun olan veri girişlerinde zamana dayalı tahminler yapmada kullanılmaktadır. Bu tez kapsamında ise ivme verisindeki gürültüleri temizlemek amacıyla kullanılmıştır.

Kullanılan Kalman fitresinin matematiksel modelini 4.3 eşitliğinde incelemek mümkündür [27].

$$\hat{X}_k = \hat{X}_{k-1} + K_k(Z_k - \hat{X}_{k-1}) \tag{4.3}$$

4.3 eşitliğinde görülen $\hat{X}_k$ verinin kalman filtresinde çıkmış halini temsil etmektedir. K_k kalman kazancı katsayısıdır. Z_k sensörden ölçülen veridir.

$$K_k = \frac{P_{k-1}}{P_{k-1}+R} \tag{4.4}$$

$$P_k = (1 - K_k)P_{k-1} \tag{4.5}$$

$$P_{k-1} = P_k + P_0 \tag{4.6}$$

4.4 eşitliğinde R, filtrelenmek istenen sinyale ait varyans değeridir. P_0 ise filtrenin tepki süresini düzenlemek ve tepki süresi ile gürültü arasında bir optimizasyon yapmak için eklenmiş bir parametredir.

$$R = \frac{\sum(X_i - \bar{X})^2}{n} \tag{4.7}$$

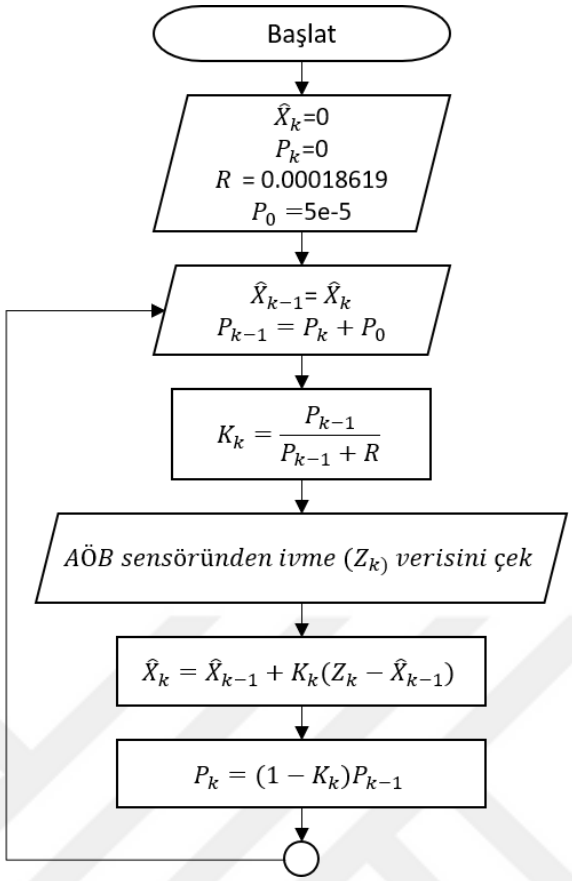
Varyans, verilerin aritmetik ortalamadan sapmalarının karelerinin toplamının veri sayısına bölümüdür. $\bar{X}$,verilerin aritmetik ortalamasını temsil etmektedir. n ise veri sayısını temsil etmektedir.

Varyans hesaplaması için AÖB sensörleri hiç hareket ettirilmeden X, Y ve Z yönlü ivme verilerinden 1000'er örnek alınmıştır ve bu veri setleriyle 4.7'deki varyans formülü kullanılarak, varyanslar hesaplanmıştır. Hesaplanan varyans değerleri Çizelge 4.1'de görülmektedir.

Çizelge 4.1: AÖB sensörlerine ait hesaplanan varyans değerleri.

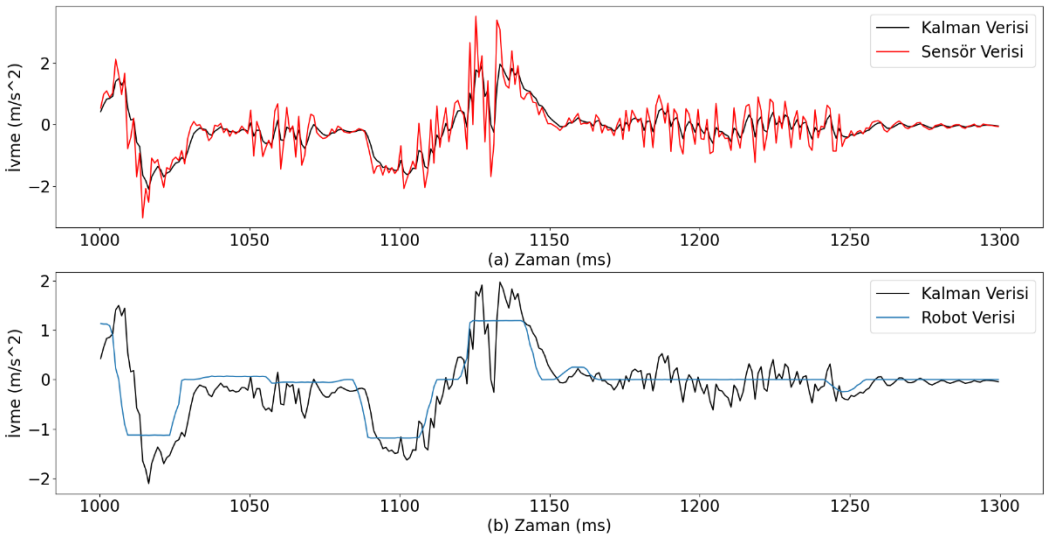
SENSÖR	X	Y	Z
1.AÖB SENSÖRÜ	0.00018619	0.00016604	0.00216906
2.AÖB SENSÖRÜ	0.00017222	0.00011852	0.00317561
3.AÖB SENSÖRÜ	0.06415628	0.92800215	1.74460271

Kalman filtresi, daha önce hazırlanan veri toplama programına fonsiyon olarak eklenmiştir. Kalman filtresi için hazırlanan fonksiyonun akış diyagramını Şekil 4.2'de görmek mümkündür.

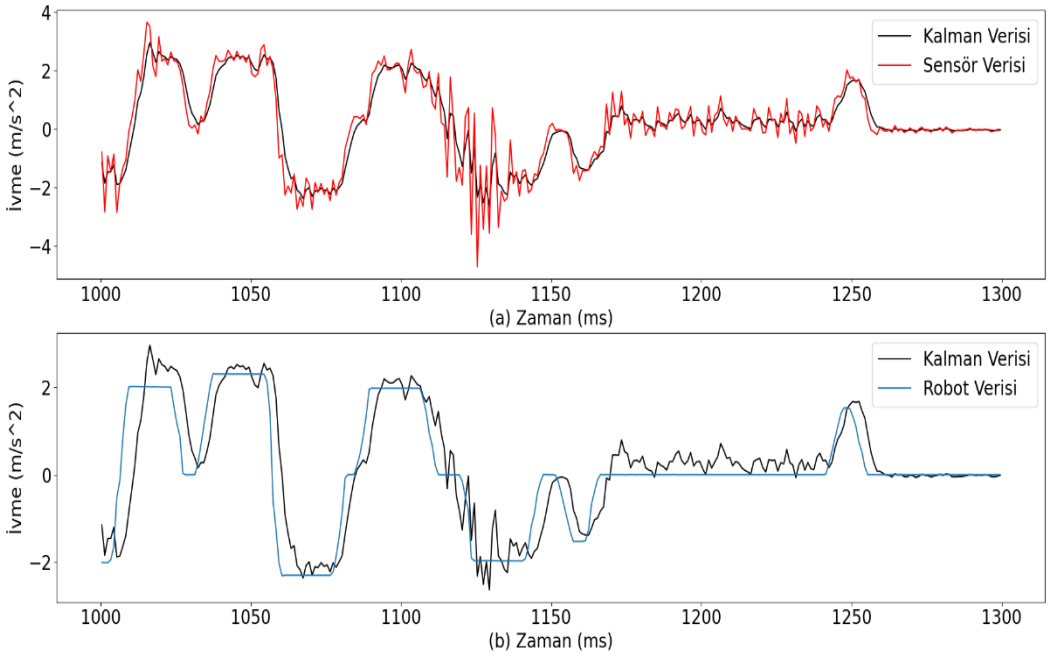


Şekil 4.2: Kalman filtresi fonksiyonunun akış diyagramı.

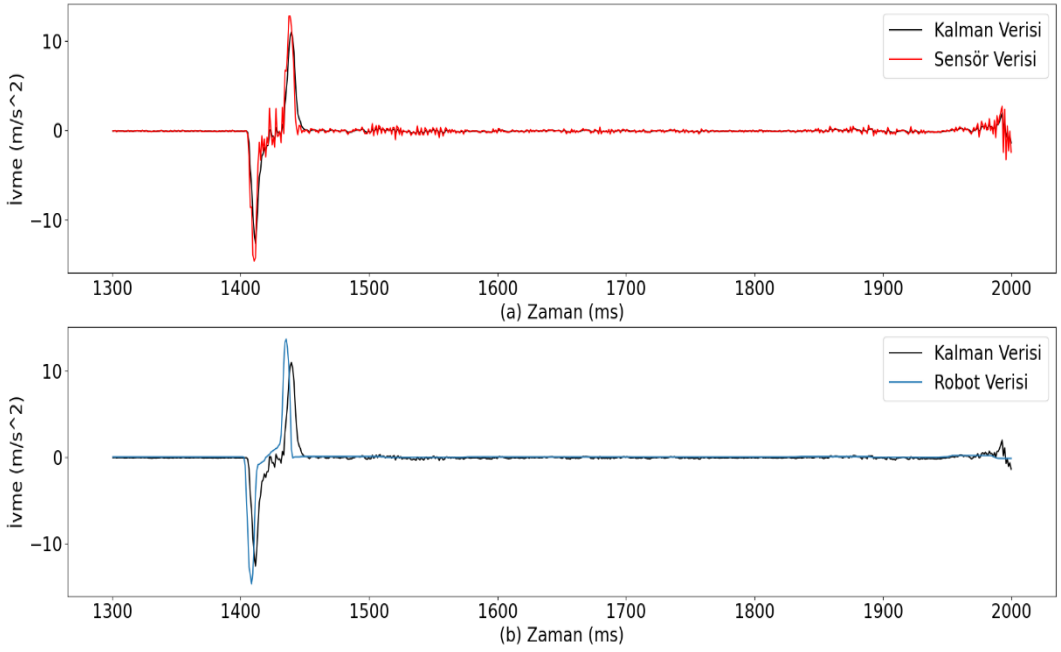
AÖB sensör ivme verilerine kalman filtresi uygulanmış durumlarının ve ham hallerinin kıyaslanmış grafiklerini ise Şekil 4.3 - 4.11 arasında incelemek mümkündür.



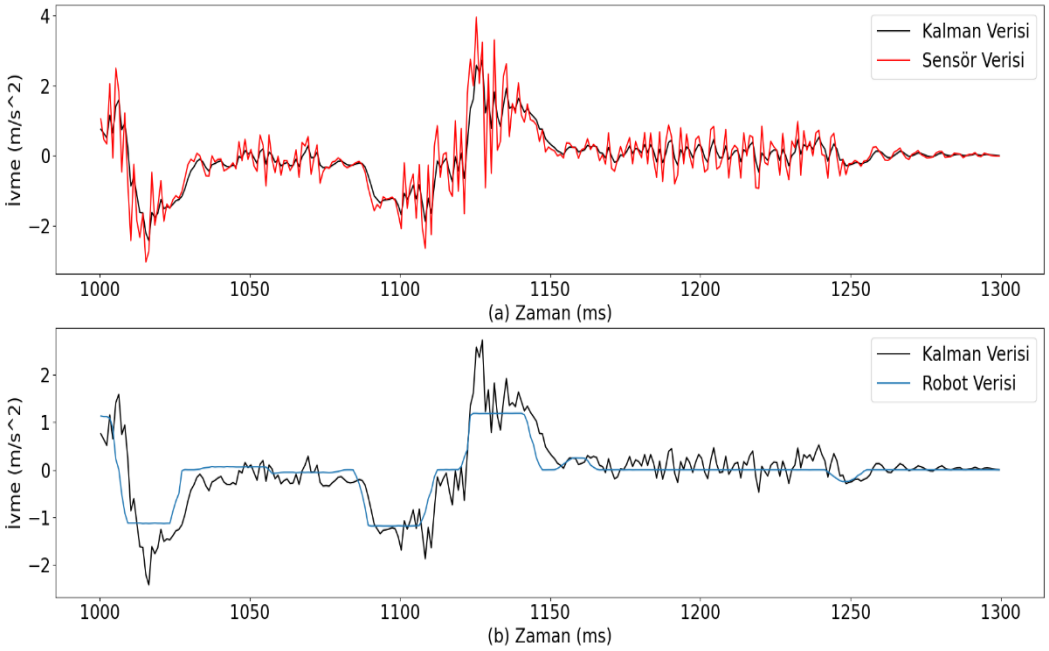
Şekil 4.3: a) 1. AÖB sensörüne ait X yönlü ham ivme verisi ve kalman verisi
b) Robot kolu flanşına ait ivme verisi ve kalman verisi.



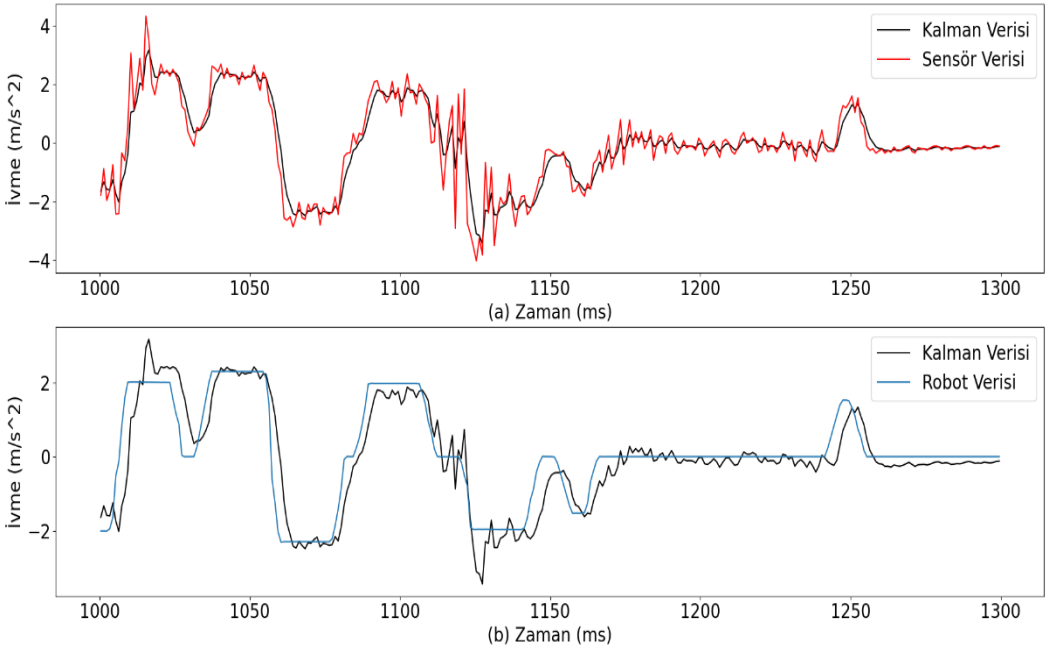
Şekil 4.4: a) 1. AÖB sensörüne ait Y yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi.



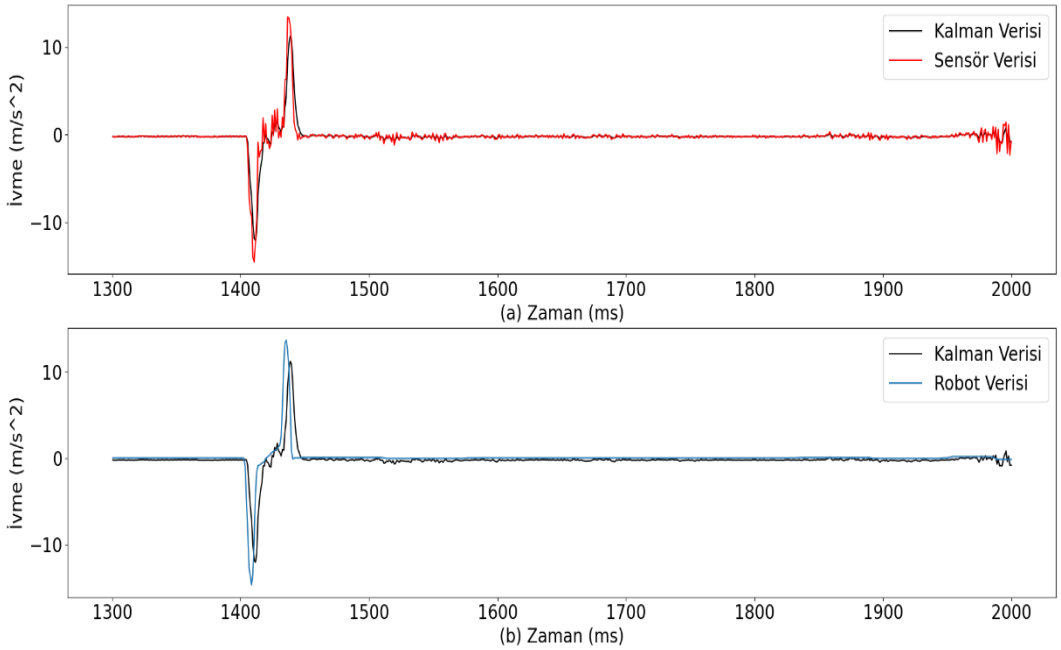
Şekil 4.5: a) 1. AÖB sensörüne ait Z yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi.



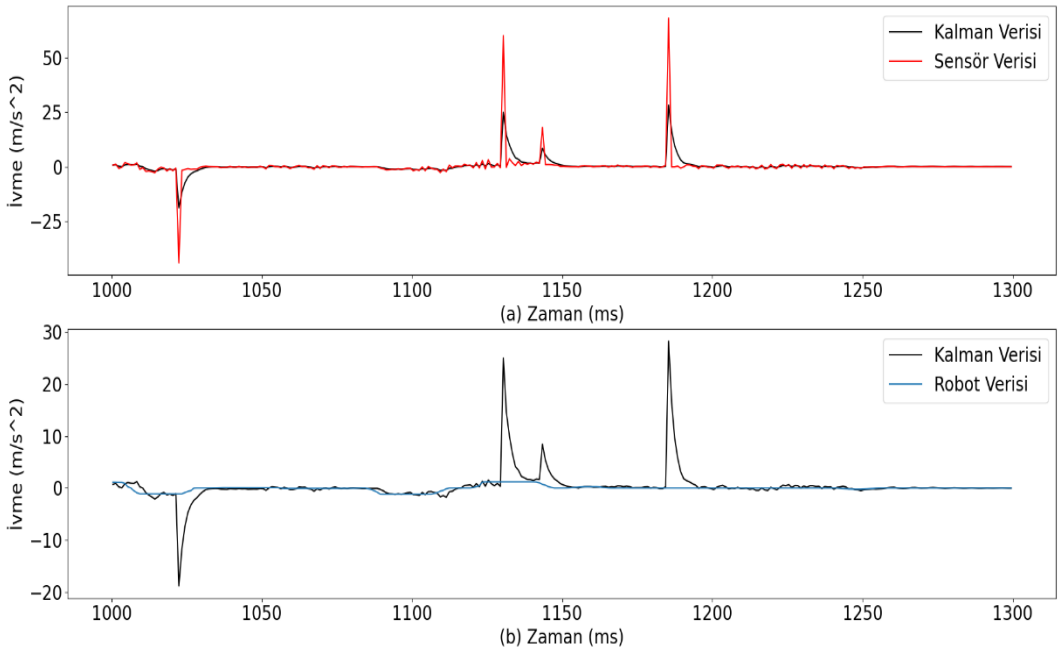
Şekil 4.6: a) 2. AÖB sensörüne ait X yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi.



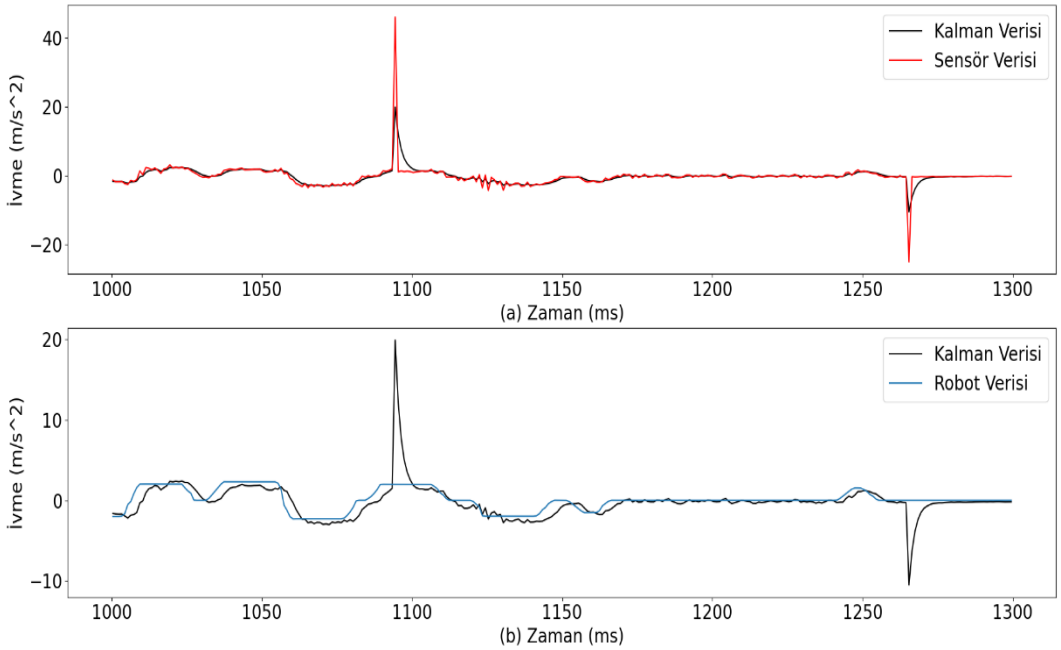
Şekil 4.7: a) 2. AÖB sensörüne ait Y yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi.



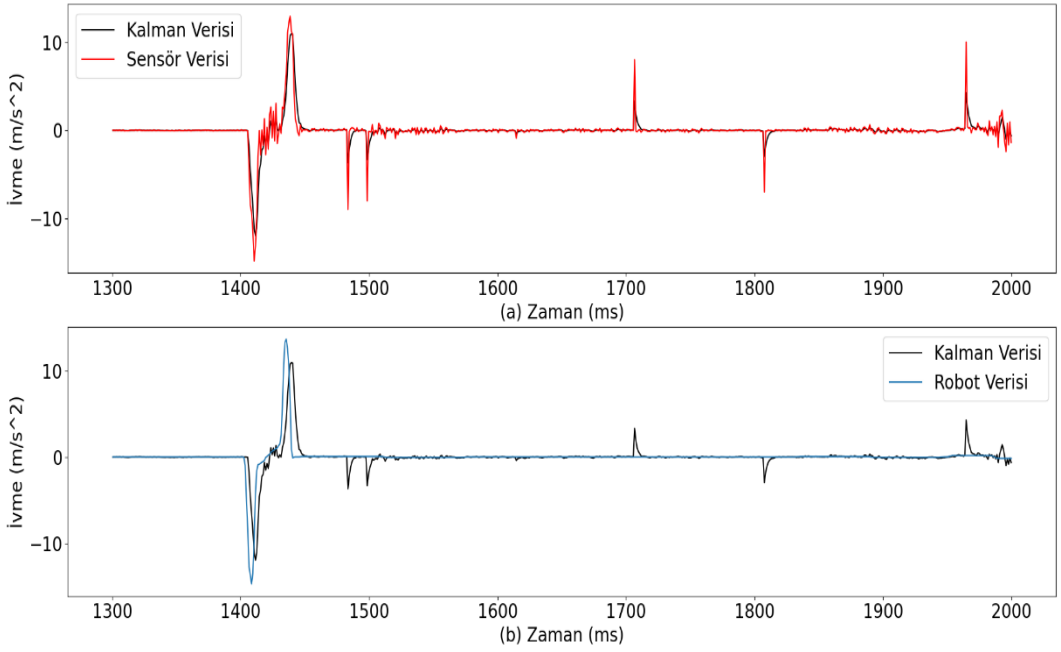
Şekil 4.8: a) 2. AÖB sensörüne ait Z yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi.



