OTONOM İNSANSIZ HAVA ARAÇLARI İÇİN NESNE TAKİBİ

Mugtaba Abdalrazig Mohamed ALHASSAN

T.C.
BURSA ULUDAĞ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

OTONOM İNSANSIZ HAVA ARAÇLARI İÇİN NESNE TAKİBİ

Mugtaba Abdalrazig Mohamed ALHASSAN
0000-0003-1611-7046

Prof. Dr. Ersen YILMAZ
(Danışman)

DOKTORA TEZİ
ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

BURSA – 2025
Her Hakkı Saklıdır

ÖZET

Doktora Tezi

OTONOM İNSANSIZ HAVA ARAÇLARI İÇİN NESNE TAKİBİ

Mugtaba Abdalrazig Mohamed ALHASSAN

Bursa Uludağ Üniversitesi
Fen Bilimleri Enstitüsü
Elektronik Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Ersen YILMAZ

Nesne algılama teknolojilerinde yaşanan hızlı gelişmeler, İnsansız Hava Araçlarının (İHA) operasyonel kapasitelerini önemli ölçüde dönüştürmüş ve bu sistemlerin gözetim, afet yönetimi ile çevresel izleme gibi farklı alanlarda etkin bir biçimde kullanılmasına olanak sağlamıştır. Ancak, İHA operasyonlarının doğasında bulunan değişken irtifalar, yüksek hızlı hareket ve öngörülemeyen çevresel koşullar gibi zorluklar, gerçek dünya koşullarında güvenilir performans elde edilebilmesi, sağlam ve yüksek verimliliğe sahip takip sistemlerinin geliştirilmesini zorunlu kılmaktadır.

Bu tez, otonom İHA'lar için özel olarak uyarlanmış nesne takip sistemlerinin tasarım ve uygulama süreçlerindeki iyileştirmeleri incelemekte olup, bu doğrultuda en güncel bilgisayarlı görü tekniklerinden, özellikle YOLO algoritmasından yararlanmaktadır. Gerçek zamanlı hedef tespiti ve takibiyle ilgili zorlukları ele almak üzere geliştirilen sistemlerin doğruluk, sağlamlık ve hesaplama verimliliği açısından performansı; örtüşmeler, değişken aydınlatma koşulları ve küçük hedef boyutları gibi farklı koşullar altında kapsamlı biçimde incelenmektedir. Önerilen yöntemlerin geçerliliğini sağlamak amacıyla hem hava veri setleri hem de gerçek dünya senaryoları kullanılmaktadır.

Bunun yanı sıra, görsel tabanlı sistemlerden sağlanan verilerin ataletsel sensörler ve Küresel Konumlama Sistemi (GPS) çıktıları ile bütünleştirilmesi aracılığıyla takip güvenilirliğinin artırılmasına yönelik sensör füzyon yöntemleri araştırılmıştır. Bu entegrasyon, hızlı İHA manevraları sırasında veya yalnızca görsel verilerin yetersiz olduğu durumlarda hedef takibini sürdürme yeteneğini geliştirmektedir. Önerilen takip sistemi, arama-kurtarma operasyonları, yaban hayatının izlenmesi ve altyapı denetimleri gibi kritik İHA uygulamalarında kullanılarak hem pratik yararları hem de çok yönlü kullanım potansiyeli gösterilmiştir.
Anahtar Kelimeler: Görüntü işleme, insansız hava araçları, nesne algılama, yolo

2025, ix + 66 sayfa.

ABSTRACT

PhD Thesis

OBJECT TRACKING FOR AUTONOMOUS UNMANNED AERIAL VEHICLES

Mugtaba Abdalrazig Mohamed ALHASSAN

Bursa Uludağ University
Graduate School of Natural and Applied Sciences
Department of Electronics Engineering

Supervisor: Prof. Dr. Ersen YILMAZ

The rapid evolution of object detection has fundamentally transformed the operational capabilities of Unmanned Aerial Vehicles (UAVs), thereby facilitating their integration into a diverse array of applications such as surveillance, disaster management, and environmental monitoring. However, the inherent challenges of UAV operations—including variable altitudes, high-speed movement, and unpredictable environmental conditions—necessitate the development of robust and efficient tracking systems to ensure reliable performance in real-world scenarios.