Şekil 4.9: a) 3. AÖB sensörüne ait X yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi.

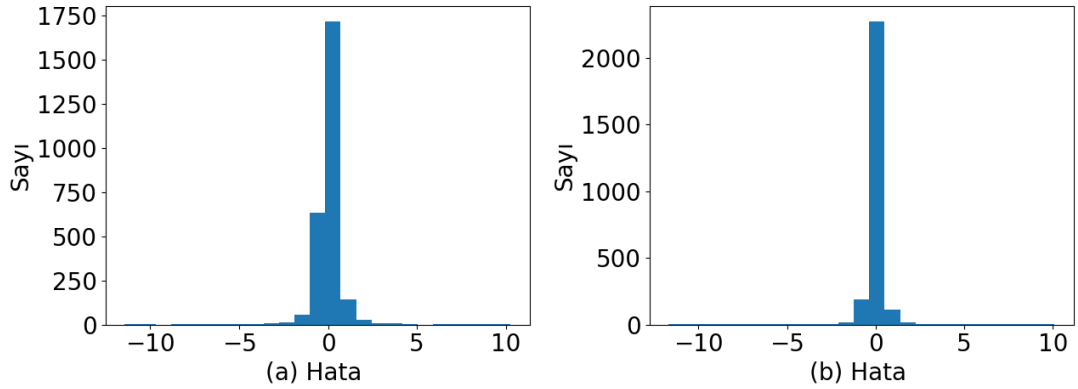


Şekil 4.10: a) 3. AÖB sensörüne ait Y yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi.

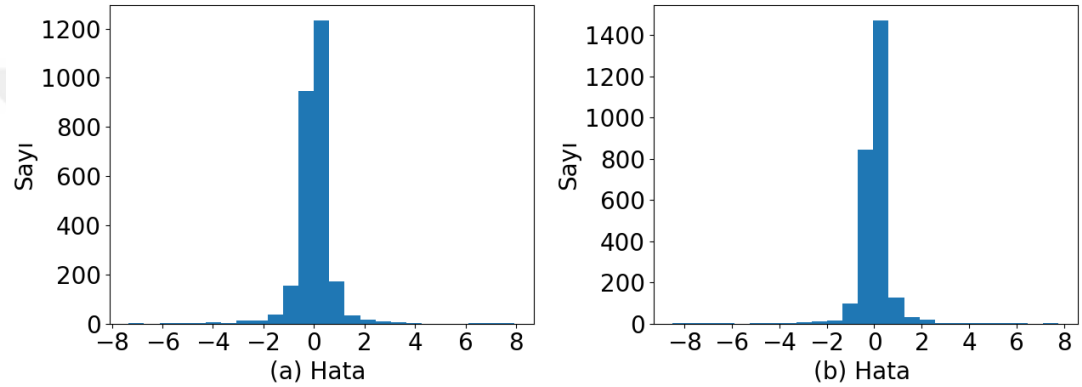


Şekil 4.11: a) 3. AÖB sensörüne ait Z yönlü ham ivme verisi ve kalman verisi b) Robot kolu flanşına ait ivme verisi ve kalman verisi.

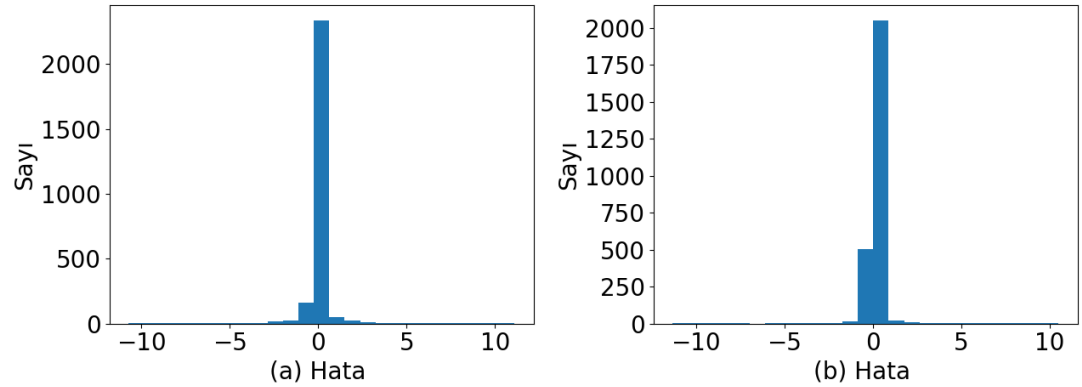
AÖB sensörlerinden gelen ivme verilerinin ve kalman filtresinden geçirilmiş durumlarının hata dağılım grafikleri Şekil 4.12-4.20 arası şekillerde görülebilmektedir. Hata hesabı yapılırken robot kolu flanşının ivme verileri kullanılmıştır.



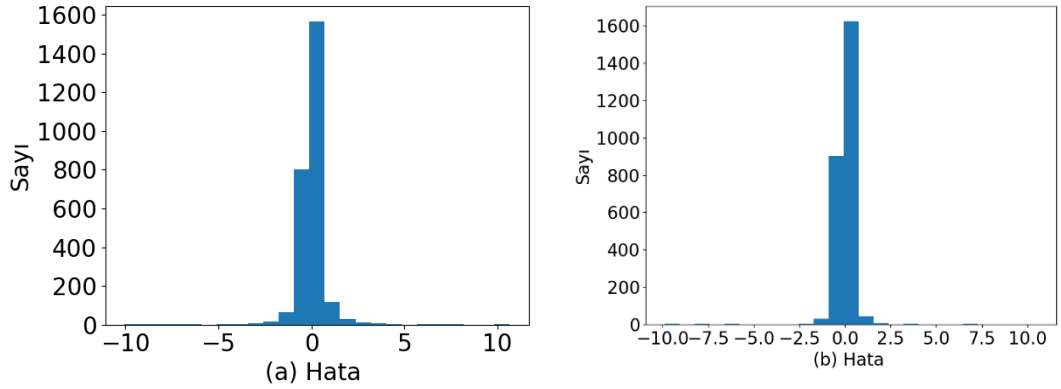
Şekil 4.12: a) 1. AÖB sensörüne ait X yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği.



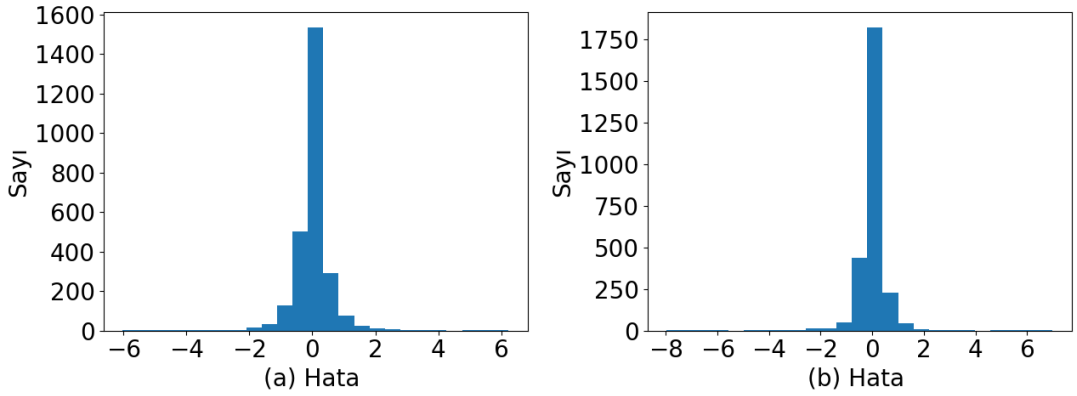
Şekil 4.13: a) 1. AÖB sensörüne ait Y yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği.



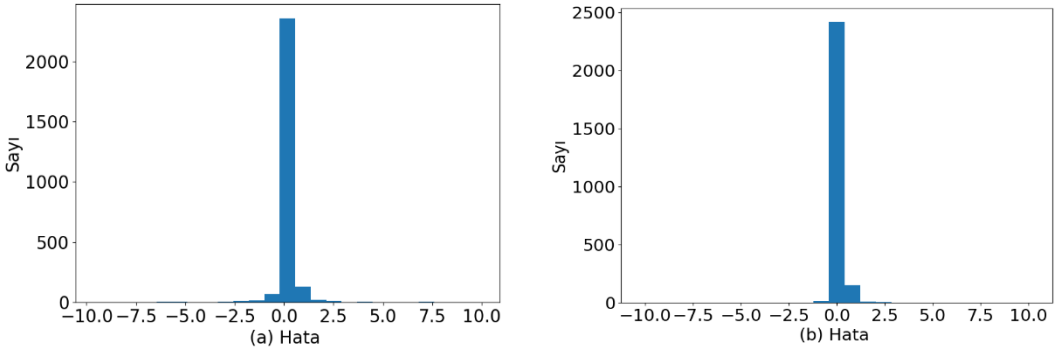
Şekil 4.14: a) 1. AÖB sensörüne ait Z yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği.



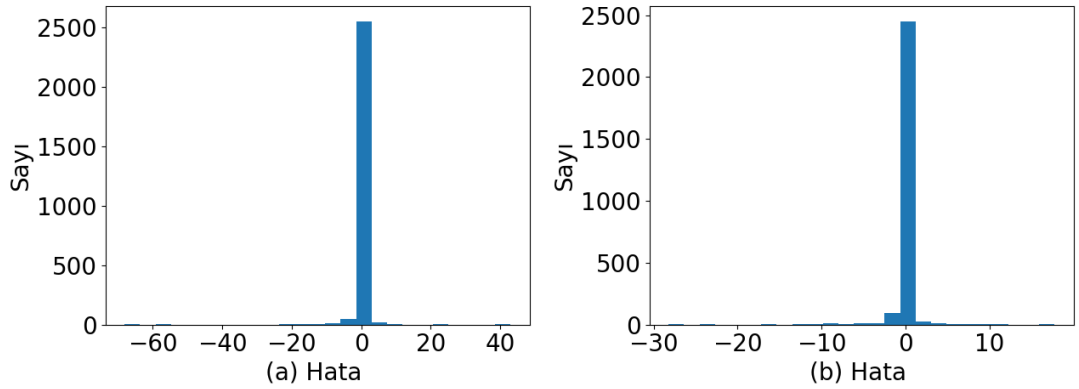
Şekil 4.15: a) 2. AÖB sensörüne ait X yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği.



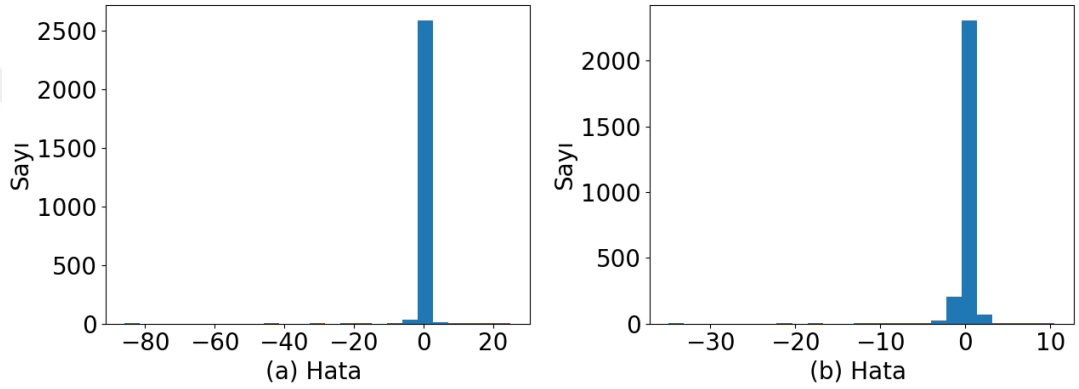
Şekil 4.16: a) 2. AÖB sensörüne ait Y yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği.



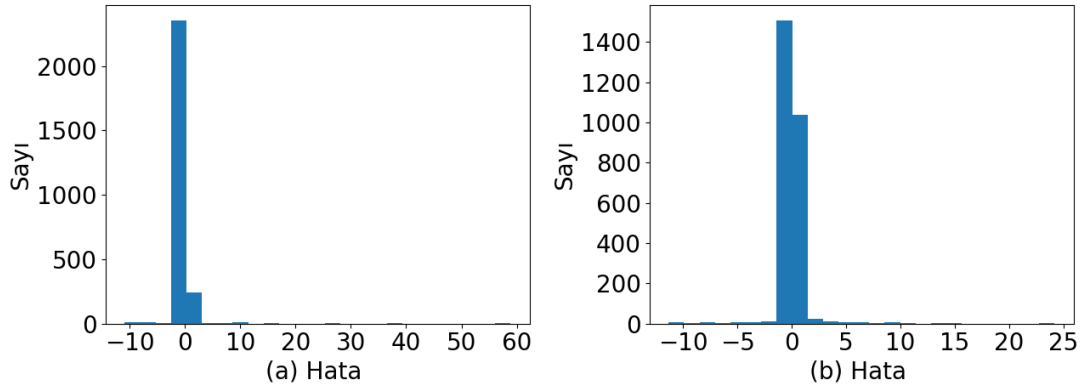
Şekil 4.17: a) 2. AÖB sensörüne ait Z yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği.



Şekil 4.18: a) 3. AÖB sensörüne ait X yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği.



Şekil 4.19: a) 3. AÖB sensörüne ait Y yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği.



Şekil 4.20: a) 3. AÖB sensörüne ait Z yönlü ham ivme verisine ait hata dağılım grafiği. b) Kalman filtresi uygulanmış haline ait hata dağılım grafiği.

Filtresiz ve filtreli hata dağılım grafikleri incelendiğinde sıfıra yakın hata sayılarının arttığı görülmüştür.

AÖB sensör ivme verilerinin kalman filtreli ve filtresiz olarak hesaplanan ortalama mutlak hata değerleri, Çizelge 4.2'de bulunmaktadır. Hata değerleri hesaplanırken robot kolu flanşından ölçülen ivme değerleri referans alınmıştır.

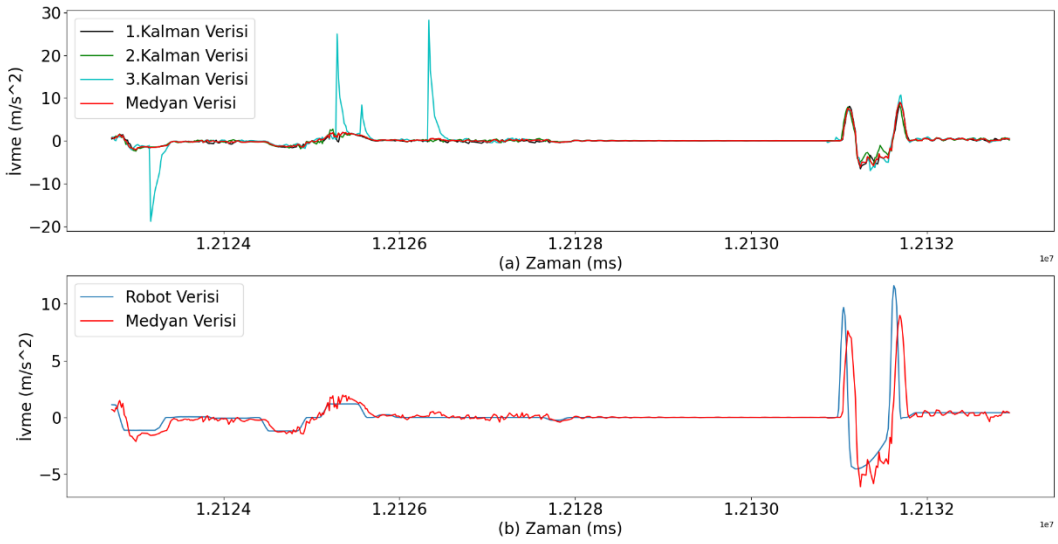
Çizelge 4.2: Hesaplanan Ortalama Mutlak Hata Değerleri.

Sensör	X Kalman Filtresiz	X Kalman Filtreli	Y Kalman Filtresiz	Y Kalman Filtreli	Z Kalman Filtresiz	Z Kalman Filtreli
1.AÖB	0.4285	0.3148	0.3899	0.3437	0.3228	0.2742
2.AÖB	0.3978	0.2885	0.3569	0.3055	0.3807	0.3640
3.AÖB	0.5444	0.4280	0.5260	0.4677	0.3730	0.3105

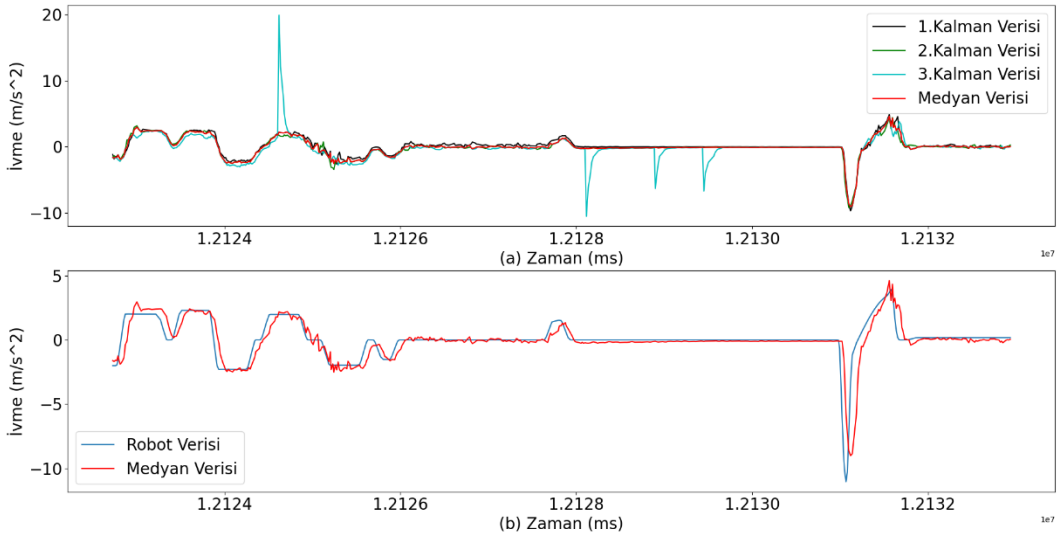
4.1.2 Medyan filtresinin uygulanması

Medyan filtresi, verileri filtre ederken geçmiş veri değerlerini kullanan bir sayısal filtreleme tekniğidir. Filtreleme işlemi sırasında belli sayıda veriyi küçükten, büyüğe sıralayarak ortanca veriyi çıkış olarak vermektedir. Tez kapsamında, medyan filtre geçmiş veriler yerine üç sensörden gelen veriler kullanılarak uygulanmıştır. Bunun sebebi, medyan filtrenin geçmiş değerler ile kullanılması tepki süresinde gecikmeye sebebiyet vermesidir.

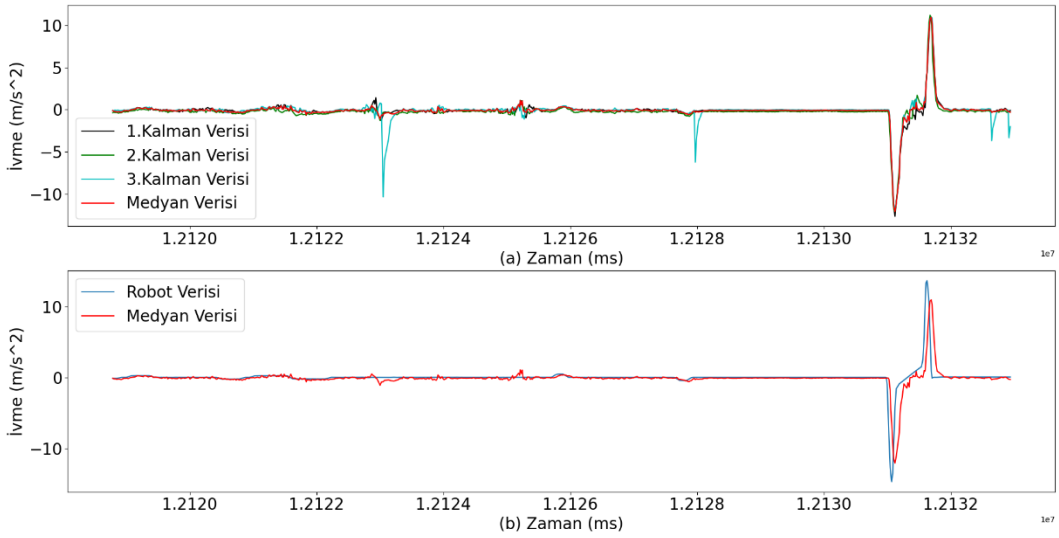
AÖB sensör ivme verilerine medyan filtresi ve kalman filtresi uygulanmış durumlarının kıyaslanmış grafikleri Şekil 4.21, 4.22 ve 4.23 sunulmaktadır.



Şekil 4.21: a) X yönlü ivme verilerine, medyan filtresi uygulanmış veri grafiği, 1. AÖB, 2.AÖB ve 3.AÖB sensörlerine kalman filtresi uygulanmış veri grafiği. b) X yönlü ivme verilerine, medyan filtresi uygulanmış veri grafiği ve robot ivme verisi grafiği.

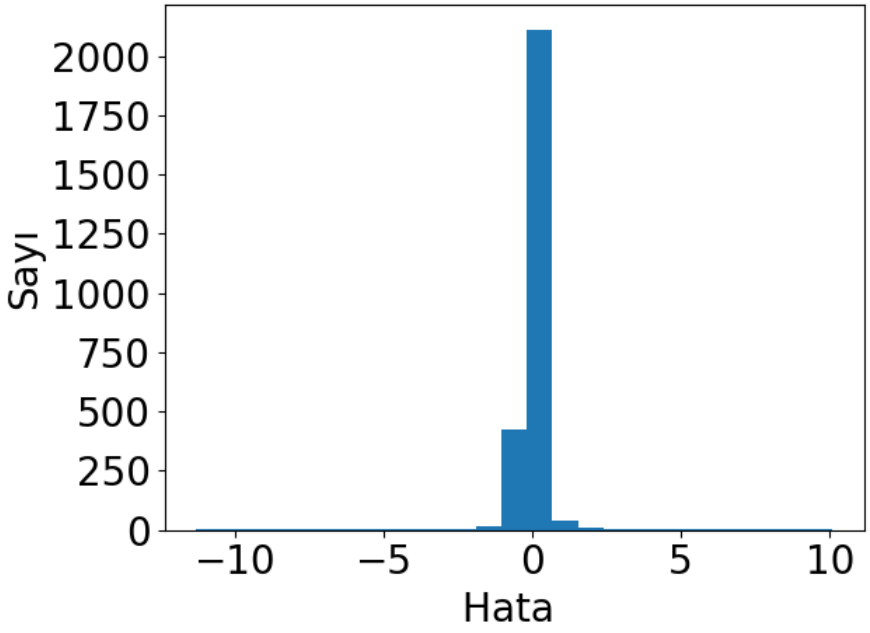


Şekil 4.22: a) Y yönlü ivme verilerine, medyan filtresi uygulanmış veri grafiği, 1. AÖB, 2.AÖB ve 3.AÖB sensörlerine kalman filtresi uygulanmış veri grafiği. b) Y yönlü ivme verilerine, medyan filtresi uygulanmış veri grafiği ve robot ivme verisi grafiği.

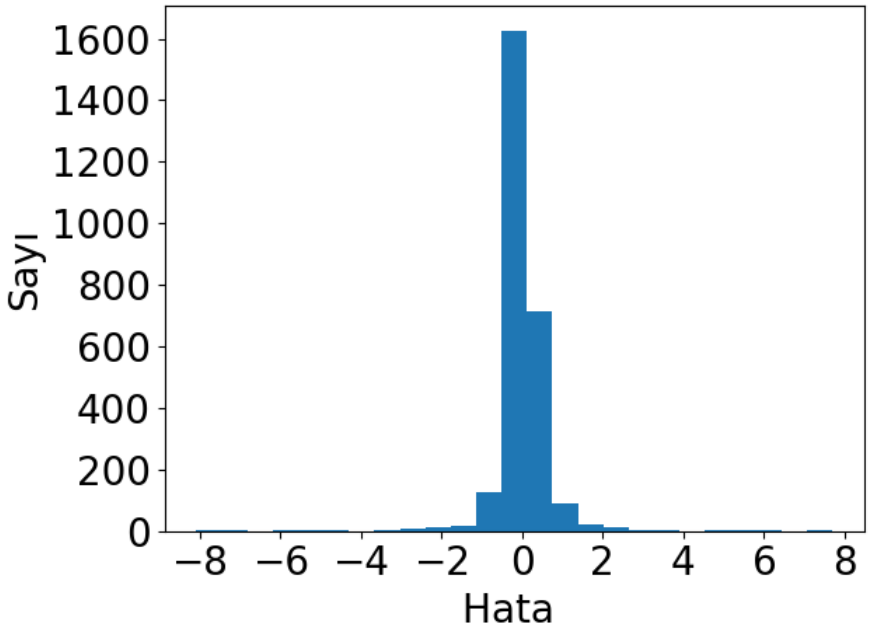


Şekil 4.23: a) Z yönlü ivme verilerine, medyan filtresi uygulanmış veri grafiği, 1. AÖB, 2.AÖB ve 3.AÖB sensörlerine kalman filtresi uygulanmış veri grafiği. b) Z yönlü ivme verilerine, medyan filtresi uygulanmış veri grafiği ve robot ivme verisi grafiği.

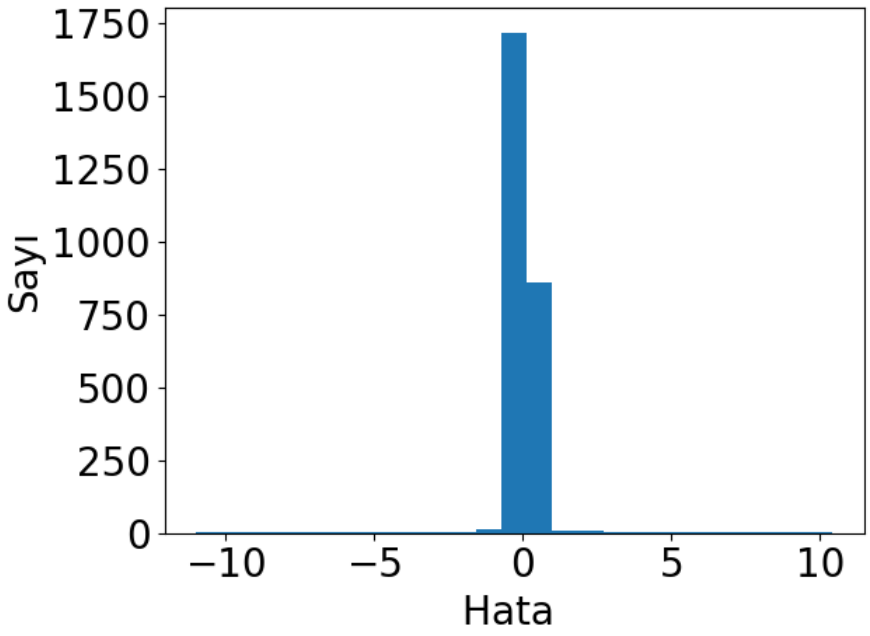
AÖB sensörleri ivme verilerinin medyan filtresinden geçirilmiş durumlarının hata dağılım grafiklerini Şekil 4.24, 4.25 ve 4.26'da görmek mümkündür. Hata hesabı yapılırken robot kolu flanşının ivme verileri kullanılmıştır.



Şekil 4.24: X yönlü ivme verilerine, medyan filtresi uygulanmış sonuçlarının hata dağılım grafiği.



Şekil 4.25: Y yönlü ivme verilerine, medyan filtresi uygulanmış sonuçlarının hata dağılım grafiği.



Şekil 4.26: Z yönlü ivme verilerine, medyan filtresi uygulanmış sonuçlarının hata dağılım grafiği.

Medyan filtreli verilerin hata dağılım grafikleri incelendiğinde sıfıra yakın hata sayılarının kalman filtreli duruma göre artış gösterdiği görülmüştür.

Üç adet AÖB sensör ivme verilerinin tek tek kalman filtreden ve daha sonra üçlü olarak medyan filtreden geçirilmiş hallerine ait ortalama mutlak hata değerleri Çizelge 4.3'te görülebilmektedir. Hata değerleri hesaplanırken robot kolu flanşından ölçülen ivme değerleri referans alınmıştır.

Çizelge 4.3: Medyan filtreli verilere ait ortalama mutlak hata değerleri.

X Yönlü Medyan Filtreli Veri	Y Yönlü Medyan Filtreli Veri	Z Yönlü Medyan Filtreli Veri
0.2827	0.3174	0.2597

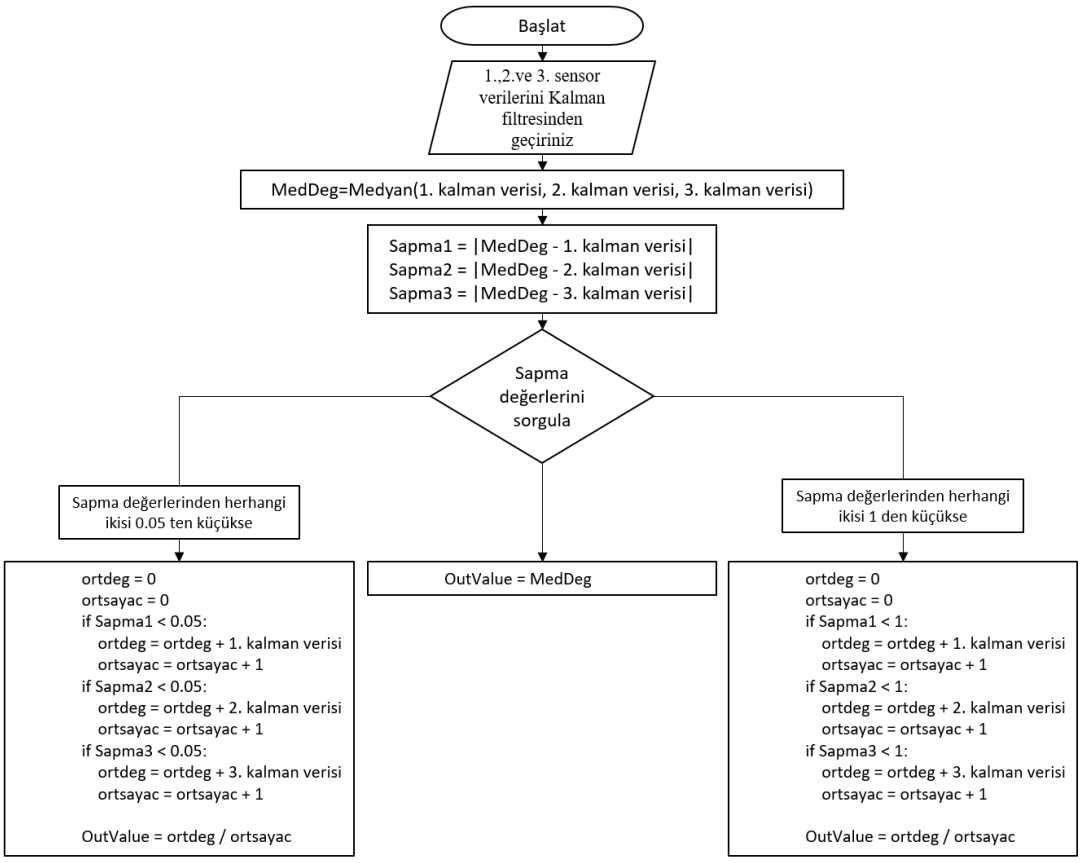
4.1.3 Ortalama filtresinin uygulanması

Ortalama filtresi, verilerin geçmiş değerler ile ortalamasını alan sayısal filtreleme tekniğidir. Bu tez kapsamında, verilerin AÖB sensörlerindeki rastgele gürültülerinin verilerin ortalamaları alınarak nötrlenebileceği düşünülmüştür.

Bazı sensörlerdeki anlık görülebilen çok yüksek gürültülerin, çıkış değerini etkilemesi istenmemektedir. Bu anlık çok yüksek gürültüleri, Şekil 4.9'da görmek mümkündür. Bu sebeple ortalama filtre bir koşula bağlı olarak uygulanmaktadır. Bu koşullar

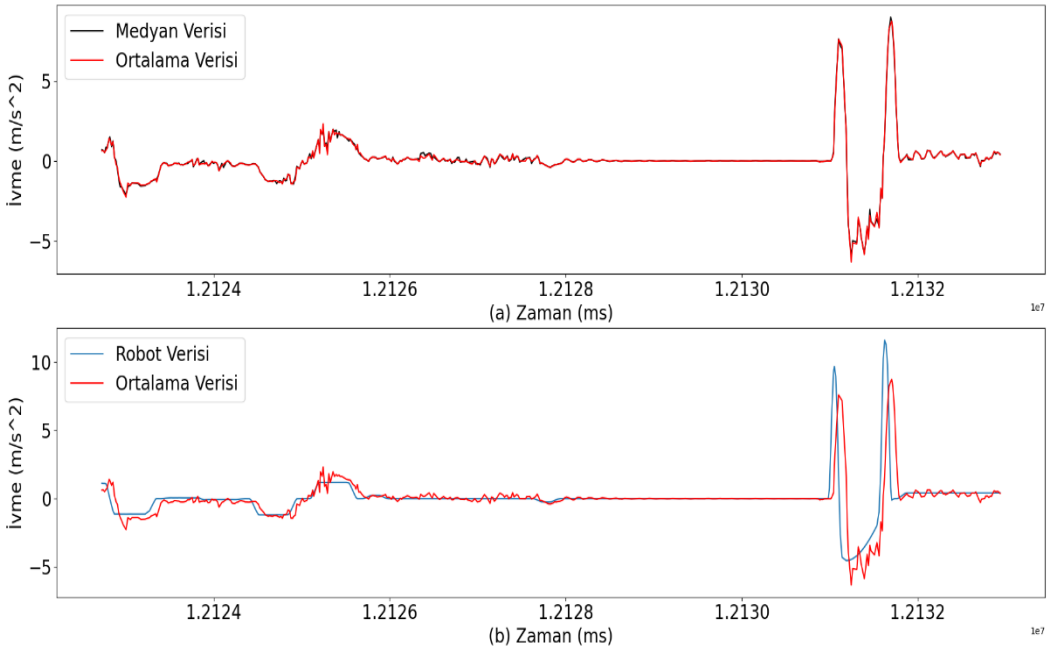
belirlenirken verilerin normal şartlarda birbirlerinden ne kadar saptıkları gözlemlenmiş ve koşullar ona göre belirlenmiştir. Gözlem sonucunda sensörlerin hareketsiz durdukları durumlarda ortalama ±0.05 m/s^2 kadar saptıkları, hareket halinde ise ±1 m/s^2 kadar saptıkları gözlemlenmiştir.

Koşullar belirlendikten sonra, program şöyle yazılmıştır: Kalman filtreden geçirilen sensör verileri, medyan filtreden geçirilmiş ve kalman filtreden geçen verilerin medyan sonucundan çıkan veriden ne kadar saptıkları hesaplanmıştır. En az iki verinin sapma miktarı ±0.05 m/s^2'nin altındaysa, sapma miktarı ±0.05 m/s^2'nin altında olan verilerin ortalaması alınmıştır. Eğer en az iki verinin sapma miktarı ±1 m/s^2'nin altındaysa, sapma miktarı ±1 m/s^2'nin altında olan verilerin ortalaması alınmıştır. Eğer iki koşul da sağlanmıyorsa çıkış değeri olarak medyan filtreden geçen veri kullanılmıştır. Yazılan bu programa ait akış diyagramı Şekil 4.27'de gösterilmiştir.

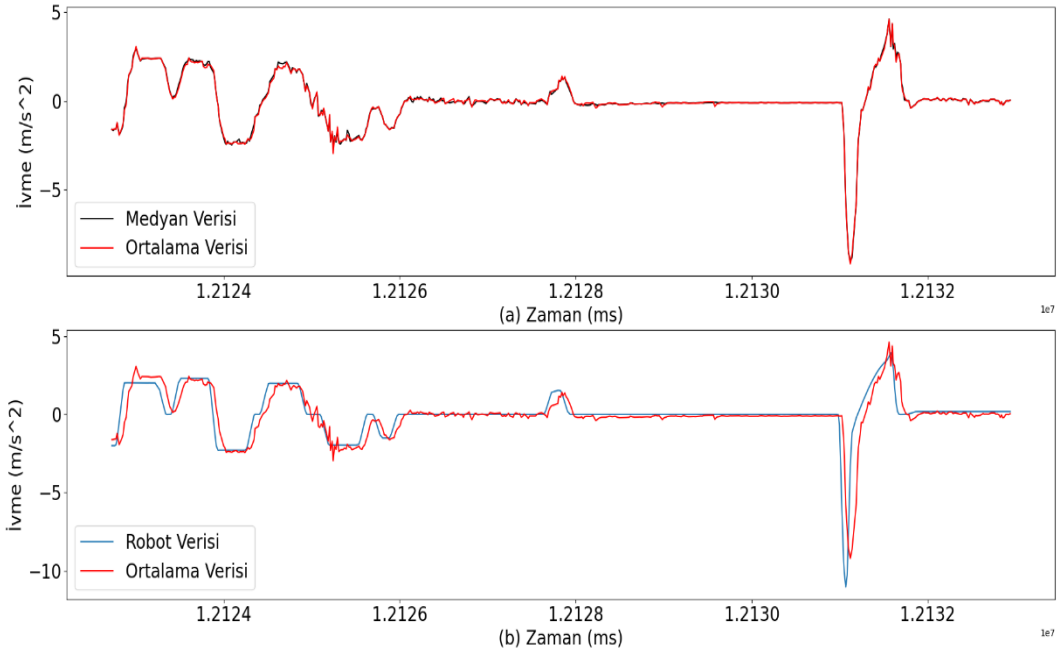


Şekil 4.27: Koşullu ortalama filtresinin akış diyagramı.

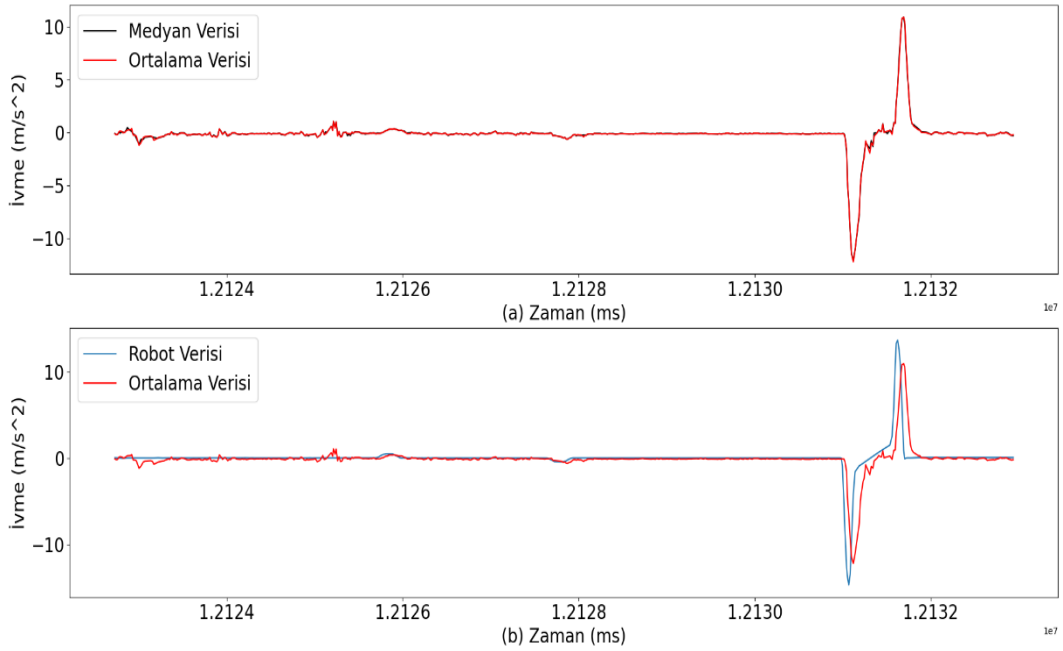
Kalman filtresinden geçirilmiş ivme verilerine koşullu ortalama filtresi ve medyan filtresi uygulanmış durumlarının kıyaslanmış grafikleri Şekil 4.28, 4.29 ve 4.30'de sunulmaktadır.



Şekil 4.28: a) X yönlü koşullu ortalama filtre ivme grafiği ve medyan filtresi ivme grafiği b) X yönlü koşullu ortalama filtre ivme grafiği ve robot ivme grafiği.

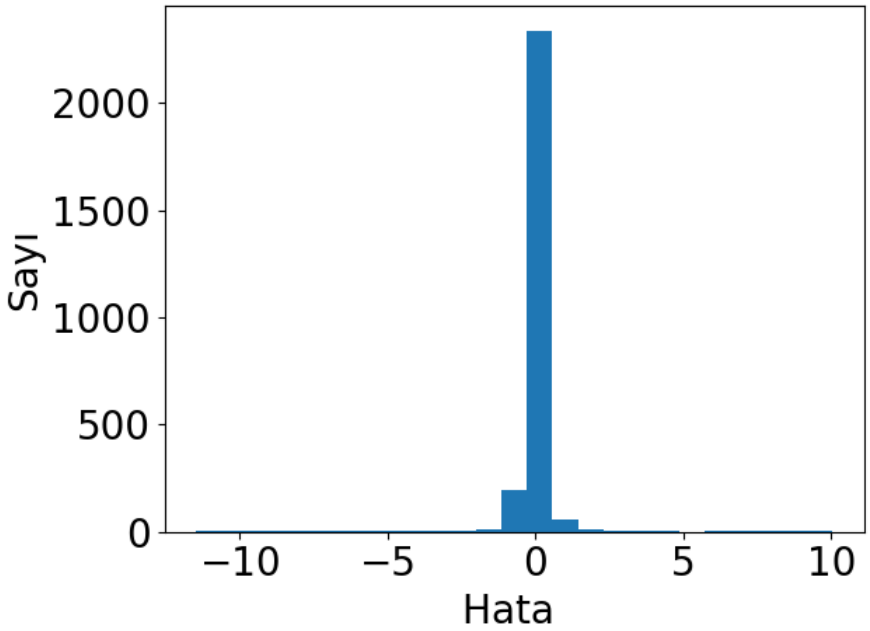


Şekil 4.29: a) Y yönlü koşullu ortalama filtre ivme grafiği ve medyan filtresi ivme grafiği b) Y yönlü koşullu ortalama filtre ivme grafiği ve robot ivme grafiği.

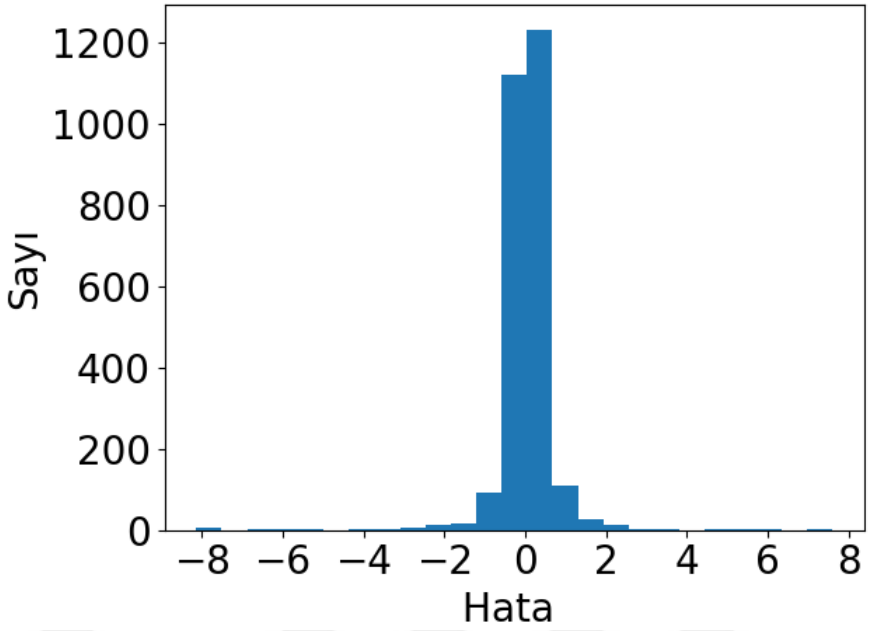


Şekil 4.30: a) Z yönlü koşullu ortalama filtre ivme grafiği ve medyan filtresi ivme grafiği b) Z yönlü koşullu ortalama filtre ivme grafiği ve robot ivme grafiği.

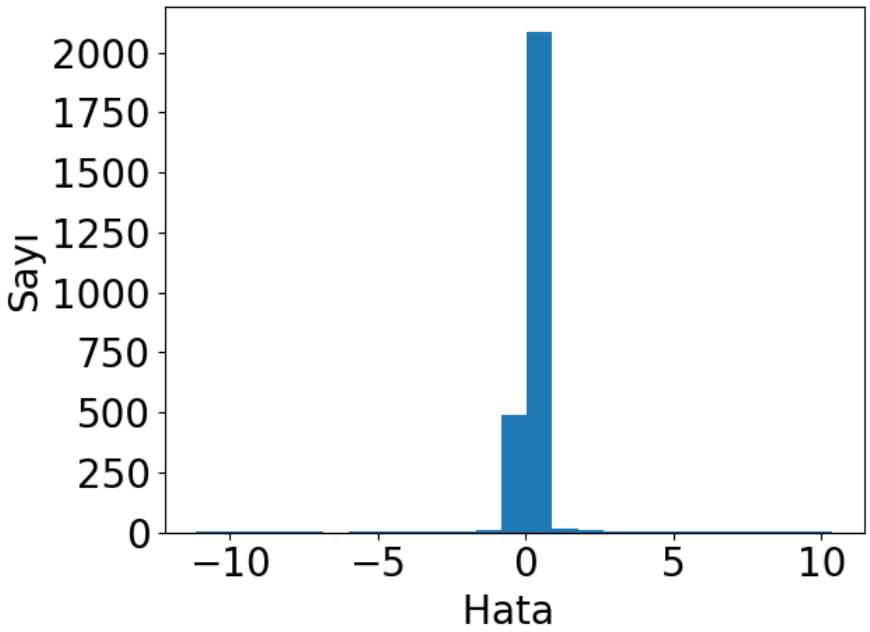
Kalman filtresinden geçirilmiş ivme verilerine koşullu ortalama filtresi uygulanmış durumlarının, hata dağılım grafikleri Şekil 4.31, 4.32 ve 4.33'da görülebilmektedir. Hata hesabı yapılırken robot kolu flanşının ivme verileri kullanılmıştır.



Şekil 4.31: X yönlü ivme verilerine, koşullu ortalama filtresi uygulanmış sonuçlarının hata dağılım grafiği.



Şekil 4.32: Y yönlü ivme verilerine, koşullu ortalama filtresi uygulanmış sonuçlarının hata dağılım grafiği.



Şekil 4.33: Z yönlü ivme verilerine, koşullu ortalama filtresi uygulanmış sonuçlarının hata dağılım grafiği.

Koşullu ortalama filtresi uygulanmış verilerin hata dağılım grafikleri incelendiğinde sıfıra yakın hata sayılarının medya filtreye göre daha fazla olduğu görülmüştür.

İvme verilerinin koşullu ortalama filtreden geçirilmiş hallerine ait ortalama mutlak hata değerlerini Çizelge 4.4'te görmek mümkündür. Hata değerleri hesaplanırken, robot kolu flanşından ölçülen ivme değerleri referans alınmıştır.

Çizelge 4.4: Koşullu ortalama filtreli verilere ait ortalama mutlak hata değerleri.

X Yönlü Ortalama Filtreli Veri	Y Yönlü Ortalama Filtreli Veri	Z Yönlü Ortalama Filtreli Veri
0.2805	0.3165	0.2638

4.2 Filtrelenmiş İvme Verilerinin Makine Öğrenmesi Tekniği İle Model Eğitilmesiyle İyileştirilmesi

Makine öğrenmesi, veri setlerini kullanarak tahmin etme algoritmaları geliştirmeyi amaçlayan bir bilim dalıdır. Bu tez kapsamında filtrelenmiş ve endüstriyel robot kolu ivme verileri ile işaretlenmiş AÖB sensörü ivme verileri kullanılarak, modeller eğitilmiştir. Bu eğitilen modeller ile ivme verilerinin iyileştirilmesi amaçlanmıştır.