This thesis examines enhancements in the design and implementation of object tracking systems specifically tailored for autonomous UAVs. State-of-the-art computer vision techniques, notably the YOLO algorithm, are employed to address the challenges associated with real-time target detection and tracking. The performance of these systems is evaluated in terms of accuracy, robustness, and computational efficiency under various conditions, including occlusions, fluctuating illumination levels, and small target sizes. To validate the proposed methods, both aerial datasets and real-world scenarios are utilized.

Additionally, sensor fusion techniques are explored to enhance tracking reliability by combining data from vision-based systems with inertial and Global Positioning System (GPS) sensors. This integration improves the system's ability to maintain target tracking during rapid UAV maneuvers or when visual data alone is insufficient. The proposed tracking system is demonstrated in key UAV applications, such as search and rescue missions, wildlife monitoring, and infrastructure inspection, showcasing its practical benefits and versatility.

Key words: Image processing, object detection, unmanned aerial vehicles, yolo

2025, ix + 66 pages.

TEŞEKKÜR

Bu süreçte bana rehberlik eden ve her zaman yanımda olan kıymetli danışmanım **Prof. Dr. Ersen YILMAZ**'a en içten teşekkürlerimi sunarım. Ayrıca, tüm hocalarıma, özellikle **Prof. Dr. Güneş YILMAZ, Prof. Dr. Cemal HANİLÇİ** ve **Doç. Dr. Esin Karpat**'a, verdikleri destek ve değerli katkıları için samimi teşekkürlerimi iletmek isterim.

Bana inanan ve her daim yanımda olan aileme sonsuz minnettarım.

Bununla birlikte, bu tezin hazırlanmasında gerekli deneyleri gerçekleştirme ve veri toplama fırsatını sunan **MR Havacılık Teknoloji** firmasına da şükranlarımı sunarım.

Mugtaba Abdalrazig Mohamed ALHASSAN

21/07/2025

İÇİNDEKİLER

Sayfa

ÖZET i
ABSTRACT ii
TEŞEKKÜR iii
SİMGELER ve KISALTMALAR DİZİNİ v
ŞEKİLLER DİZİNİ vii
ÇİZELGELER DİZİNİ ix
1. GİRİŞ 1
2. KAYNAK ARAŞTIRMASI 4
3. MATERYAL ve YÖNTEM 8
3.1. İHA Uygulamalarında YOLO Algoritmalarının Entegrasyonu 8
3.2. YOLO Algoritmasının Diğer Nesne Tespit Algoritmalarına Kıyasla Üstünlükleri 8
3.2.1. Hız ve Gerçek Zamanlı İşlem Yeteneği 8
3.2.2. YOLO Mimari Verimlilik 9
3.2.3. Doğruluk ve Hız Dengesi 10
3.3. Elde edilen YOLOv4 ve YOLOv5 puan sonuçları 10
3.4. YOLO versiyonlarını karşılaştırma: 16
3.5. Performans ölçümü 18
3.6. Metodoloji için Anahtar Değerlendirmeler 20
3.7. Kullanılan Cihazlar ve Ekipmanlar 22
3.7.1. İHA platformu 22
3.7.2. Uçuş Kontrol Sistemi 25
3.7.3. Haberleşme Sistemi 28
3.7.4. Robot İşletim Sistemi (ROS) 29
3.7.5. MAVLink 34
3.7.6. Gömülü Yapay Zekâ İşlemci Cihazı 38
3.7.7. Kamera ve Sabitleme Sistemi (Gimbal) 40
3.8. Kalman Filtresinin Uygulanması 44
3.8.1. Gimbal Takip Algoritması 45
3.8.2. Hedef Konum Hesabı 47
4. BULGULAR ve TARTIŞMA 53
4.1. Tespit Doğruluğu 56
4.2. Hız ve Verimlilik 56
4.3. Güven Değerleri 56
4.4 İHA Uygulamaları İçin Uygunluk 57
5. SONUÇ 61
KAYNAKLAR 64
ÖZGEÇMİŞ 66

SİMGELER ve KISALTMALAR DİZİNİ

Simgeler	**Açıklama**
$\dot{\theta}$	Camera Tilt
$\dot{\psi}$	Camera Pan
θ_{des}	Desired Tilt
ψ_{des}	Desired Pan
IoU	Intersection over Union
mAP	Mean Average Precision