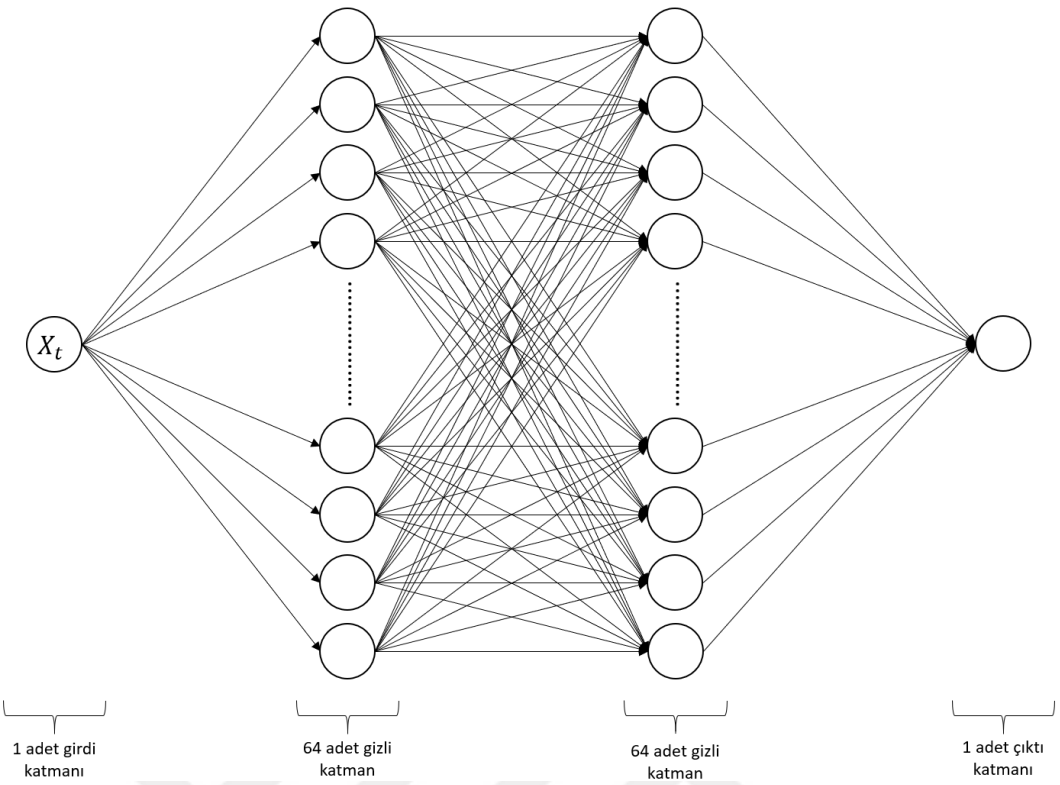
Makine öğretmesinde tahmin etme algoritmaları geliştirmeyi amaçlayan birçok yöntem vardır. Tez kapsamında, derin sinir ağları regresyonu (DNN regression) yöntemi kullanılmıştır. Bunun sebebi, çıkış olarak sayısal bir çıkış almak istememiz ve veri skalasının çok geniş olmasıdır.

Derin sinir ağları regresyonu modelleri eğitmek için, TensorFlow kütüphanesi kullanılmıştır. TensorFlow kütüphanesi, Google Brain ekibinin geliştirdiği ücretsiz açık kaynak bir kütüphanedir.

Tahmin etme modelleri üç farklı şekilde eğitilip aralarındaki farklar incelenmiştir. 1. modelde t anındaki ivme hesabı için t anındaki ölçülen ivme verisi kullanılmıştır. 2. modelde t anındaki ivme hesabı için t anındaki ve ondan önceki dokuz adet ivme verileri kullanılmıştır. 3. modelde t anındaki ivme hesabı için t anındaki, ondan önceki dokuz ve sonraki dokuz ivme verileri kullanılmıştır.

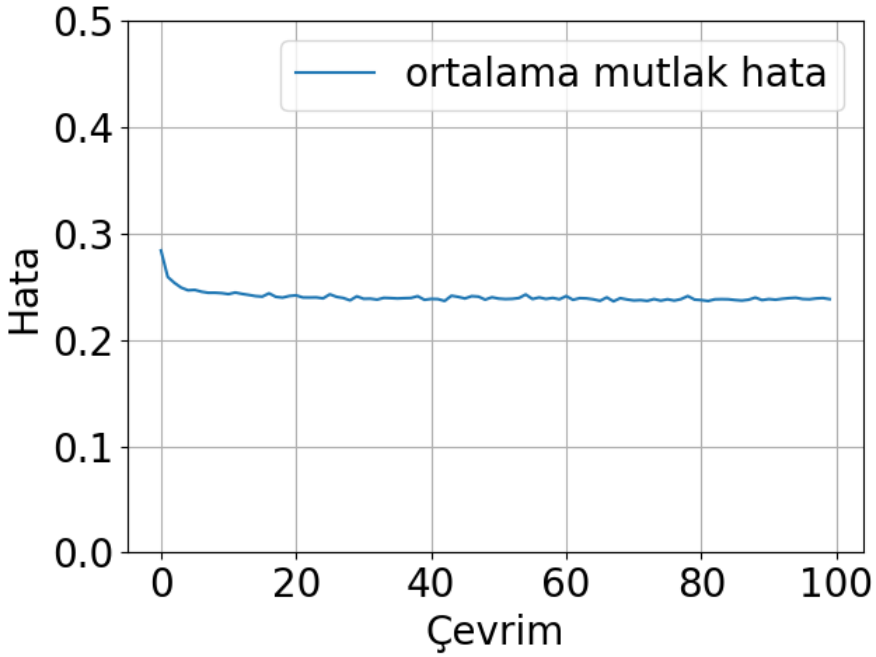
4.2.1 Birinci model: tek girdili DNN regresyon modeli

Tek girdili model eğitilmeden önce, veriler normalize edildi. Bu işlem için tensorflow kütüphanesinin “Normalization” fonksiyonu kullanıldı. Model 1 girdi katmanı, iki adet altmışdört parametreli gizli katman ve bir çıktı katmanı olacak şekilde inşa edildi (Şekil 4.34). Böylelikle toplamda 4356 parametre oluşturuldu, bunların 4353 tanesi eğitilebilir parametredir. Model derlenirken ortalama mutlak hatanın 0.001’e ulaşması hedef gösterildi ve eğitimin 100 çevrim ile yapılması ayarlandı. Tek girdiden t anındaki ivmeye karşılık, t anında sensörden ölçülen değer verilmiştir..



Şekil 4.34: Tek girdili derin sinir ağı modeli.

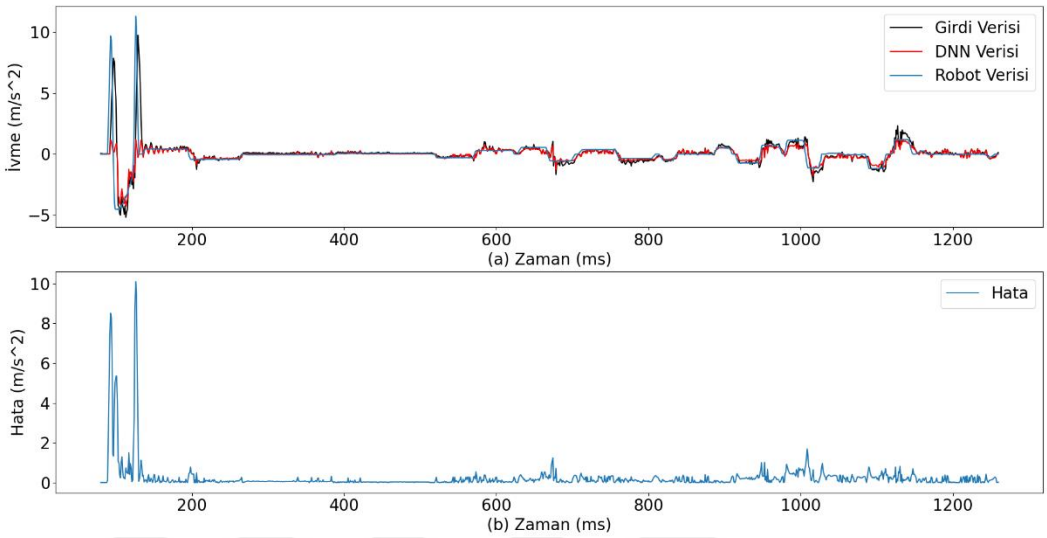
X yönlü ivme tahmini modeli için her eğitim çevriminden sonra ulaşılan ortalama mutlak hata değerlerini Şekil 4.35'deki grafikten görmek mümkündür.



Şekil 4.35: X ivmesi modeli için çevrim – ortalama mutlak hata grafiği.

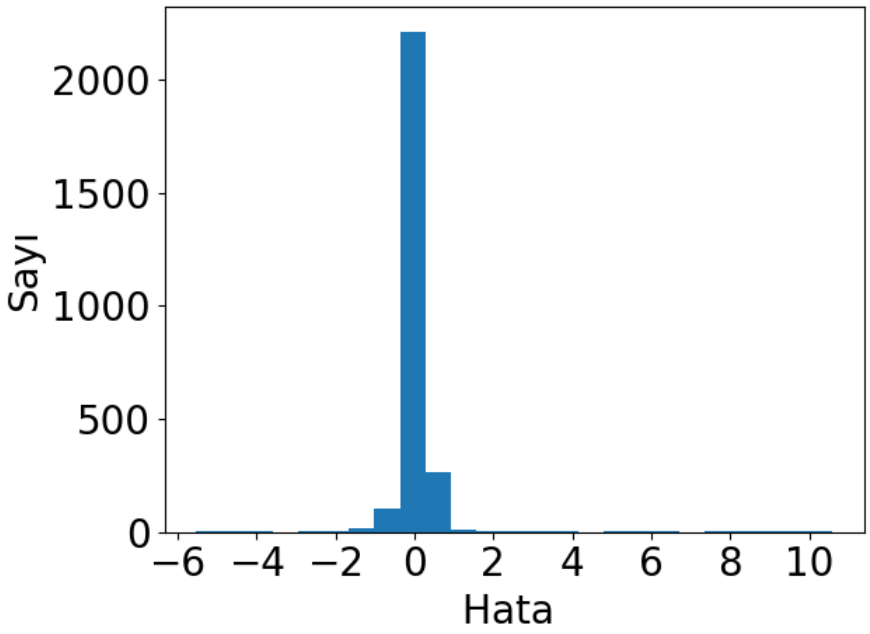
Bu model eğitiminde, mutlak hata değeri 0.2288'e düşürülmüştür. Modelden hesaplanan değerlerin karşılaştırmalı grafiği Şekil 4.36'da görülebilmektedir. Şekil

4.36.a'da X yönlü ivme için eğitilen modelden hesaplanan ivme grafiği, filtrelenmiş ve modelde girdi olarak kullanılan ivme grafiği ve robot kolu flanşından ölçülen ivme grafiği görülmektedir. Şekil 4.36.b'de modelden hesaplanan ivme ile robottan ölçülen ivme arasındaki hata miktarını gösteren grafik görülmektedir.



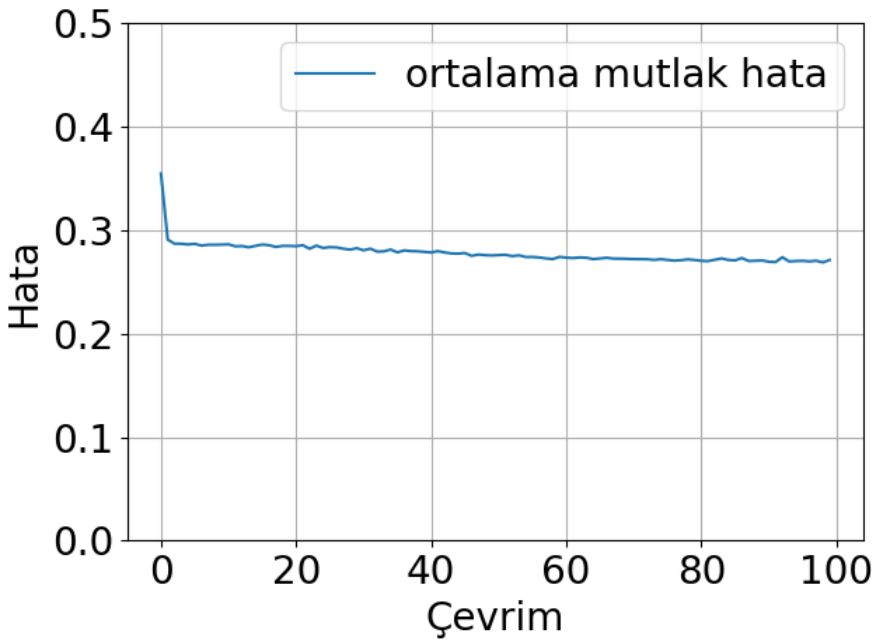
Şekil 4.36: a) X yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği.

X yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği Şekil 4.37'de incelenebilmektedir. Hata hesabı yapılırken robot kolu flanşının ivme verileri kullanılmıştır.



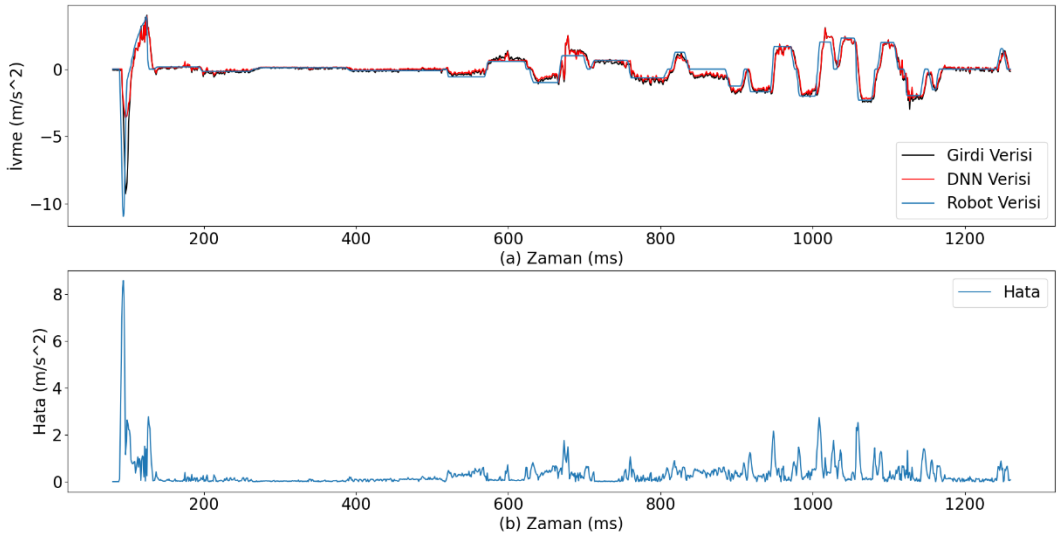
Şekil 4.37: X yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği.

Y yönlü ivme tahmini modeli için her eğitim çevriminden sonra ulaşılan ortalama mutlak hata değerlerini Şekil 4.38'deki grafikten görmek mümkündür.



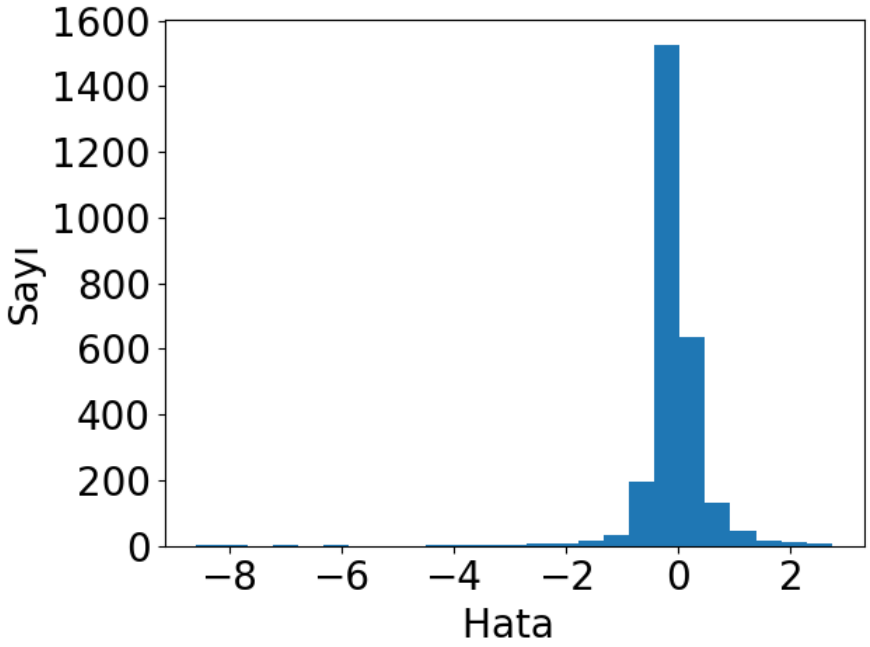
Şekil 4.38: Y ivmesi modeli için çevrim – ortalama mutlak hata grafiği.

Bu model eğitiminde mutlak hata değeri 0.2915'e düşürülmüştür. Modelden hesaplanan değerlerin karşılaştırmalı grafiği Şekil 4.39'de mevcuttur. Şekil 4.39.a'da Y yönlü ivme için eğitilen modelden hesaplanan ivme grafiği, filtrelenmiş ve modelde girdi olarak kullanılan ivme grafiği ve robot kolu flanşından ölçülen ivme grafiği görülmektedir. Şekil 4.39.b'de modelden hesaplanan ivme ile robottan ölçülen ivme arasındaki hata miktarını gösteren grafik görülmektedir.



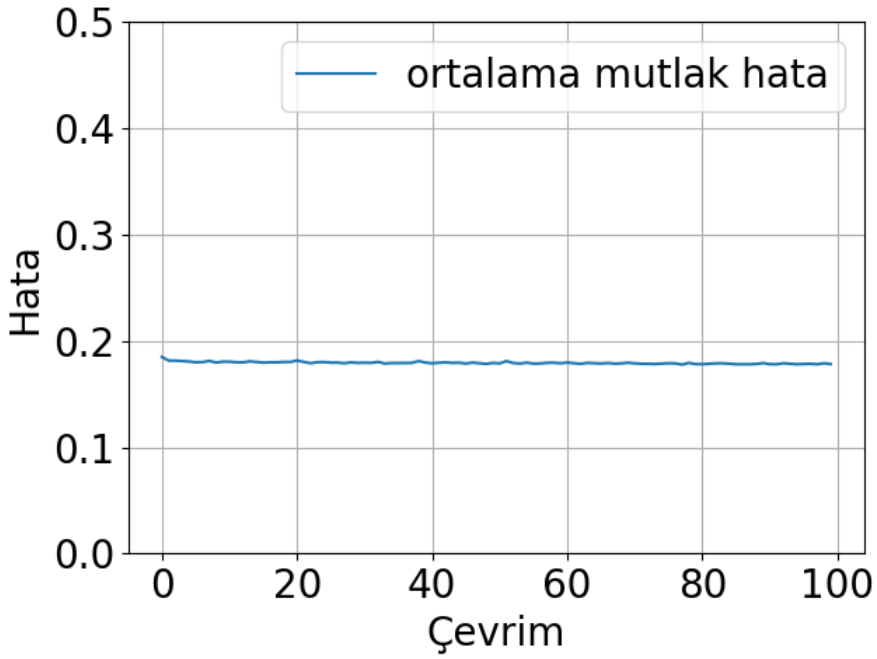
Şekil 4.39: a) Y yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği.

Y yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği Şekil 4.40'ta görülebilmektedir. Hata hesabı yapılırken, robot kolu flanşının ivme verileri kullanılmıştır.



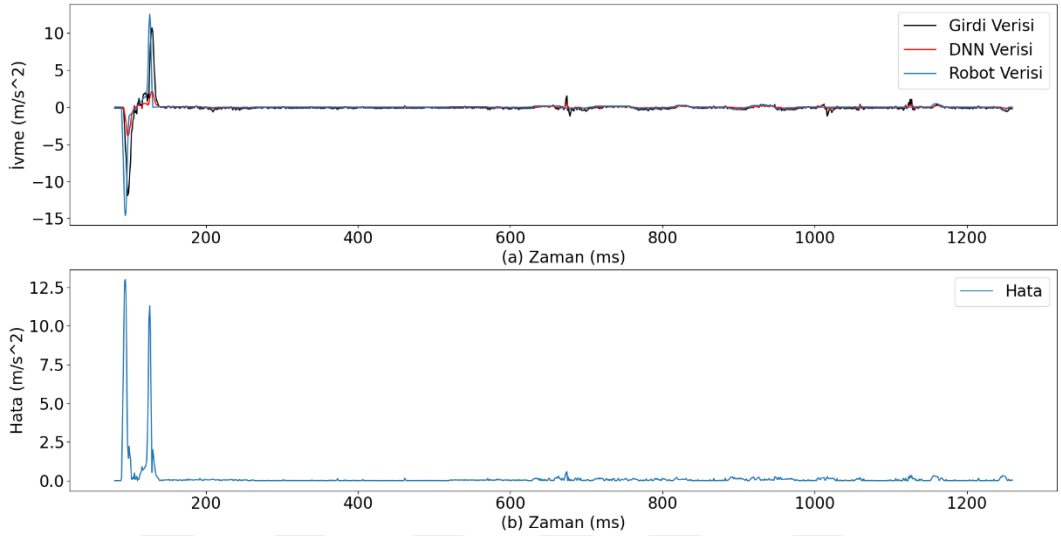
Şekil 4.40: Y yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği.

Z yönlü ivme tahmini modeli için her eğitim çevriminden sonra ulaşılan ortalama mutlak hata değerleri Şekil 4.41'deki grafikte mevcuttur.



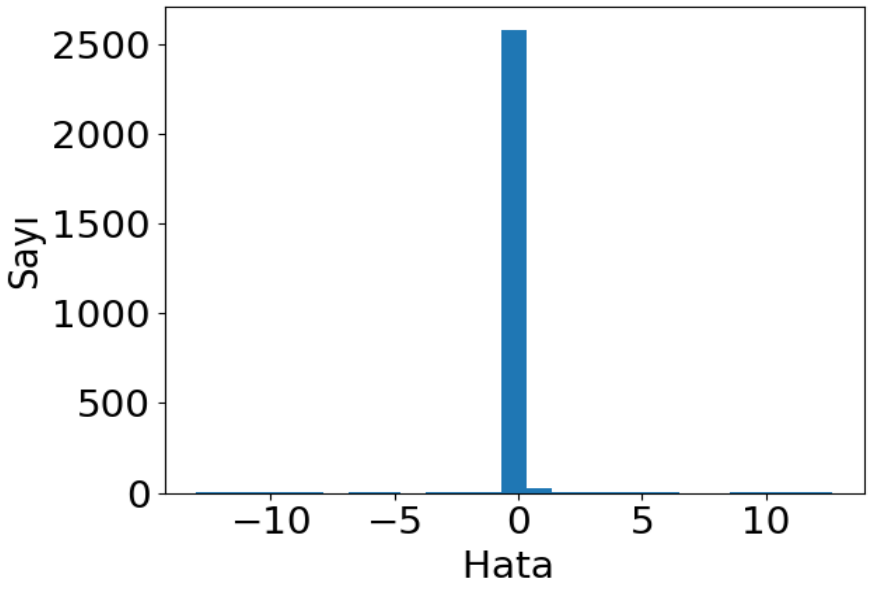
Şekil 4.41: Z ivmesi modeli için çevrim – ortalama mutlak hata grafiği.

Bu model eğitiminde mutlak hata değeri 0.1527'e düşürülmüştür. Modelden hesaplanan değerlerin karşılaştırmalı grafiği Şekil 4.42'de görülebilmektedir. Şekil 4.42.a'da Z yönlü ivme için eğitilen modelden hesaplanan ivme grafiği, filtrelenmiş ve modelde girdi olarak kullanılan ivme grafiği ve robot kolu flanşından ölçülen ivme grafiği görülmektedir. Şekil 4.42.b'de modelden hesaplanan ivme ile robottan ölçülen ivme arasındaki hata miktarını gösteren grafik görülmektedir.