Kısaltmalar	**Açıklama**
AB	Avrupa Birliği
ABD	Amerika Birleşik Devletleri
CCW	Counter Clock Wise
CNN	Convolutional Neural Network
CRC	Cyclic Redundancy Check
CSPDarknet	Cross Stage Partial Darknet
CVDF	Common Visual Data Foundation
CW	Clock Wise
DNN	Deep Neural Network
EO	Electro-Optical Sensor
FC	Flight Controller
FN	False Negative
FNR	False Negative Rate
FP	False Positive
FPN	Feature Pyramid Network
FPR	False Positive Rate
FPS	Frame Per Second
GAN	Generative Adversarial Network
GBP	Grid-Based Prediction
GNSS	Global Navigation Satellite System
GPU	Graphics processing unit
GPS	Global Positioning System
IMU	Inertial Measure Unit
KG	Kilogram
LiPo	Lityum Polimer
mAh	Milliampere-hour
MS COCO	Microsoft Common Objects in Context
PANet	Path Aggregation Network
PID	Proportional İntegral Derivative
PWM	Pulse-width modulation
RC	Radio Controller
R-CNN	Region Based Convolutional Neural Networks
ROS	Robot Operating System
RTH	Return To Home

SAR	Search and Rescue
S.BUS	Serial BUS
SFP	Single Forward Pass
SSD	Single Shot MultiBox Detector
SPP	Spatial Pyramid Pooling
SPPF	Spatial Pyramid Pooling – Fast
TCP	Transmission Control Protocol
TOPS	Tera Operations Per Second
TP	True Positive
TTL	Transistor-Transistor-Logic
UAV	Unmanned Aerial Vehicles
UDP	User Datagram Protocol
XML	Extensible Markup Language
YOLO	You Only Look Once

ŞEKİLLER DİZİNİ

Sayfa

Şekil 3.1. Farklı çözünürlüklere sahip giriş görüntüleri kullanılarak Daha Hızlı R-CNN, R-FCN, SSD ve YOLO modellerini uygulayan saniyede işlenen karelerin (FPS) karşılaştırılması. 9
Şekil 3.2. MS COCO sunucusu kullanılarak elde edilen YOLOv4 ve YOLOv5 puan sonuçları (Sonuçlar COCO sunucularına kaydedilir). 10
Şekil 3.3. YOLOv4 için genel performans sonuçları. 10
Şekil 3.4. YOLOv5_v5'e karşı YOLOv5_v6'ya yapılan geliştirmenin sonuçları (Ultralytics. 2024) 11
Şekil 3.5. YOLOv4'ün hızı ve doğruluğu (Alexey ve ark. 2020) 13
Şekil 3.6. Önerilen yaklaşımın blok şeması 15
Şekil 3.7. (a) ikonik nesne görüntüleri, (b) ikonik sahne görüntüleri ve (c) ikonik olmayan görüntüler örneği (Lin ve ark. 2015). 16
Şekil 3.8. Nesne algılama ve kutular örtüşse de ayrıntılar kaybolabilir; kişi örneği segmentasyonun zorluğunu, MS COCO'daki ortalama örtüşme ise problemin boyutunu göstermektedir (Lin ve ark., 2015) 16
Şekil 3.9. Kesişim/Birleşim Oranı (IoU) metriği 19
Şekil 3.10. Elektronik yerleşimi 22
Şekil 3.11. IHA platformu 23
Şekil 3.13. Motor Numaraları ve Dönüş Yönleri 25
Şekil 3.14. Pixhawk CUAV V6X (Pixhawk. 2025) 26
Şekil 3.15. CUAV V6X Motor Bağlantı Pinleri (Pixhawk. 2025) 26
Şekil 3.16. Neo3 Pro GPS alıcısı (Pixhawk. 2025) 28
Şekil 3.17. Skydroid H16 Pro Uzaktan Kumanda (Skydroid. 2025) 28
Şekil 3.18. Telemetri ve RF Kontrol sinyallerini alan ve gönderen Transmitter/receiver donanımı 29
Şekil 3.19. ROS Bağlantı Şeması (ROS. 2024) 30
Şekil 3.20. Mavros Kütüphanesi tarafından yayın yapılan konu başlıkları 31
Şekil 3.21. ROS eklenti diagramları 33
Şekil 3.22. Gazebo Gimbal Model Eklentisi 33
Şekil 3.23. MAVLink mesaj yapısı (MAVLink. 2024) 36
Şekil 3.24. Jetson Modelleri Teknik Özellikler (NVIDIA Jetson. 2025) 39
Şekil 3.25. Jetson Orin NX 16 GB modelin yerleşimi 40
Şekil 3.26. Kamera Görüş Alanı ve Takip Nesnesi 42
Şekil 3.27.Viewpro A40 Pro (Viewpro. 2024) 43
Şekil 3.29. Gimbal PI Kontrolcü 47
Şekil 3.30. Bağıl Koordinatlar 47
Şekil 3.31. Başakşehir bölgesinde drone ile kaydedilen bir videodan alınan görüntünün YOLOV4'e beslendi. 49
Şekil 3.32. Şekil 3.30'teki görüntünün YOLOv5 ile işlenmesi. 50
Şekil 3.33. Başakşehir yakınlarında drone'dan çektiğim bir videodan bir görüntü YOLOV4'e beslendi 50
Şekil 3.34. Şekil 3.33'teki aynı görüntü ancak YOLOV5'e aktarıldı 51
Şekil 4.1. YOLOv4 ve YOLOv5 için kişi, otobüs ve araba kategorilerinde Ortalama Kesinlik (AP) dağılımını gösteren kutu grafiği. 57

Şekil 4.2. YOLOv4 ve YOLOv5 için kişi, otobüs ve araba kategorilerinde Ortalama Kesinlik (AR) dağılımını gösteren kutu grafiği.....58
Şekil 4.3. YOLO ve Kalman Filtresi kullanılarak hareketli bir aracın takibi 58
Şekil 4.4. Takip edilen nesne kaybolduğunda bile Kalman Filtresi konumunu tahmin edebilmektedir. 59
Şekil 4.5. Nesne yeniden göründüğünde takibin devam etmesi.....59
Şekil 4.6. YOLO'nun her nesneye kimlik (ID) ataması ve Kalman Filtresi ile birlikte çalışarak benzer nesneler arasında aynı nesnenin kaybolmasını önlemesi..60

ÇİZELGELER DİZİNİ

Sayfa

Çizelge 3.1. ViewPro A40Pro Kamera Teknik Özellikleri. 44
Çizelge 4.1. MS COCO değerlendirme sunucusunda, YOLOv4 ve YOLOv5 algoritmaları için ortalama kesinlik (Average Precision) değerleri, "araba" kategorisi temel alınarak hesaplanmıştır. 53
Çizelge 4.2. MS COCO değerlendirme sunucusunda, YOLOv4 ve YOLOv5 algoritmaları için ortalama kesinlik (Average Precision) değerleri, "otobüs" kategorisi temel alınarak hesaplanmıştır. 53
Çizelge 4.3. MS COCO değerlendirme sunucusunda, YOLOv4 ve YOLOv5 algoritmaları için ortalama kesinlik (Average Precision) değerleri, "kişi" kategorisi temel alınarak hesaplanmıştır. 54
Çizelge 4.4. YOLOv4 ve YOLOv5 için kişi, otobüs ve araba nesne kategorilerinde Ortalama Hassasiyet (AP) değerlerinin ortalama ± standart sapma değerleri. 57
Çizelge 4.5. YOLOv4 ve YOLOv5 için kişi, otobüs ve araba nesne kategorilerinde Ortalama Hassasiyet (AR) değerlerinin ortalama ± standart sapma değerleri. 57

1. GİRİŞ

Nesne algılama, bilgisayarlı görü alanında önemli bir araştırma alanı olarak öne çıkmakta ve makinelerin görüntü veya video içindeki nesneleri tanımlayıp konumlandırmasını sağlamaktadır. Bu yetenek, otonom araçlar, gözetim sistemleri ve robotik gibi birçok alanda kritik bir rol oynamaktadır. Nesne algılama algoritmalarının gelişimi, derin öğrenme alanındaki ilerlemelerden, özellikle Evrişimli Sinir Ağlarının (Convolutional Neural Networks, CNNs) geliştirilmesinden büyük ölçüde etkilenmektedir. Geleneksel, elle tasarlanmış özelliklere dayanan yöntemlerden, doğrudan veriden hiyerarşik özellik temsillerini öğrenebilen karmaşık modellere doğru bir dönüşüm gerçekleşmektedir.