Şekil 4.42: a) Z yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği.

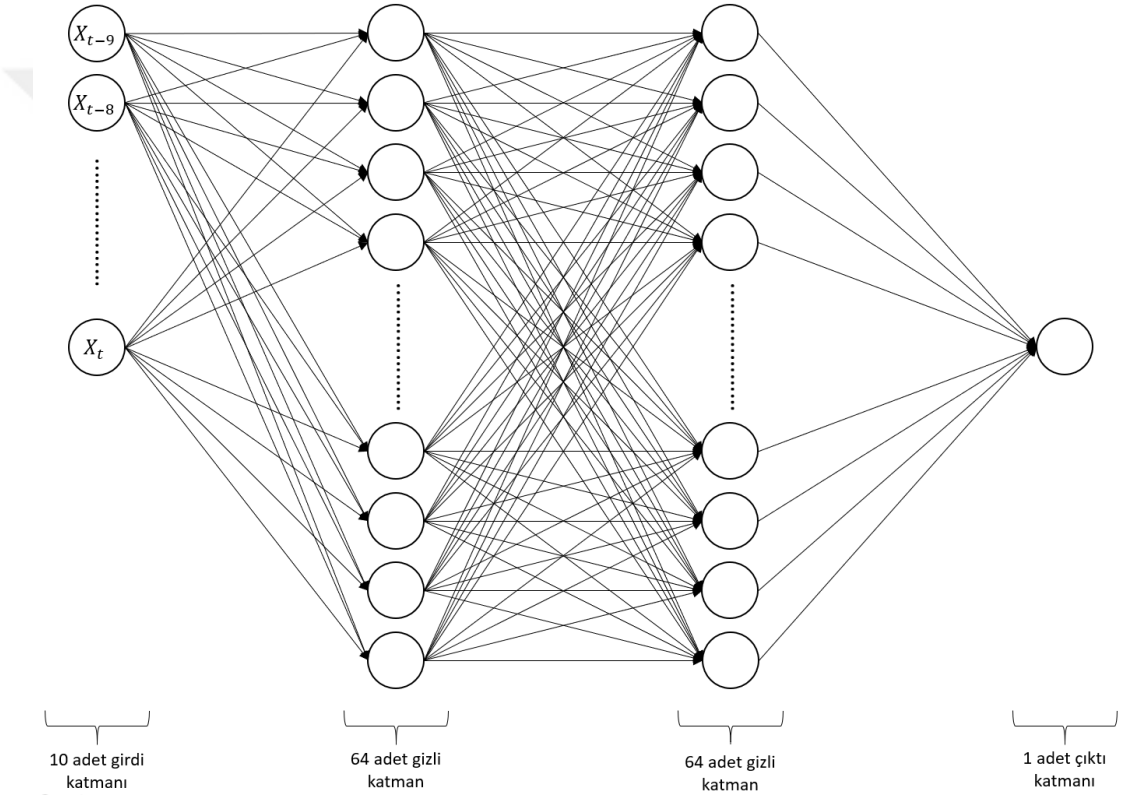
Z yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği Şekil 4.43'de incelenebilmektedir. Hata hesabı yapılırken robot kolu flanşının ivme verileri kullanılmıştır.



Şekil 4.43: Z yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği.

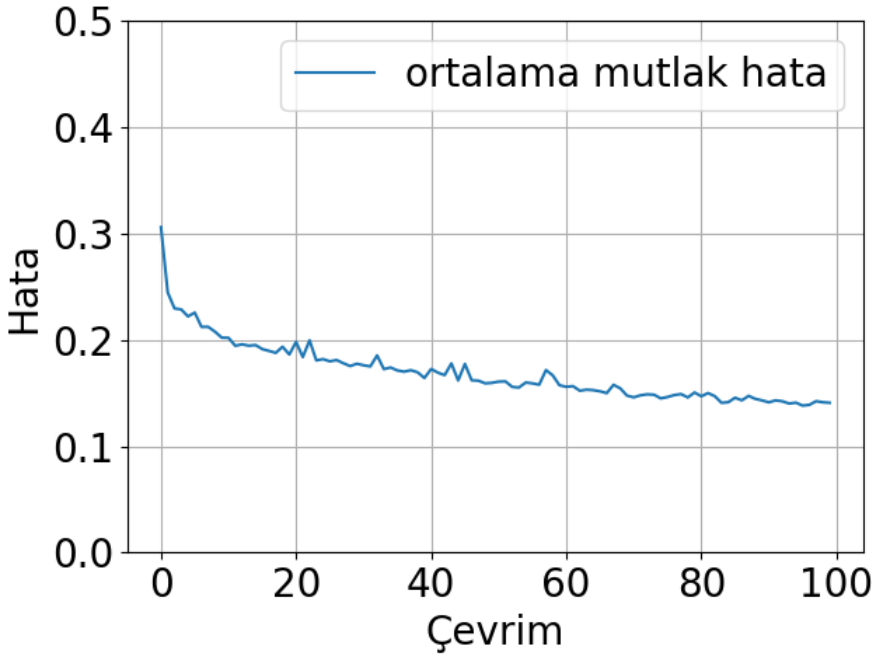
4.2.2 İkinci model: geçmiş veriler ile birlikte 10 girdili DNN regresyon modeli

On girdili model eğitilmeden önce veriler normalize edildi. Bu işlem için tensorflow kütüphanesinin "Normalization" fonksiyonu kullanıldı. Model on girdi katmanı, iki adet altmışdört parametreli gizli katman ve bir çıktı katmanı olacak şekilde inşa edildi (Şekil 4.44). Böylelikle toplamda 4950 parametre oluşturuldu, bunların 4929 tanesi eğitilebilir parametredir. Model derlenirken ortalama mutlak hatanın 0.001'e ulaşması hedef gösterildi ve eğitimin 100 çevrim ile yapılması ayarlanmıştır. On girdiden t anındaki ivmeye karşılık, t anında sensörden ölçülen değer ve t anından önce ölçülen dokuz değer verilmiştir.



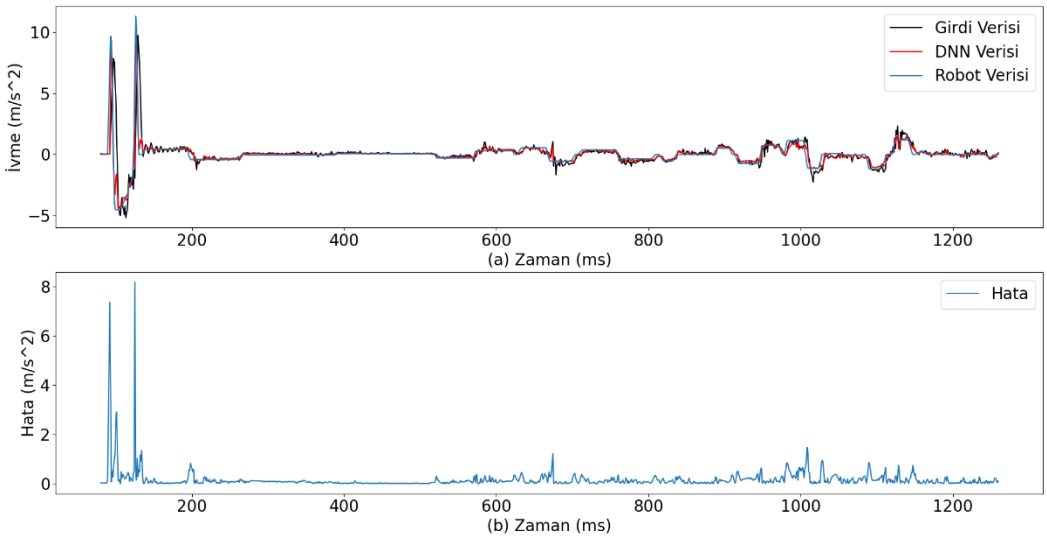
Şekil 4.44: On girdili derin sinir ağı modeli.

X yönlü ivme tahmini modeli için her eğitim çevriminden sonra ulaşılan ortalama mutlak hata değerleri Şekil 4.45'deki grafikten görülebilmektedir.



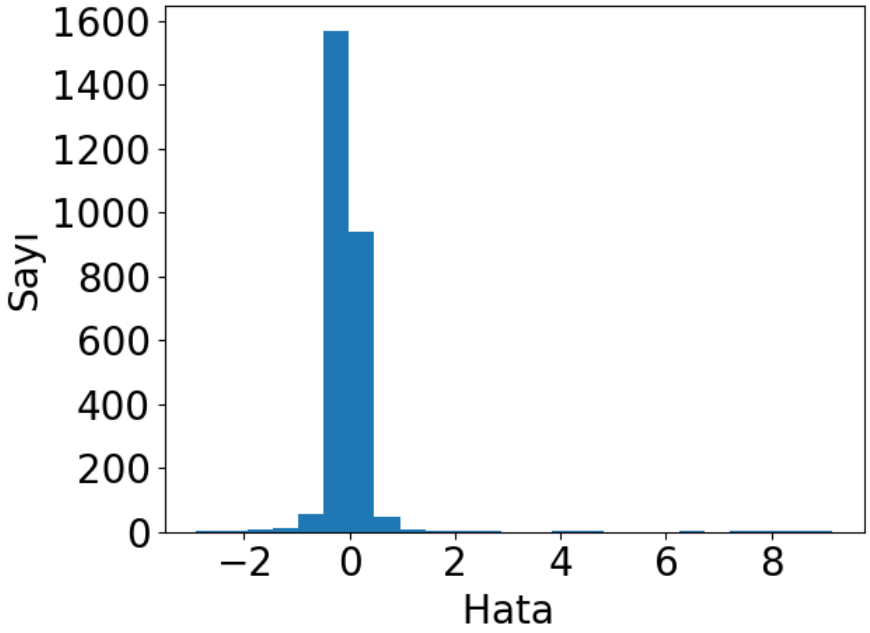
Şekil 4.45: X ivmesi modeli için çevrim – ortalama mutlak hata grafiği.

Bu model eğitiminde ortalama mutlak hata değeri 0.1295'e düşürülmüştür. Modelden hesaplanan değerlerin karşılaştırmalı grafiğini Şekil 4.46'da görmek mümkündür. Şekil 4.46.a'da X yönlü ivme için eğitilen modelden hesaplanan ivme grafiği, filtrelenmiş ve modelde girdi olarak kullanılan ivme grafiği ve robot kolu flanşından ölçülen ivme grafiği görülmektedir. Şekil 4.46.b'de modelden hesaplanan ivme ile robottan ölçülen ivme arasındaki hata miktarını gösteren grafik görülmektedir.



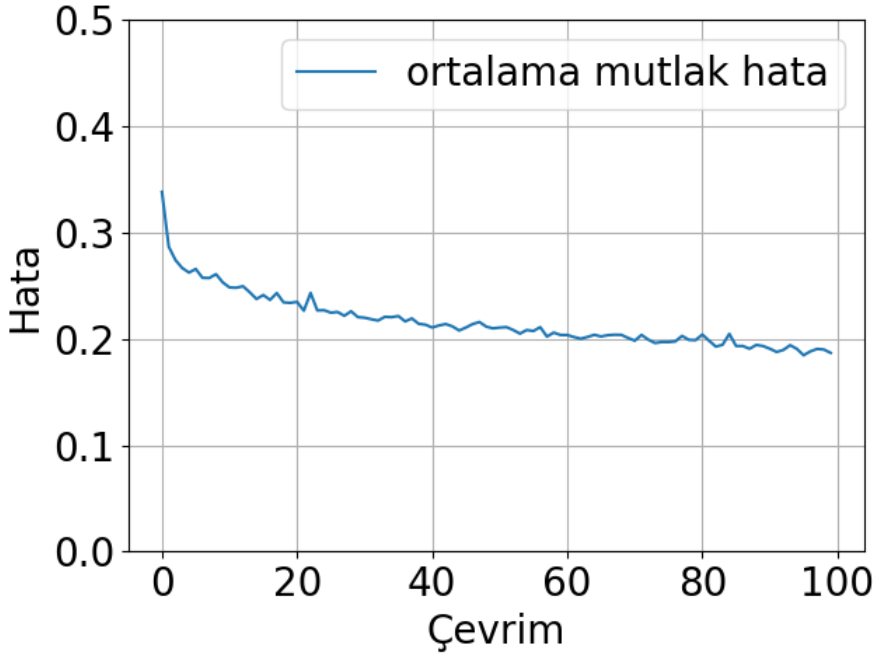
Şekil 4.46: a) X yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği.

X yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği Şekil 4.47'te görmek mümkündür. Hata hesabı yapılırken robot kolu flanşının ivme verileri kullanılmıştır.



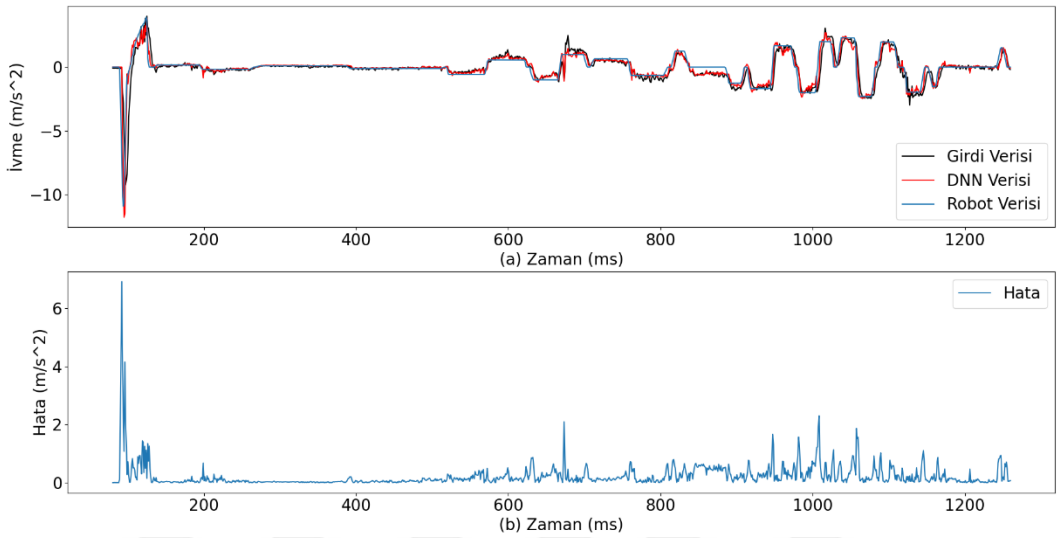
Şekil 4.47: X yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği.

Y yönlü ivme tahmini modeli için her eğitim çevriminden sonra ulaşılan ortalama mutlak hata değerleri Şekil 4.48'deki grafikten incelenebilmektedir.



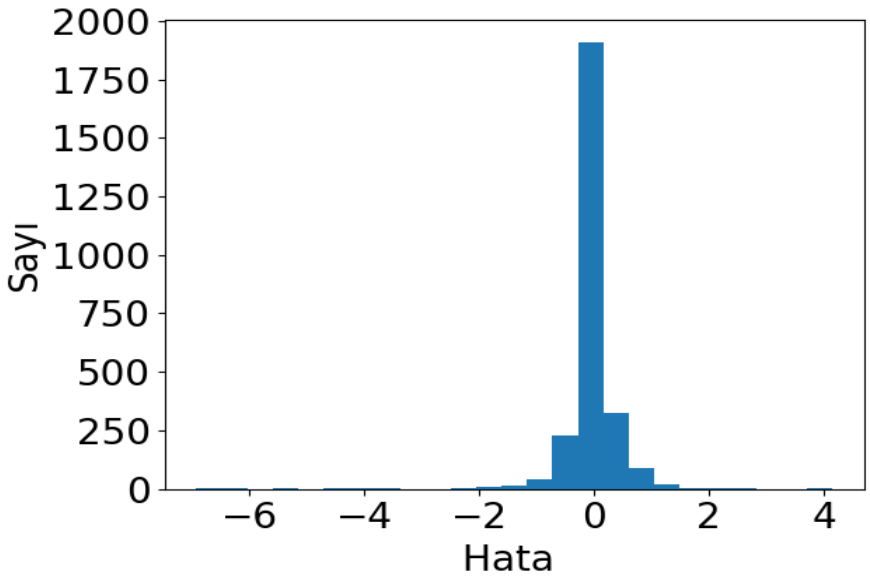
Şekil 4.48: Y ivmesi modeli için çevrim – ortalama mutlak hata grafiği.

Bu model eğitiminde ortalama mutlak hata değeri 0.2086 e düşürülmüştür. Modelden hesaplanan değerlerin karşılaştırmalı grafiği Şekil 4.49'de görülebilmektedir. Şekil 4.49.a'da Y yönlü ivme için eğitilen modelden hesaplanan ivme grafiği, filtrelenmiş ve modelde girdi olarak kullanılan ivme grafiği ve robot kolu flanşından ölçülen ivme grafiği görülmektedir. Şekil 4.49.b'de modelden hesaplanan ivme ile robottan ölçülen ivme arasındaki hata miktarını gösteren grafik görülmektedir.



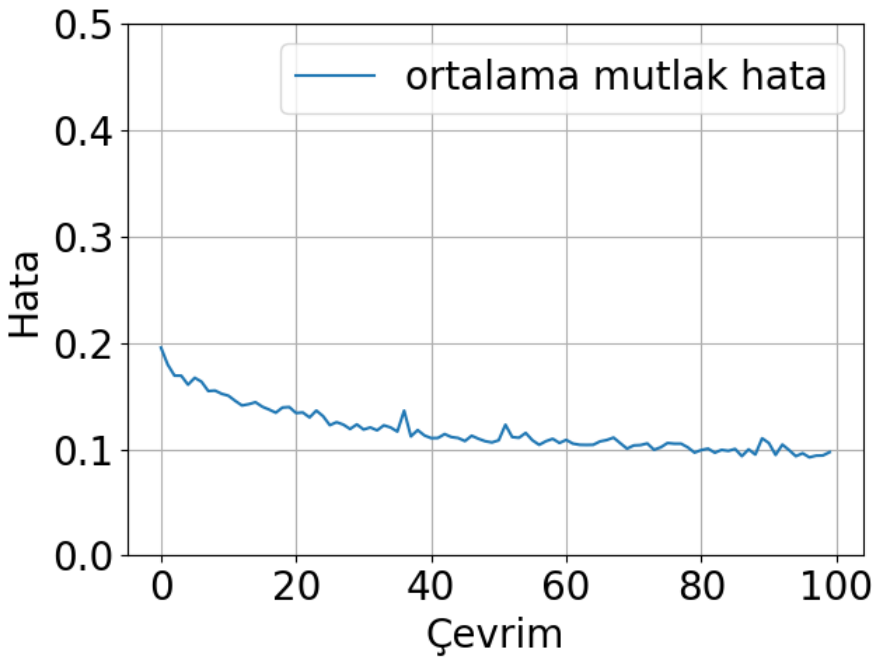
Şekil 4.49: a) Y yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği.

Y yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği Şekil 4.50'de incelenebilmektedir. Hata hesabı yapılırken robot kolu flanşının ivme verileri kullanılmıştır.



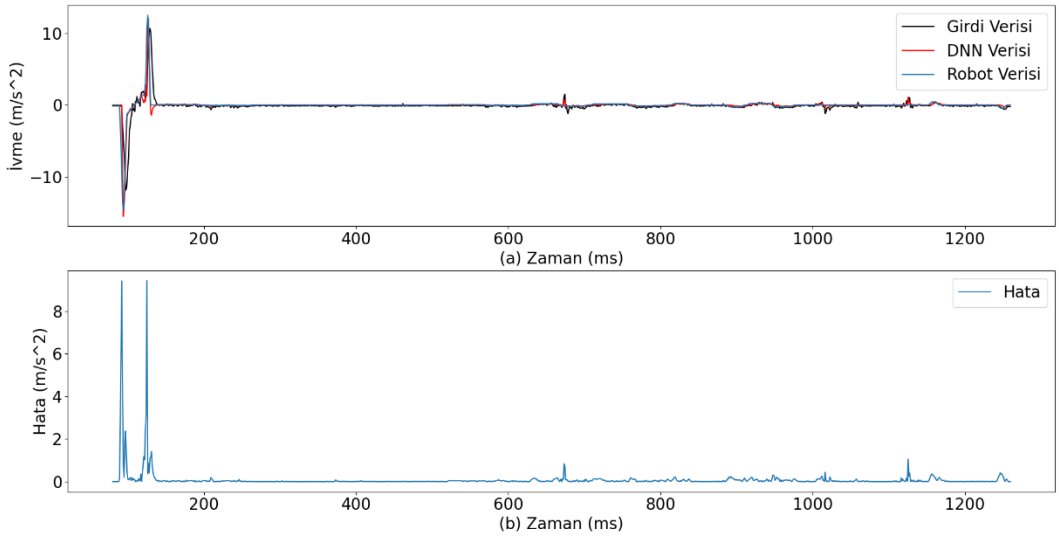
Şekil 4.50: Y yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği.

Z yönlü ivme tahmini modeli için her eğitim çevriminden sonra ulaşılan ortalama mutlak hata değerleri Şekil 4.51'deki grafikten görülebilmektedir.



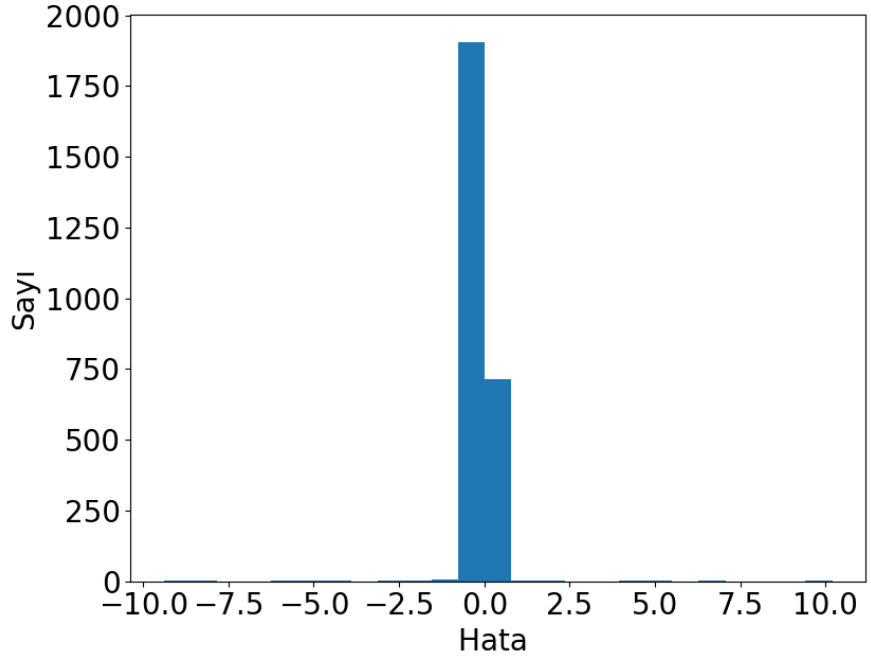
Şekil 4.51: Z ivmesi modeli için çevrim – ortalama mutlak hata grafiği.

Bu model eğitiminde ortalama mutlak hata değeri 0.0828'e düşürülmüştür. Modelden hesaplanan değerlerin karşılaştırmalı grafiği Şekil 4.52'te görülebilmektedir. Şekil 4.52.a'da Z yönlü ivme için eğitilen modelden hesaplanan ivme grafiği, filtrelenmiş ve modelde girdi olarak kullanılan ivme grafiği ve robot kolu flanşından ölçülen ivme grafiği görülmektedir. Şekil 4.52.b'de modelden hesaplanan ivme ile robottan ölçülen ivme arasındaki hata miktarını gösteren grafik görülmektedir.



Şekil 4.52: a) Z yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği.

Z yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği Şekil 4.53'da mevcuttur. Hata hesabı yapılırken robot kolu flanşının ivme verileri kullanılmıştır.

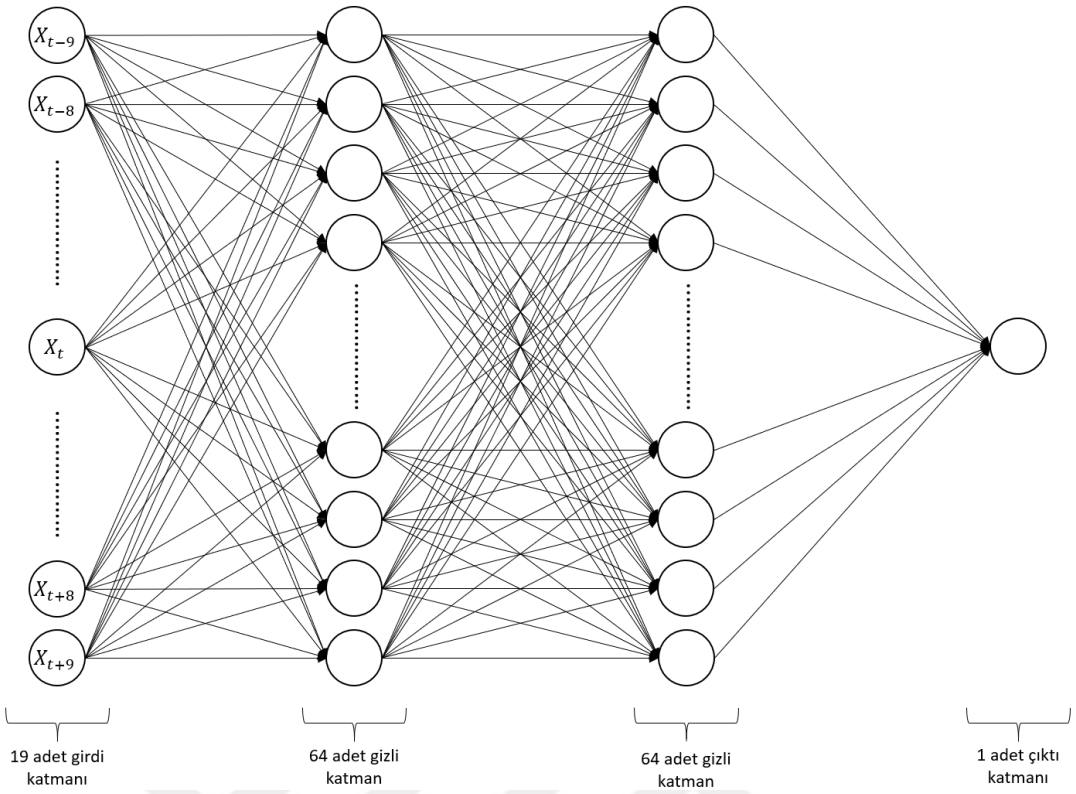


Şekil 4.53: Z yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği.

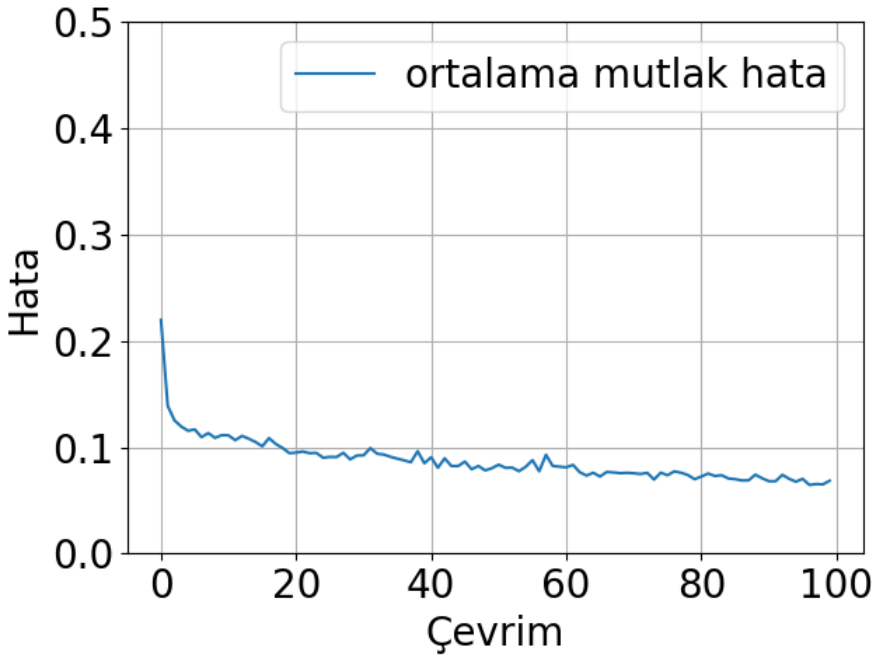
4.2.3 Üçüncü model: geçmiş ve gelecek veriler ile birlikte 19 girdili DNN regresyon modeli

Ondokuz girdili model eğitilmeden önce veriler normalize edildi. Bu işlem için tensorflow kütüphanesinin "Normalization" fonksiyonu kullanıldı. Model ondokuz girdi katmanı, iki adet altmışdört parametreli gizli katman ve bir çıktı katmanı olacak şekilde inşa edildi (Şekil 4.54). Böylelikle toplamda 5544 parametre oluşturuldu, bunların 5505 tanesi eğitilebilir parametredir. Model derlenirken ortalama mutlak hatanın 0.001'e ulaşması hedef gösterildi ve eğitimin 100 çevrim ile yapılması ayarlandı. Ondokuz girdiden t anındaki ivmeye karşılık, t anında sensörden ölçülen değer ile birlikte t anından önce ölçülen dokuz değer ve t anından sonra ölçülen dokuz değer verilmiştir.

X yönlü ivme tahmini modeli için her eğitim çevriminden sonra ulaşılan ortalama mutlak hata değerleri Şekil 4.55'teki grafikte mevcuttur.



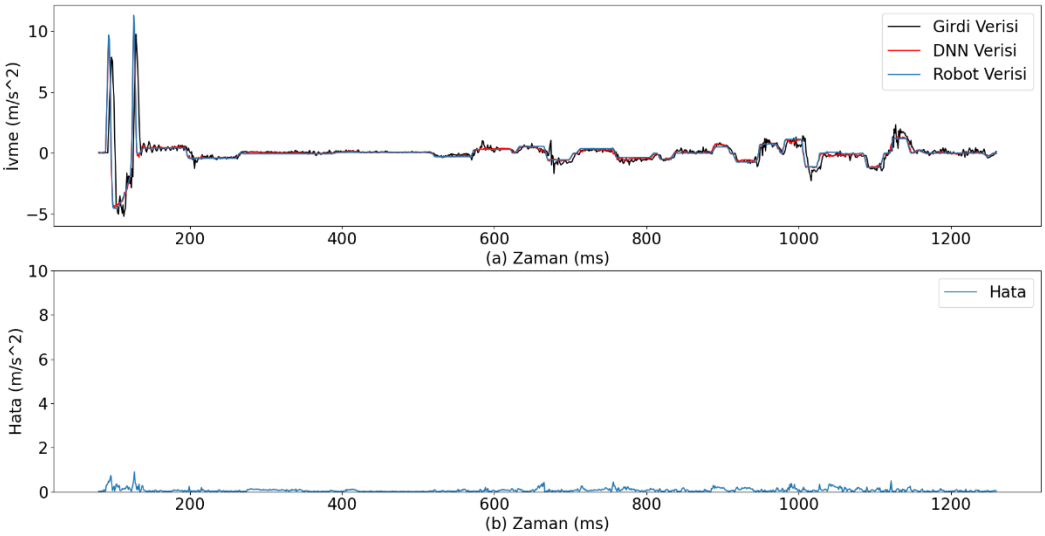
Şekil 4.54: On dokuz girdili derin sinir ağı modeli.



Şekil 4.55: X ivmesi modeli için çevrim – ortalama mutlak hata grafiği.

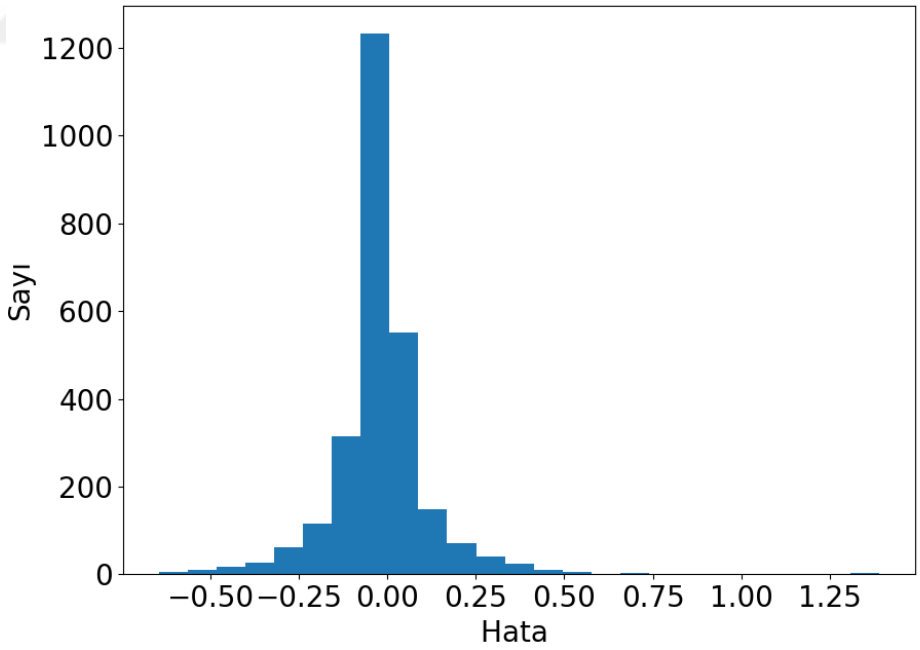
Bu model eğitiminde ortalama mutlak hata değeri 0.0794'e düşürülmüştür. Modelden hesaplanan değerlerin karşılaştırmalı grafiği Şekil 4.56'da mevcuttur. Şekil 4.56.a'da X yönlü ivme için eğitilen modelden hesaplanan ivme grafiği, filtrelenmiş ve modelde girdi olarak kullanılan ivme grafiği ve robot kolu flanşından ölçülen ivme grafiği

görülmektedir. Şekil 4.56.b'de modelden hesaplanan ivme ile robottan ölçülen ivme arasındaki hata miktarını gösteren grafik görülmektedir.



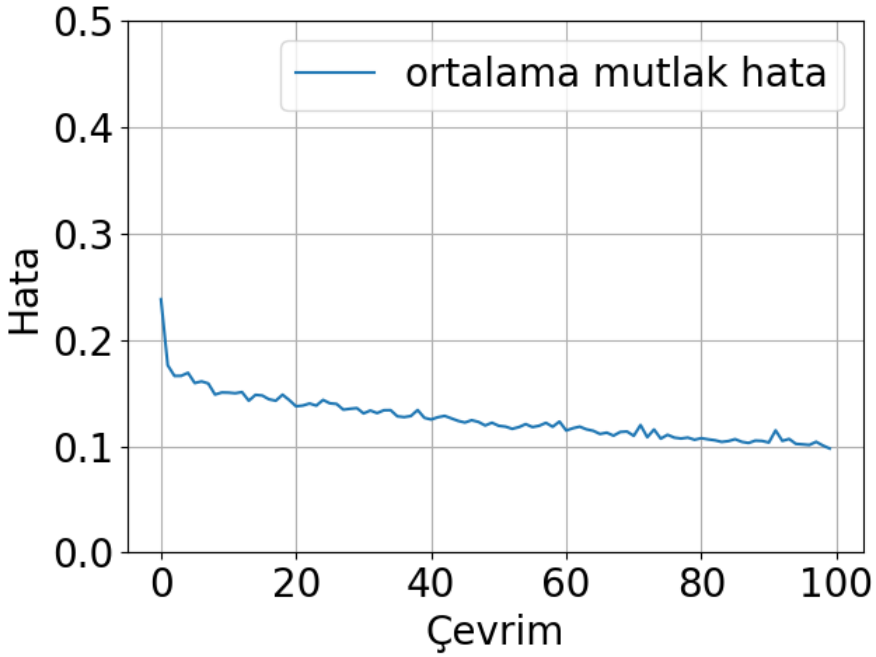
Şekil 4.56: a) X yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği.

X yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği Şekil 4.57'de görülebilmektedir. Hata hesabı yapılırken robot kolu flanşının ivme verileri kullanılmıştır.



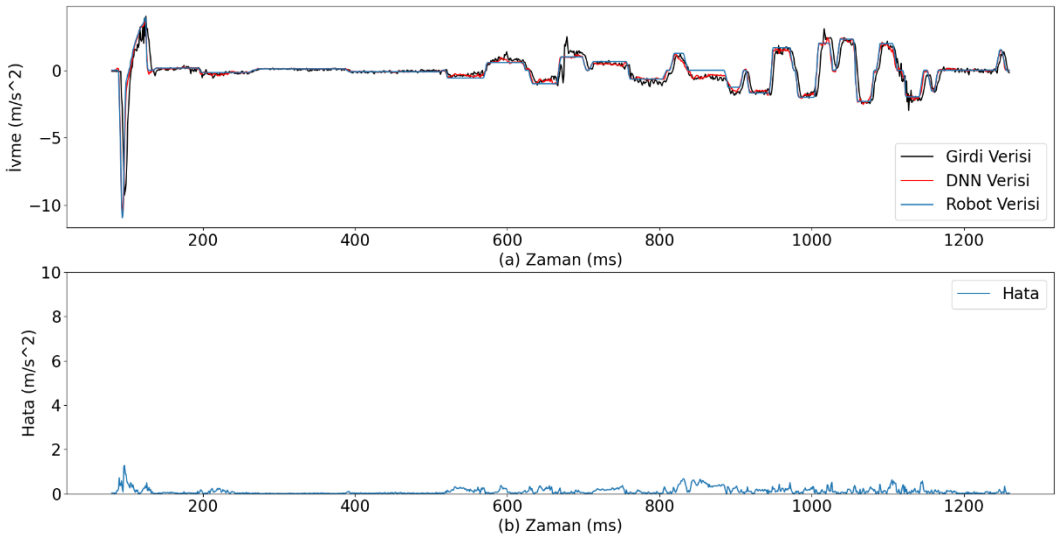
Şekil 4.57: X yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği.

Y yönlü ivme tahmini modeli için her eğitim çevriminden sonra ulaşılan ortalama mutlak hata değerlerini Şekil 4.58'deki grafikte incelemek mümkündür.



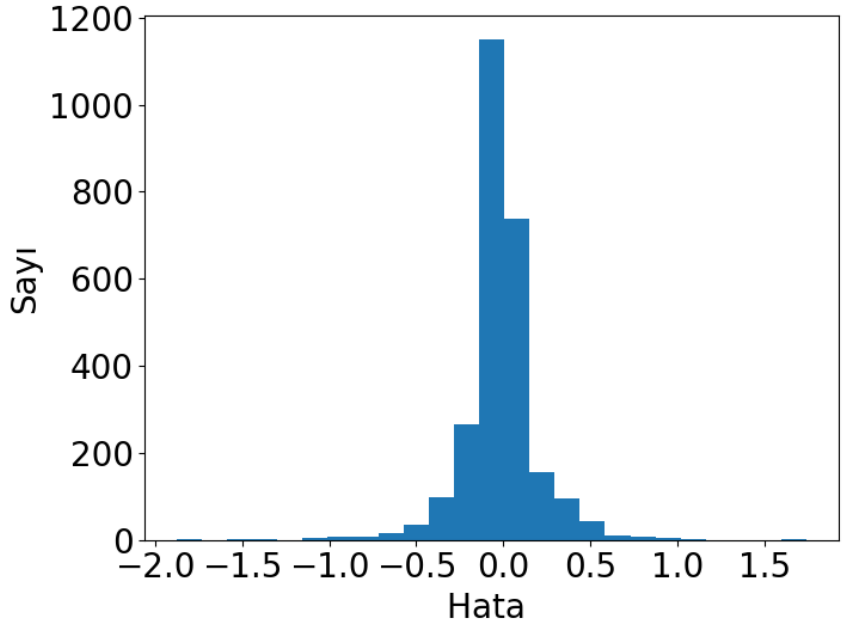
Şekil 4.58: Y ivmesi modeli için çevrim – ortalama mutlak hata grafiği.

Bu model eğitiminde ortalama mutlak hata değeri 0.1217'e düşürülmüştür. Modelden hesaplanan değerlerin karşılaştırmalı grafiği Şekil 4.59'da mevcuttur. Şekil 4.59.a'da Y yönlü ivme için eğitilen modelden hesaplanan ivme grafiği, filtrelenmiş ve modelde girdi olarak kullanılan ivme grafiği ve robot kolu flanşından ölçülen ivme grafiği görülmektedir. Şekil 4.59.b'de modelden hesaplanan ivme ile robottan ölçülen ivme arasındaki hata miktarını gösteren grafik görülmektedir.



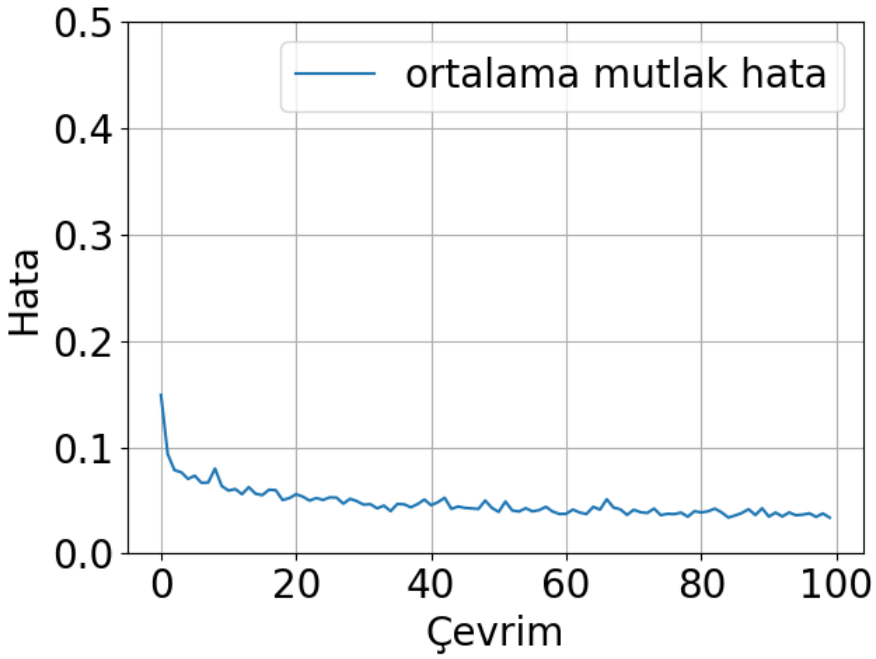
Şekil 4.59: a) Y yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği.

Y yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği, Şekil 4.60'da görmek mümkündür. Hata hesabı yapılırken robot kolu flanşının ivme verileri kullanılmıştır.



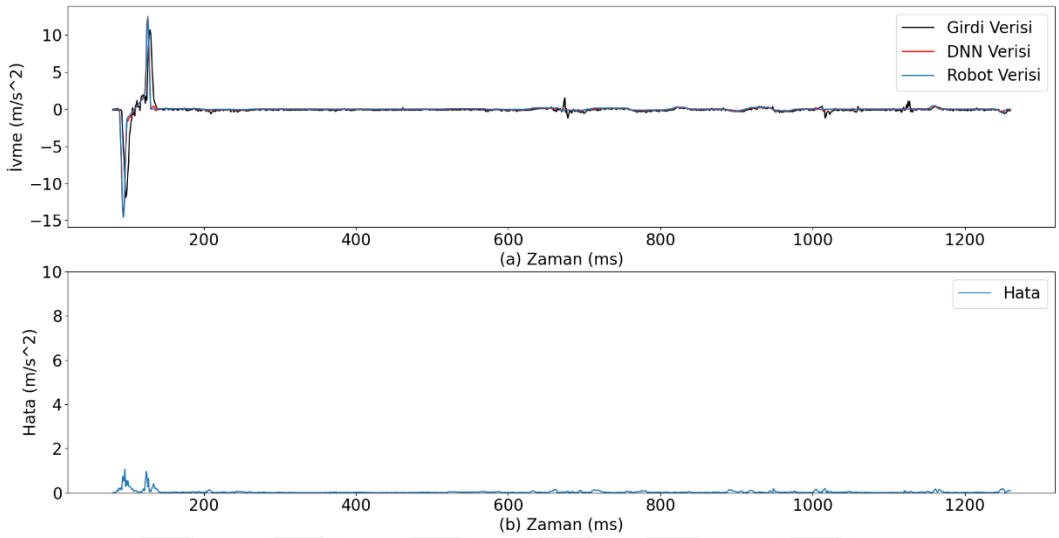
Şekil 4.60: Y yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği.

Z yönlü ivme tahmini modeli için her eğitim çevriminden sonra ulaşılan ortalama mutlak hata değerleri Şekil 4.61'deki grafikten görülebilmektedir.



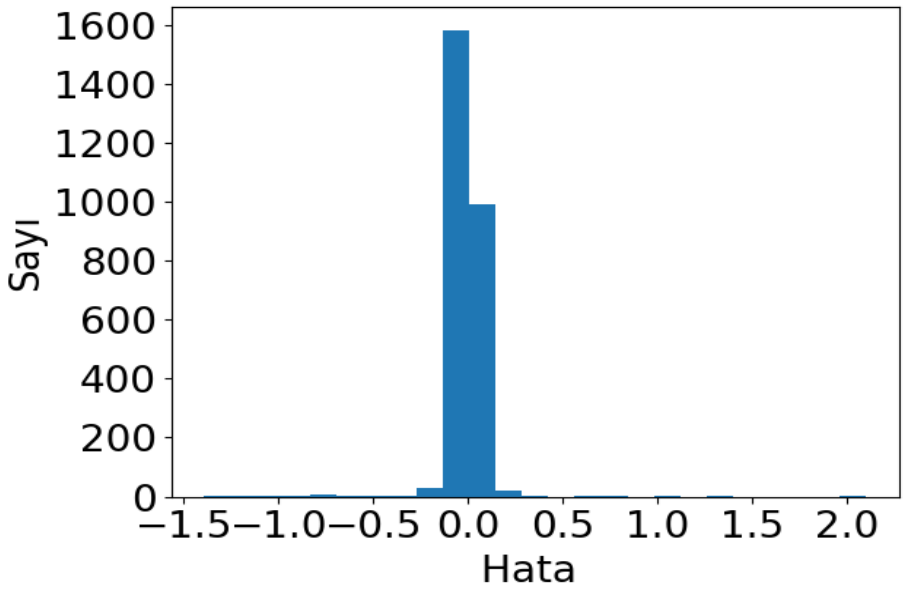
Şekil 4.61: Z ivmesi modeli için çevrim – ortalama mutlak hata grafiği.

Bu model eğitiminde ortalama mutlak hata değeri 0.0378'e düşürülmüştür. Modelden hesaplanan değerlerin karşılaştırmalı grafiği Şekil 4.62'te mevcuttur. Şekil 4.62.a'da Z yönlü ivme için eğitilen modelden hesaplanan ivme grafiği, filtrelenmiş ve modelde girdi olarak kullanılan ivme grafiği ve robot kolu flanşından ölçülen ivme grafiği görülmektedir. Şekil 4.62.b'de modelden hesaplanan ivme ile robottan ölçülen ivme arasındaki hata miktarını gösteren grafik görülmektedir.



Şekil 4.62: a) Z yönlü ivme değerleri için karşılaştırma grafiği, b) Hata grafiği.

Z yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği Şekil 4.63'te görülebilmektedir. Hata hesabı yapılırken robot kolu flanşının ivme verileri kullanılmıştır.



Şekil 4.63: Z yönlü ivme verileri için geliştirilen modelden hesaplanan değerlerin hata dağılım grafiği.

5. İYİLEŞTİRİLEN VERİNİN KULLANIMI

Her sayısal filtre ve makine öğrenmesi uygulamasından sonra, uygulamaların kıyaslanabilmesi için sonuçların ortalama mutlak hataları hesaplanmıştır. Ortalama mutlak hata değerleri kıyaslanınca en yüksek hassasiyete, makine öğrenmesi uygulamasında eğitilen üçüncü model olan, geçmiş ve gelecek verilerinin birlikte kullanılarak oluşturulduğu derin sinir ağı regresyon modeli olmuştur. Tüm uygulamaların ortalama mutlak hata değerlerinin kıyaslaması Çizelge 5.1'de mevcuttur.

Çizelge 5.1: Tüm Uygulamalara Ait Ortalama Mutlak Hata Değerleri.

		Ham Veri	Kalman filtresi	Medyan filtresi	Koşullu ortalama filtresi	1.DNN Modeli	2.DNN Modeli	3.DNN Modeli
X	1.Sensör	0.4285	0.3148	0.2827	0.2805	0.2288	0.1295	0.0794
	2.Sensör	0.3978	0.2885					
	3.Sensör	0.5444	0.4280					
Y	1.Sensör	0.3899	0.3437	0.3174	0.3165	0.2915	0.2086	0.1217
	2.Sensör	0.3569	0.3055					
	3.Sensör	0.5260	0.4677					
Z	1.Sensör	0.3228	0.2742	0.2597	0.2638	0.1527	0.0828	0.0378
	2.Sensör	0.3807	0.3640					
	3.Sensör	0.3730	0.3105					

Üçüncü modelin iyileştirme oranlarına bakıldığında, X yönlü ivme verilerinde %81.47, Y yönlü ivme verilerinde %68.78 ve Z yönlü ivme verilerinde %89.86 seviyelerine çıktığı görülmüştür.

AÖB sensörlerinin ivme verilerinde yapılan bu iyileştirmelerden sonra, çalışmanın asıl amacı olan ivme verilerinden hız ve konum tahminine bu iyileştirmelerin etkisi kıyaslanmıştır. Yine kıyaslama yaparken referans olarak robot kolu flanşından ölçülen

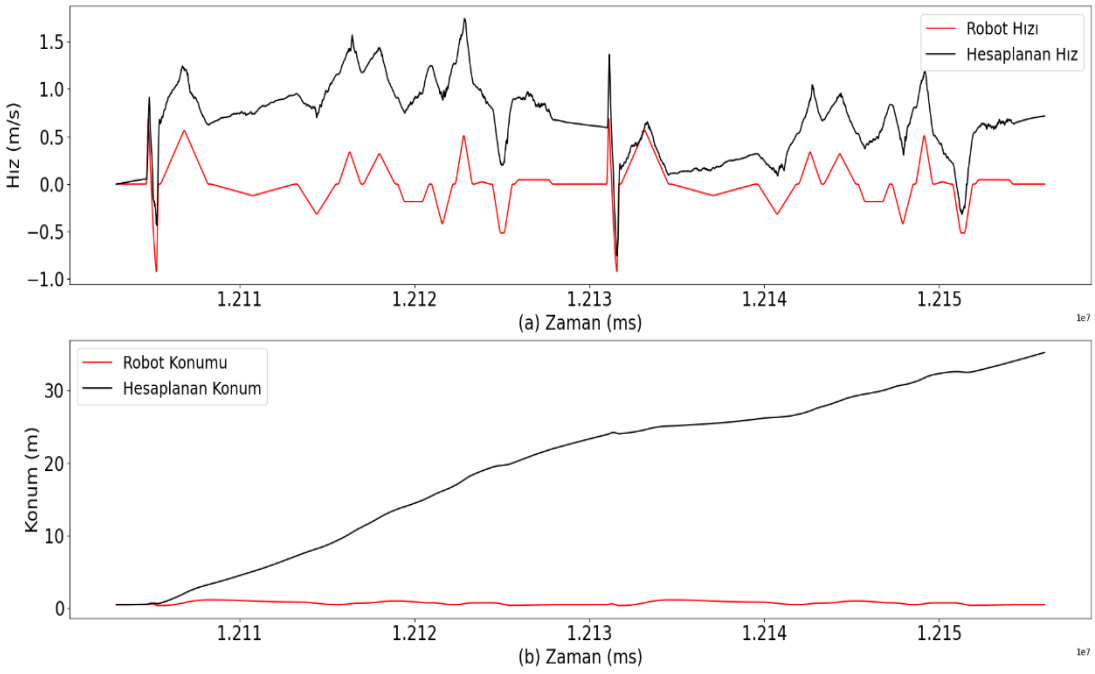
hız ve konum değerleri kullanılmıştır. İvme değerinden hız hesabı yaparken 5.1'deki formül, hız değerinden konum hesabı yaparken 5.2'deki formül kullanılmıştır.

$$\overrightarrow{V_k} = \overrightarrow{V_{k-1}} + \overrightarrow{a_k}(t_k - t_{k-1}) \quad (5.1)$$

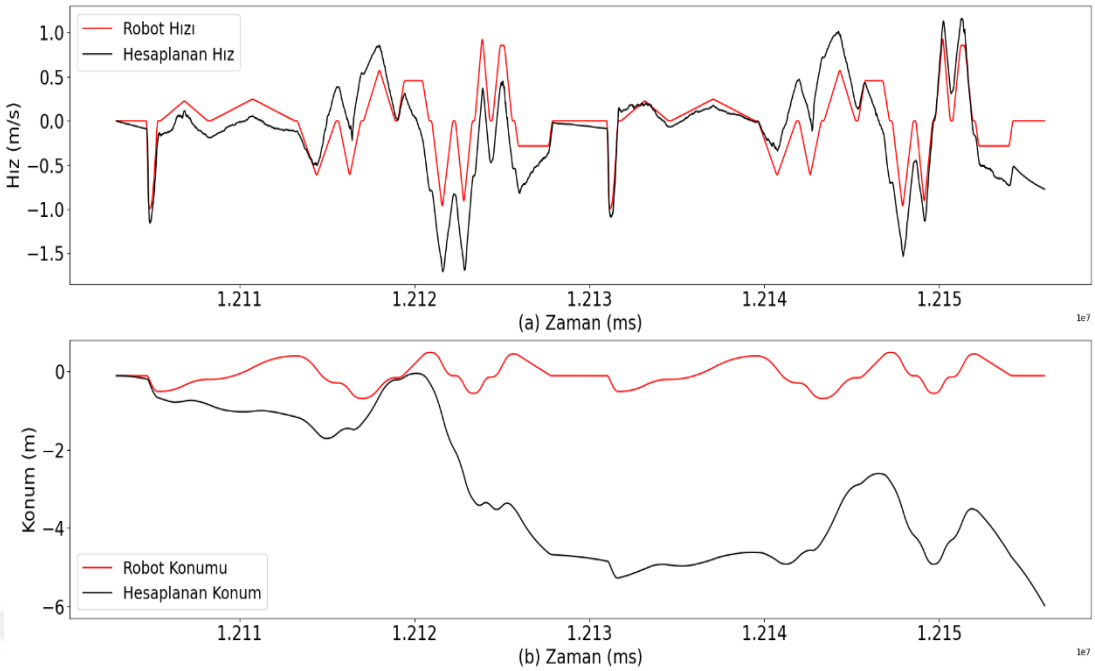
$$\overrightarrow{X_k} = \overrightarrow{X_{k-1}} + \overrightarrow{V_k}(t_k - t_{k-1}) \quad (5.2)$$

$\overrightarrow{V_k}$, k anındaki hız değerini temsil etmektedir. $\overrightarrow{a_k}$, k anındaki ivme değerini temsil etmektedir. $\overrightarrow{X_k}$, k anındaki konum değerini temsil etmektedir. t_k, k anındaki zamanı temsil etmektedir.

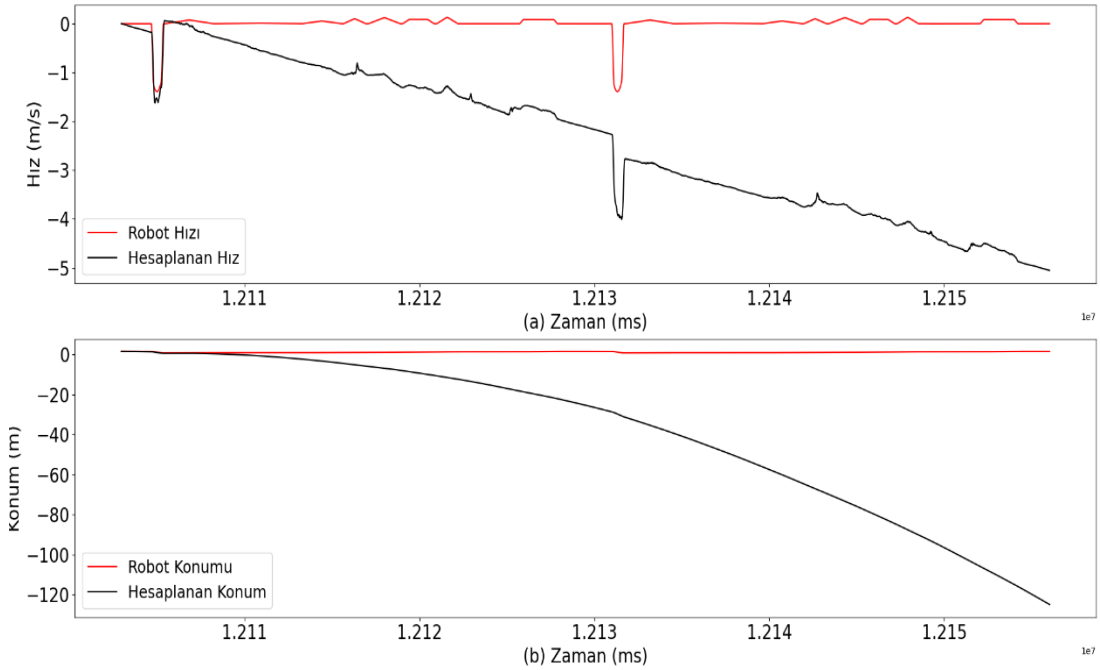
Hız ve ivme değerlerindeki iyileştirmeleri kıyaslamak için, 1. AÖB sensöründen ölçülen ivme değerleri, tüm filtrelerin kullanılarak iyileştirme yapılmış ivme değerleri ve üçüncü model ile iyileştirme yapılmış ivme değerleri kullanılmıştır. İntegral hesaplamalardan dolayı hata değerleri giderek büyümektedir. Bu sebeple değerlerin ortalama mutlak hatalarının yanında en son değerlerin hataları da kıyaslanmıştır. Hesaplanan hız ve konum değerleri için karşılaştırmalı grafikleri Şekil 5.1 ve 5.9 arası görmek mümkündür.



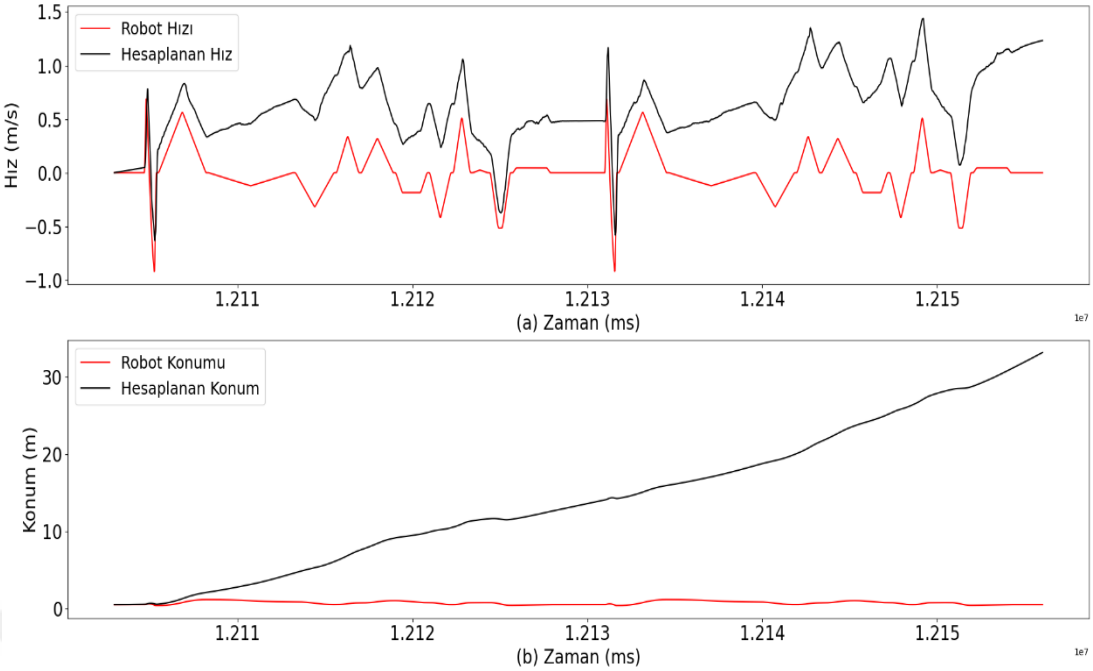
Şekil 5.1: a) 1.AÖB sensörünün X yönlü ivme verileri kullanılarak hesaplanan hız grafiği. b) 1.AÖB sensörünün X yönlü ivme verileri kullanılarak hesaplanan konum grafiği.



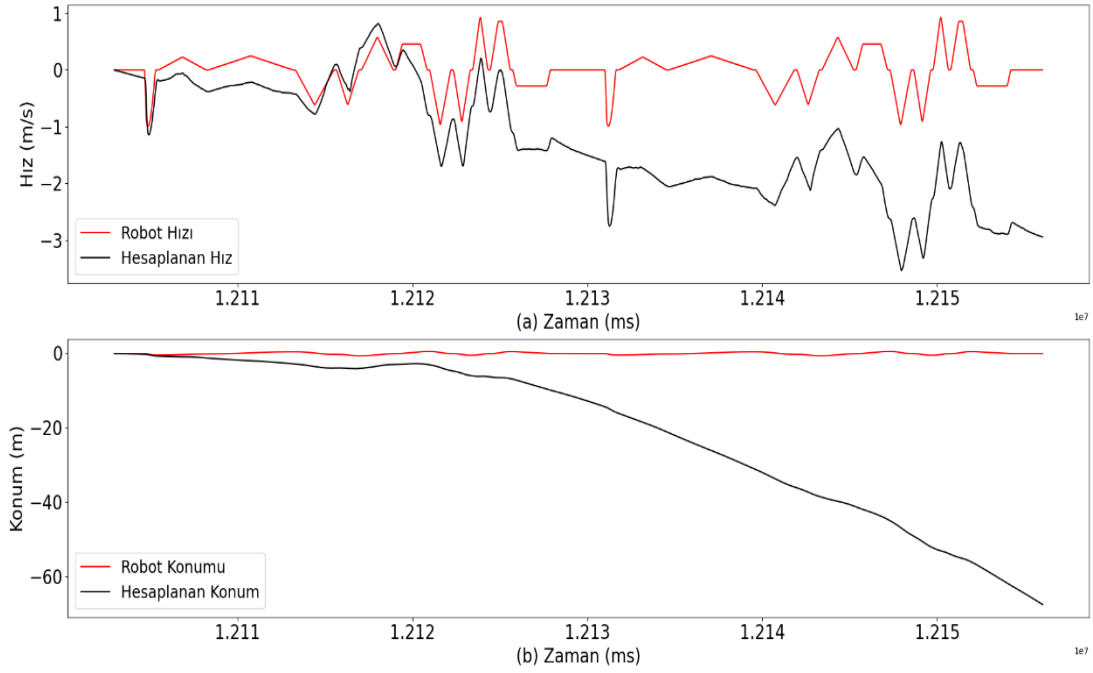
Şekil 5.2: a) 1.AÖB sensörünün Y yönlü ivme verileri kullanılarak hesaplanan hız grafiği. b) 1.AÖB sensörünün Y yönlü ivme verileri kullanılarak hesaplanan konum grafiği.



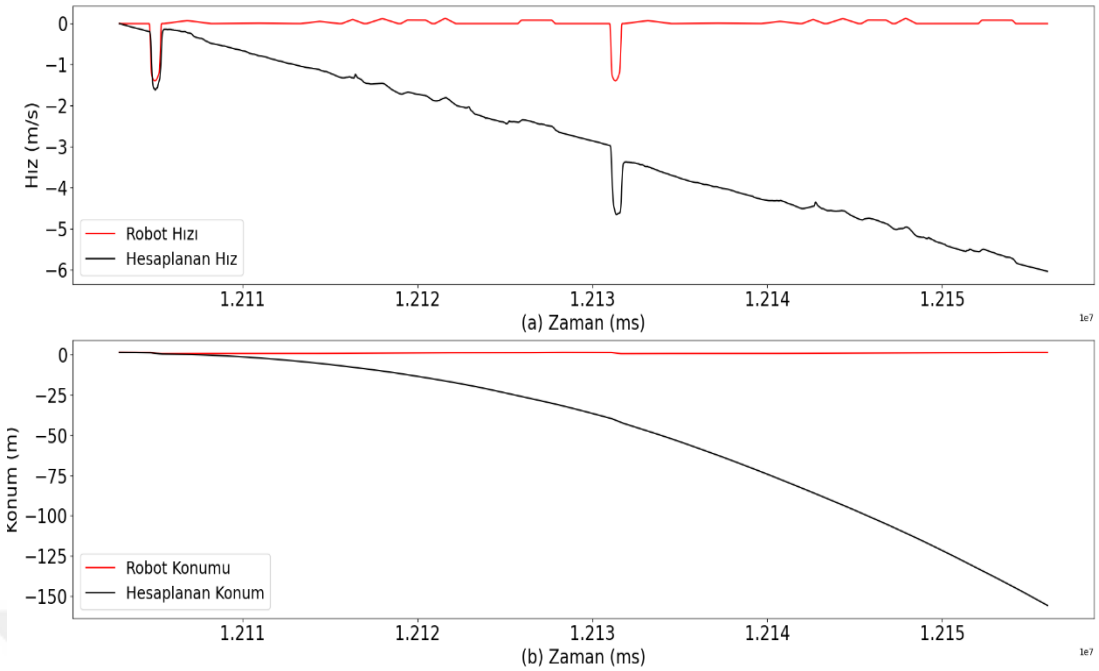
Şekil 5.3: a) 1.AÖB sensörünün Z yönlü ivme verileri kullanılarak hesaplanan hız grafiği. b) 1.AÖB sensörünün Z yönlü ivme verileri kullanılarak hesaplanan konum grafiği.



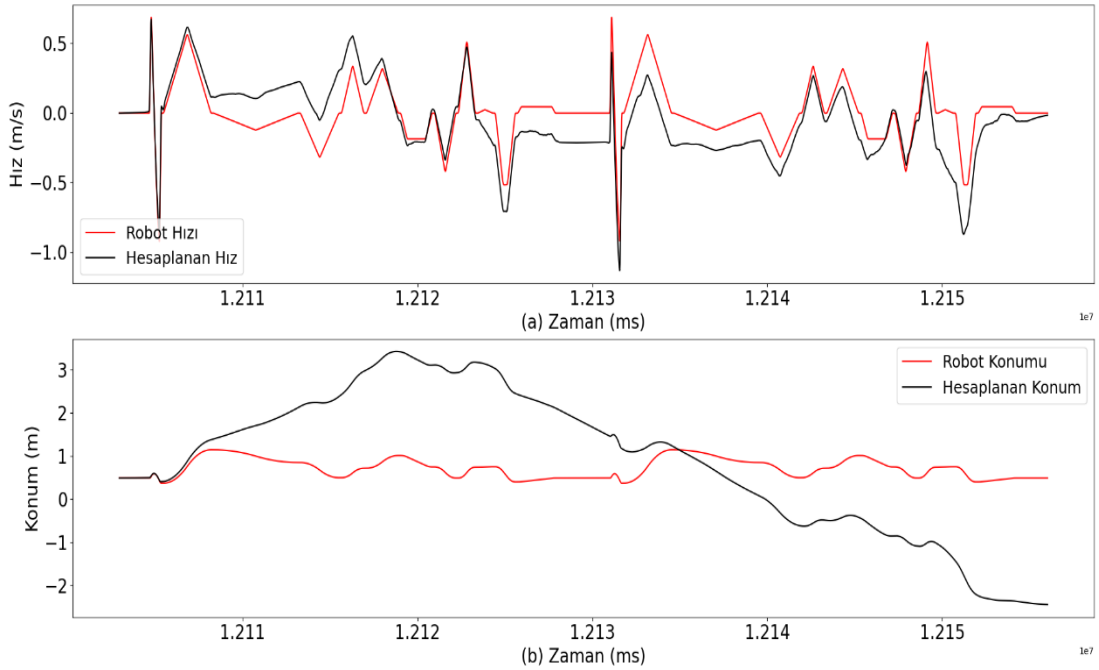
Şekil 5.4: a) X yönlü filtrelenmiş ivme verileri kullanılarak hesaplanan hız grafiği. b) X yönlü filtrelenmiş ivme verileri kullanılarak hesaplanan konum grafiği.



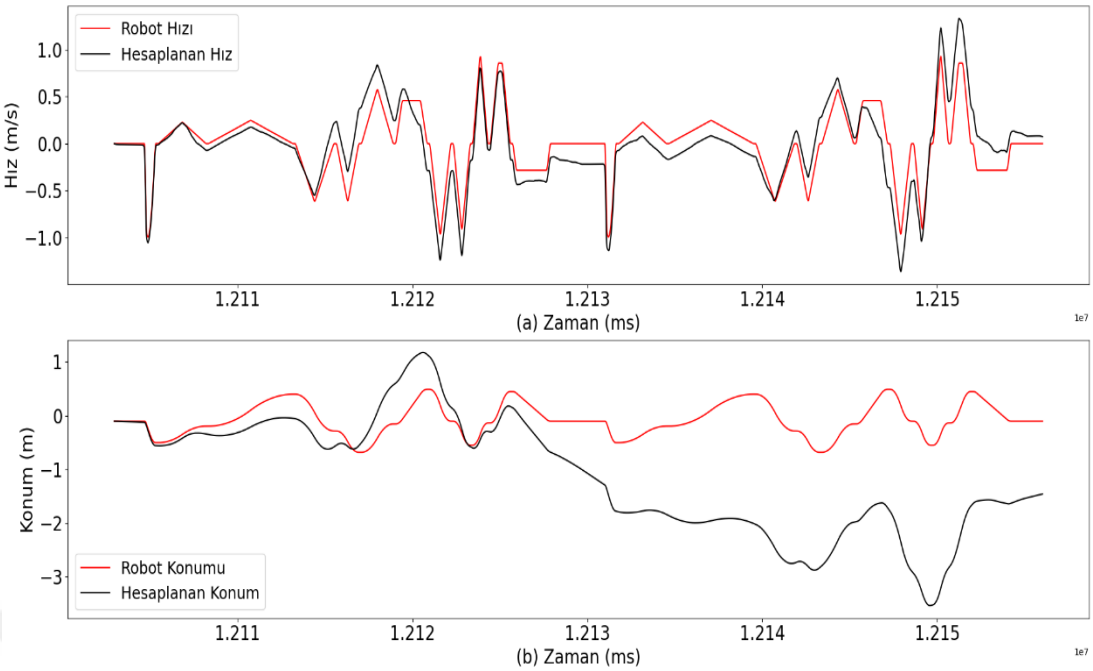
Şekil 5.5: a) Y yönlü filtrelenmiş ivme verileri kullanılarak hesaplanan hız grafiği. b) Y yönlü filtrelenmiş ivme verileri kullanılarak hesaplanan konum grafiği.



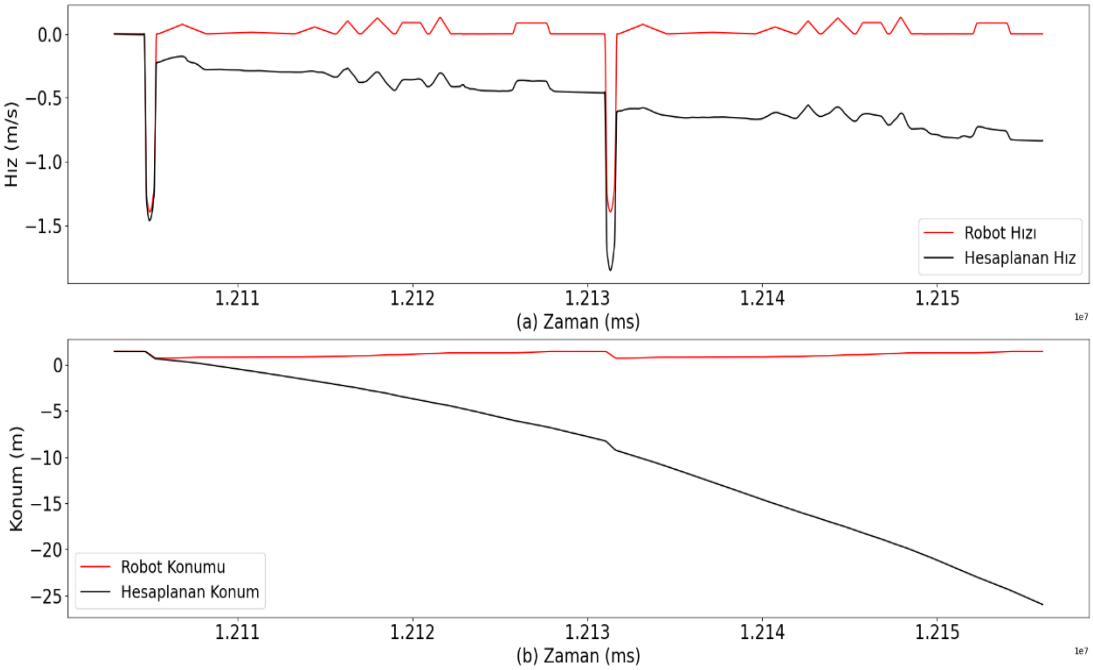
Şekil 5.6: a) Z yönlü filtrelenmiş ivme verileri kullanılarak hesaplanan hız grafiği. b) Z yönlü filtrelenmiş ivme verileri kullanılarak hesaplanan konum grafiği.



Şekil 5.7: a) X yönlü 3. DNN modeliyle iyileştirilmiş ivme verileri kullanılarak hesaplanan hız grafiği. b) X yönlü 3. DNN modeliyle iyileştirilmiş ivme verileri kullanılarak hesaplanan konum grafiği.



Şekil 5.8: a) Y yönlü 3. DNN modeliyle iyileştirilmiş ivme verileri kullanılarak hesaplanan hız grafiği. b) Y yönlü 3. DNN modeliyle iyileştirilmiş ivme verileri kullanılarak hesaplanan konum grafiği.



Şekil 5.9: a) Z yönlü 3. DNN modeliyle iyileştirilmiş ivme verileri kullanılarak hesaplanan hız grafiği. b) Z yönlü 3. DNN modeliyle iyileştirilmiş ivme verileri kullanılarak hesaplanan konum grafiği.

Hesaplanan hız değerlerine ait ortalama mutlak hata değerlerini ve son değerlerin mutlak hata değerlerini Çizelge 5.2'de incelemek mümkündür.

Çizelge 5.2: Hız verilerine ait ortalama mutlak hatalar ve son değerlerin mutlak hatası.

	1. AÖB sensörüne ait ham hız verileri		Filtrelenmiş hız verileri		3. Model ile iyileştirilmiş hız verileri	
	Ortalama mutlak hata	Son değerin mutlak hatası	Ortalama mutlak hata	Son değerin mutlak hatası	Ortalama mutlak hata	Son değerin mutlak hatası
X	0.65457	0.714640	0.617191	1.232877	0.15228	0.01543
Y	0.27883	0.774360	1.296094	2.947446	0.16646	0.07248
Z	2.37474	5.055920	2.954280	6.040636	0.51438	0.83482

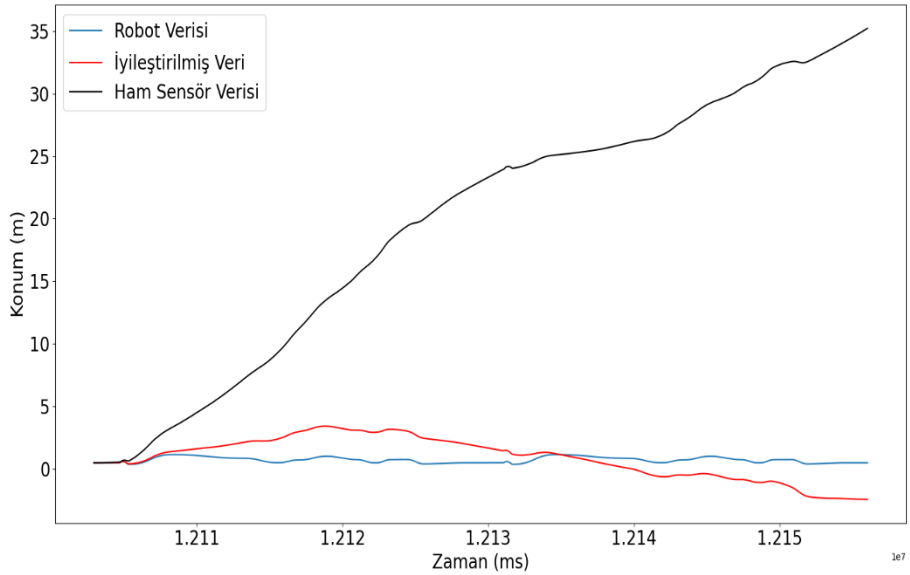
Üçüncü model ile iyileştirilen ivme verileri ile hesaplan hız değerlerinin ortalama mutlak hatalarına göre iyileşme oranları X yönünde %76.73, Y yönünde %40.30 ve Z yönünde %78.33'tür, son değerlerin mutlak hatalarına göre hesaplanınca iyileşme oranları X yönünde %97.84, Y yönünde % 90.64 ve Z yönünde %83.48'dir.

Hesaplanan konum değerlerine ait ortalama mutlak hata değerlerini ve son değerlerin mutlak hata değerlerini Çizelge 5.3'de incelemek mümkümdür.

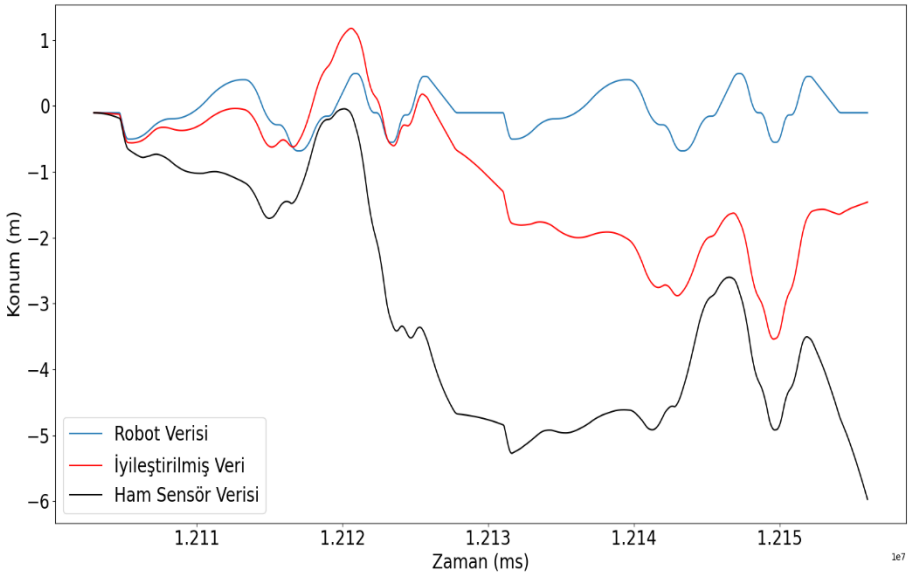
Çizelge 5.3: Konum verilerine ait ortalama mutlak hatalar ve son değerlerin mutlak hatası.

	1. AÖB sensörüne ait ham konum verileri		Filtrelenmiş konum verileri		3. Model ile iyileştirilmiş konum verileri	
	Ortalama mutlak hata	Son değerin mutlak hatası	Ortalama mutlak hata	Son değerin mutlak hatası	Ortalama mutlak hata	Son değerin mutlak hatası
X	18.806656	34.736607	13.54470	32.71391	1.377581	2.92739
Y	2.94752	5.86956	20.66824	67.39546	1.154819	1.35675
Z	39.639033	126.29403	50.92625	157.1859	10.54246	27.3597

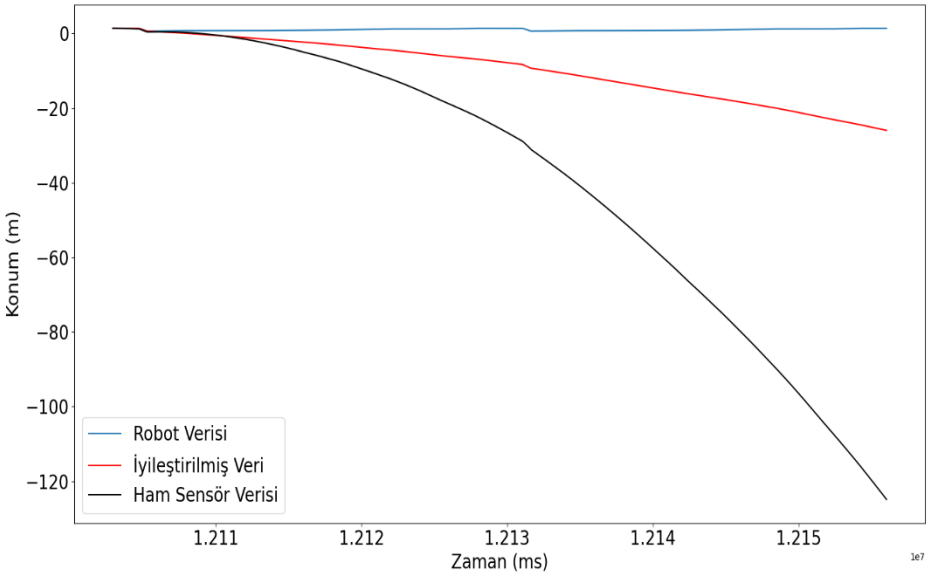
Üçüncü model ile iyileştirilen ivme verileri ile hesaplanan konum değerlerinin ortalama mutlak hatalarına göre iyileşme oranları X yönünde %92.67, Y yönünde %60.82, Z yönünde %73.4'tür, son değerlein mutlak hatalarına göre hesaplanınca iyileşme oranları X yönünde %91.57, Y yönünde %76.88 ve Z yönünde %78.33'tür. İyileştirmelerin konum hesaplamalarına etkileri Şekil 5.10-5.12 arası grafiklerde görülmektedir. Grafiklerde endüstriyel robottan çekilen konum verisi, ham sensör ivme verisi kullanılarak hesaplanan konum verisi ve üçüncü model ile iyileştirilen ivme verisi kullanılarak hesaplanan konum verisi karşılaştırmalı olarak gösterilmiştir.



Şekil 5.10: X yönlü konum değerlerine ait karşılaştırma grafiği.



Şekil 5.11: Y yönlü konum değerlerine ait karşılaştırma grafiği.



Şekil 5.12: Z yönlü konum değerlerine ait karşılaştırma grafiği.

6. SONUÇLAR VE ÖNERİLER

Tez kapsamında, sayısal filtreler ve makine öğrenmesi uygulamaları ile ivme sensörlerinin hassasiyetleri arttırılmaya çalışılmış ve bu sayısal filtrelerin ve makine öğrenmesi uygulamalarının, ivme verilerine etkileri incelenmiştir. Bu amaçla kalman filtresi, medyan filtresi ve ortalama filtresi uygulanmıştır. Makine öğrenmesi uygulaması olarak DNN regresyon yöntemi ile üç farklı model eğitilmiştir.

Kalman filtresi, her sensör için hesaplanan varyans değeri kullanılarak ayrı ayrı uygulanmıştır ve gürültüleri azaltmada etkisi olmuştur. Sensör sayıları arttırılarakta olasılıksal hata ihtimali azaltılmıştır. Bunu sağlamak için tüm sensör verileri medyan filtereden ve koşullu ortalama filtreden geçirilmiştir.

Makine öğrenmesi yaklaşımı ile oluşturulan iyileştirme modelleri kıyaslandığında en yüksek hassasiyete DNN regresyon yöntemiyle eğitilen üçüncü modelde rastlanmıştır. Üçüncü model eğitilirken geçmiş ve gelecek veriler bir arada kullanılarak ondokuz girdili bir derin sinir ağı modeli oluşturulmuştur. Üçüncü model ile iyileştirilen ivme verileri kullanılarak hesaplanan hız ve konum verilerinde de en yüksek iyileştirme oranına ulaşılmıştır.

Üçüncü modelin derin sinir ağı modelinde dokuz adet gelecek verinin kullanılması, anlık ivme hesabında yaklaşık 135 ms'lik bir gecikmeye sebep olmaktadır. Sebebi sensörlerin veri gönderme aralığı ortalama 15 ms'dir. Bu gecikme bazı uygulamalar için yeterli gelebilir. Bunun haricinde, sensör hareketi tamamlandıktan sonra toplanan ivme verileri iyileştirilebilir.

İleride yapılacak çalışmalar için farklı firmalara ait sensörlerle aynı denemelerin yapılması önerilmektedir. Veri gönderme aralığı daha kısa süren bir sensör kullanılarak anlık hesaplamalardaki gecikme süresi azaltılabilir. Bunun yanında, farklı g ölçüm aralıklarına ait birkaç sensörün birlikte kullanılarak füzyon edilmesi ve makine öğrenmesi ile model eğitilirken tüm sensörlerdeki değerlerin kullanılması, sonuçlarda faydalı olabilir. Yazılım tarafında ise ivme değerlerine uygulanan iyileştirmelerin hesaplanan hız ve konum değerlerine uygulanması, hız ve konum değerleri için makine öğrenmesi modeli eğitilirken hız ve konum değerlerinin yanında

ivme değerlerinin ve zaman değerinin kullanılması hız ve konum sonuçlarında faydalı olacağı düşünülmektedir. Son olarak ivme iyileştirme modelinin pekiştirmeli öğrenme ile tekrar eğitilmesi uzun süreli hesaplamalarda daha hassas sonuçlar sağlayabilir.

KAYNAKLAR

[1] **Arıkan, S.** (t.y.). Endüstriyel Robotlar Ve Üretimde Uygulama Alanları (Ders Notları), Orta Doğu Teknik Üniversitesi, Makina Mühendisliği Bölümü, Ankara.

[2] **Kaya, B.** (2020). *Geçmişten günümüze robotlar*, Erişim: 25 Ocak 2021, https://blog.quicksigorta.com/teknoloji/gecmisten-gunumuze-robotlar-1566.

[3] **Karatay, C.** (2020). *DÜNYA ROBOTİK RAPORU YAYIMLANDI*, Erişim: 27 Ocak 2021, https://www.stendustri.com.tr/robot-yatirimlari/dunya-robotik-raporu-yayimlandi-h108331.html.

[4] **Heijdens, R.** (2017). *Implementing machine learning in industrial robots for better human-robot cooperation* (Master's thesis), University of Twente, Engineering Technology, Industrial Design BSc, Overijssel.

[5] **Ramon, J. A. C., Herias, F. A. C., & Medina, F. T.** (2008). Hybrid tracking of human operators using IMU/UWB data fusion by a Kalman filter, *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, HRI 2008, Amsterdam, The Netherlands, March 12-15.

[6] **Colgate, J. E., Peshkin, M. A., & Wannasuphoprasit, W.** (1996). Nonholonomic Haptic Display, *Proceedings of the IEEE International Conference on Robotics and Automation*, 1, (pp.539-544), USA: Minneapolis, April 22-28.

[7] **Pires, J. N.** (2004). Robot-by-voice: Experiments on commanding an industrial robot using the human voice, *Industrial Robot*, (32-6), 505-511, December 2004.

[8] **Tasevski, J., Nikolić, M., & Miskovic, D.** (2013). Integration of an Industrial Robot with the Systems for Image and Voice Recognition, *Serbian Journal of Electrical Engineering*, (10-1), 219-230, January 2013.

[9] **Lenz, C., Nair, S., Rickert, M., & Knoll, A.** (2008). Joint-Action for Humans and Industrial Robots for Assembly Tasks, *IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, Munich, August 1-3.

[10] **Simão, M., Neto, P., & Gibaru, O.** (2016). Natural control of an industrial robot using hand gesture recognition with neural networks, *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, Florence, October 23-26.

[11] **Gültekin, Y.** (2012). *Bir endüstriyel robotun insan kolu hareketlerinin ataletsel ölçümü ile uzaktan kontrolü* (Yüksek Lisans Tezi), TOBB Ekonomi ve

Teknoloji Üniversitesi, Fen Bilimleri Enstitüsü, Makine Mühendisliği Anabilim Dalı, Ankara.

[12] **Gelen, G., Özcan, S.** (2019). İnsan-robot etkileşiminin biyomimetik yaklaşımla sağlanması, *Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi*, (25-2),188-198.

[13] **Bannat, A., Gast, J., Rehrl, T., Rösel, W., Rigoll, G., & Wallhoff, F.** (t.y.). *A Multimodal Human-Robot-Interaction Scenario: Working Together with an Industrial Robot*, Technical University of Munich, Faculty of Mechanical Engineering, Munich.

[14] **Liu, H., Fang, T., Zhou, T., Wang, Y., & Wang, L.** (2018). Deep Learning-based Multimodal Control Interface for Human-Robot Collaboration, *51st CIRP Conference on Manufacturing Systems*, Stockholm.

[15] **Gannon, M.** (2018). *MANUS: Exploring Pack Behaviors in Autonomous Robots*, Erişim: 25 Ocak 2021, https://atonaton.com/manus/.

[16] **Günhan, Y.** (2014). *Ataletsel ölçüm birimi hatalarının belirlenmesi ve düzeltilmesi* (Yüksek Lisans Tezi), Hacettepe Üniversitesi, Fen Bilimleri Enstitüsü, Elektrik-Elektronik Mühendisliği Anabilim Dalı, Ankara.

[17] **Mahoney, J. M., Rhudy, M. B.** (2019). Methodology and validation for identifying gait type using machine learning on IMU data, *Journal of Medical Engineering & Technology*, (43-1), 25-32 .

[18] **KUKA GmbH.** (2019). *KR 16 R1610*, Erişim: 25 Ocak 2021, https://www.kuka.com/-/media/kuka-downloads/imported/6b77eecacfe542d3b736af377562ecaa/0000262125_en.pdf.

[19] **KUKA GmbH.** (2009). *KUKA.RobotSensorInterface 2.3*, Erişim: 25 Ocak 2021, http://vip.gatech.edu/wiki/images/3/3c/KUKARobotSensorInterface.pdf .

[20] **Bosch Sensortec.** (2014). *Data sheet: BNO055 Intelligent 9-axis absolute orientation sensor*, Erişim: 25 Ocak 2021, https://cdn-shop.adafruit.com/datasheets/BST_BNO055_DS000_12.pdf.

[21] **İvmeölçer.** (2016). V*ikipedi* Erişim: 25 Ocak, 2021, https://tr.wikipedia.org/wiki/%C4%B0vme%C3%B6l%C3%A7er

[22] **İVMEÖLÇER ÇEŞİTLERİ.** (2015). Erişim: 25 Ocak 2021, https://elektronikhobi.net/ivmeolcer-cesitleri/.

[23] **ROHM Semiconductor.** *Accelerometer sensors*, Erişim: 25 Ocak 2021, https://www.rohm.com/electronics-basics/sensor/accelerometer-sensor.

[24] **Robotpark.** *ARDUİNO MEGA 2560 MİKRODENETLEYİCİ REV 3*, Erişim: 25 Ocak 2021, http://www.robotpark.com.tr/Arduino-MEGA-2560-Tr.

[25] **Sparkfun.** *Arduino Pro Mini 328 - 3.3V/8MHz*, Erişim: 25 Ocak 2021, https://www.sparkfun.com/products/11114.

[26] **Robotistan.** *FTDI Programlama Kartı*, Erişim: 25 Ocak 2021, https://www.robotistan.com/ftdi-programlama-karti-33v-5v-secilebilir.

[27] **Welch, G., Bishop, G.,** (2001). *An Introduction to the Kalman Filter*. SIGGRAPH 2001, Los Angeles, August 12-17.

[28] **Median filter.** (2020). V*ikipedi*. Erişim: 28 Ocak, 2021, https://en.wikipedia.org/wiki/Median_filter

[29] **ALKAN, M. A.** (t.y.). *Makine Öğrenimi Nedir ?*, Erişim: 28 Ocak 2021, https://www.endustri40.com/makine-ogrenimi-nedir/.

[30] **Meshine.** *Bilgisayarlı Görü*, Erişim: 28 Ocak 2021, https://www.meshine.tech/tr/services.html.

[31] **Seker, E. S.** (2019). *Yapay Zeka ile Borsa Analizi*, Erişim: 28 Ocak 2021, https://medium.com/@sadievrenseker/yapay-zeka-ile-borsa-analizi-57f7120171c1.

[32] **Alpaydın, E.** (2013). *Yapay Öğrenme* (2. bs.), İstanbul: Boğaziçi Üniversitesi Yayınevi

[33] **Yapay sinir ağları.** (2021). V*ikipedi*. Erişim: 28 Ocak, 2021, https://tr.wikipedia.org/wiki/Yapay_sinir_a%C4%9Flar%C4%B1

[34] **Zaloğlu, O.** (2015). *Dünyayı Değiştirmekte Olan Yapay Sinir Ağları Nedir ?*, Erişim: 28 Ocak 2021, https://bilimfili.com/dunyayi-degistirmekte-olan-yapay-sinir-aglari-nedir.

[35] **O'Reilly.** (t.y.). *The basic architecture of DNN*, Erişim: 28 Ocak 2021, https://www.oreilly.com/library/view/python-natural-language/9781787121423/fa56fade-f7c3-494e-807e-2e1f0970d9be.xhtml.

[36] **Ulgen, E. K.** (2018). *Python ile Kümeleme Algoritmaları (Makine Öğrenimi Bölüm-8)*, Erişim: 28 Ocak 2021, https://medium.com/@k.ulgen90/python-ile-k%C3%BCmeleme-algoritmalar%C4%B1-makine-%C3%B6%C4%9Frenimi-b%C3%B6l%C3%BCm-8-8204ffa702f2.

[37] **TensorFlow.** (2021). *Basic regression: Predict fuel efficiency*, Erişim: 28 Ocak 2021, https://www.tensorflow.org/tutorials/keras/regression?hl=en.

[38] **Townsend, K.** (t.y.). *Adafruit BNO055 Absolute Orientation Sensor*, Erişim: 28 Ocak 2021, https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor?view=all#assembly.

[39] **TextSpeak Design Group.** (2012). *Datasheet: Arduino Mega 2560*, Erişim: 28 Ocak 2021, http://scholar.ppu.edu/bitstream/handle/123456789/638/1.pdf?sequence=3&isAllowed=y.

[40] **Components101.** (2018). *Datasheet: Arduino Pro Mini*, Erişim: 28 Ocak 2021, https://components101.com/microcontrollers/arduino-pro-mini.

ÖZGEÇMİŞ

Ad-Soyad **:** Sinan ÖZCAN

ÖĞRENİM DURUMU:

- **Lisans** **:** 2017, Bursa Teknik Üniversitesi, Doğa Bilimleri, Mimarlık ve Mühendislik Fakültesi, Mekatronik Mühendisliği
- **Yüksek Lisans** **:** 2021, Bursa Teknik Üniversitesi, Elektrik - Elektronik Mühendisliği Anabilim Dalı

MESLEKİ DENEYİM VE ÖDÜLLER:

- Yazılım ve Robotik Arge Uzmanı – Eurobotik Otomasyon ve Görüntü İşleme Teknolojileri (Mart 2018 – devam ediyor)
- Arge & Tasarım Mühendisi – ECS Elektrik A.Ş. (Temmuz 2017 – Şubat 2018)
- Bölüm İkincisi – Bursa Teknik Üniversitesi Mekatronik Mühendisliği (2017)
- Stajyer - Robotistan Elektronik Ticaret A.Ş. (Temmuz 2016 – Eylül 2016)
- Stajyer - Baykal Makine San. ve Tic. A.Ş. (Ağustos 2015 – Eylül 2015)

DİĞER ESERLER, SUNUMLAR VE PATENTLER:

- **Özcan, S., Çoban, M. & Gelen, G**. (2017). An Easy to Use Joystick Controller Design for Industrial Robots, 5th International Conference on Advanced Technology and Science (ICAT'17), (ss. 795-798). İstanbul : Bahçeşehir Üniversitesi, Mayıs 09-12.
- **Gelen, G., Özcan, S.** (2019). İnsan-robot etkileşiminin biyomimetik yaklaşımla sağlanması, Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi, 25/2 (2019), s. 188-198.