Nesne algılama algoritmaları genel olarak iki ana kategoriye ayrılmaktadır: iki aşamalı ve tek aşamalı algılayıcılar. İki aşamalı algoritmalar, örneğin R-CNN ve türevleri (Fast R-CNN, Faster R-CNN), önce bölge önerileri oluşturmakta ve ardından bu bölgeleri sınıflandırmaktadır. Bu yöntem, yüksek doğruluk sunmakla birlikte genellikle hızdan ödün verilmesini gerektirmektedir. Buna karşılık, YOLO (You Only Look Once) ve SSD (Single Shot MultiBox Detector) gibi tek aşamalı algoritmalar, nesne sınıflarını ve sınırlayıcı kutuları tek bir geçişte tahmin ederek süreci basitleştirmektedir. Bu algoritmalar, gerçek zamanlı performansı önceliklendirirken makul bir doğruluk seviyesini korumaktadır.

Son yıllarda gerçekleştirilen gelişmeler, dikkat mekanizmalarından yararlanarak nesneler arası ilişkileri daha etkin biçimde modelleyen transformatör tabanlı yaklaşımların nesne algılama süreçlerine entegre edilmesine olanak sağlamıştır. Bu yöntemler, algılama doğruluğunu ve işlem verimliliğini artırmada önemli bir potansiyel sunmakta, aynı zamanda mevcut yöntemlerin yeteneklerini genişletmektedir. Alanın sürekli ilerlemesiyle birlikte, uygun algoritma seçimi çoğunlukla uygulama gereksinimlerine bağlı olarak belirlenmekte; bu süreçte doğruluk ile hesaplama verimliliği arasındaki denge temel bir kriter olarak değerlendirilmektedir.

YOLO algoritmasının evrimi, nesne algılama teknolojisinde en çığır açıcı ilerlemelerden birini temsil etmektedir. İlk kez Haziran 2016'da Joseph Redmon ve ekibi tarafından

tanıtılan ve GitHub üzerinde paylaşılan YOLO, nesne algılama paradigmasını yeniden tanımlayarak bu süreci bir regresyon problemi olarak ele almaktadır. Bu yenilikçi yaklaşım, tek bir Evrişimli Sinir Ağı (CNN) kullanarak nesnelerin konumlarını ve ilgili olasılıklarını aynı anda tahmin etmeyi mümkün kılmaktadır. Sonuç olarak, YOLO, benzersiz bir hız ve doğruluk seviyesi yakalamakta ve genelleme yeteneklerini önemli ölçüde geliştirmektedir.

Orijinal YOLO algoritması, ardından gelen geliştirmelere temel oluşturarak önemli bir başlangıç noktası sağlamaktadır. Bu gelişim süreci, Aralık 2016'da YOLOv2'nin tanıtılmasıyla başlamıştır. YOLOv2, tüm evrişimsel katmanlarda batch normalization yöntemini uygulayarak aşırı öğrenme riskini azaltmış ve eğitim sürecinin verimliliğini kayda değer ölçüde artırmıştır. Bunun yanı sıra, söz konusu sürüm modelin kararlılığını ve genel performansını da geliştirmiştir. YOLOv2'deki bu yenilikler, Nisan 2018'de sunulan YOLOv3 için temel oluşturmuştur. YOLOv3, özellikle çok ölçekli nesne algılama yeteneğiyle dikkat çekmiş; bu özellik sayesinde model, üç farklı ölçekte nesne tahmini yaparak farklı boyutlardaki nesneler üzerinde daha yüksek bir doğruluk elde etmiştir. Ayrıca, daha verimli bir omurga mimarisi olan Darknet-53'ün entegrasyonu, hem algılama doğruluğunu hem de işlem hızını belirgin biçimde artırmıştır.

Nisan 2020'de yayımlanan YOLOv4, nesne algılama alanında önemli bir dönüm noktası olarak değerlendirilmektedir. Bu sürüm, kaynak kullanımını stratejik biçimde optimize ederek, uç cihazlar da dahil olmak üzere geniş bir yelpazedeki bilgi işlem platformlarına uyum sağlamayı kolaylaştırmıştır. YOLOv4'te kullanılan yenilikçi CSPDarknet53 omurga mimarisi, hem algılama doğruluğunu hem de çıkarım hızını artırmış ve bu sayede modeli dönemin en gelişmiş nesne algılama yaklaşımlarından biri haline getirmiştir.

Bu temelin üzerine inşa edilen YOLOv5, Ultralytics tarafından geliştirilmiş ve CSP tabanlı bir omurga mimarisiyle tanıtılmıştır. Bu sürüm, modelin hızlı çıkarım hızını korurken doğruluğunu daha da artırmaktadır. CSP mimarisi, YOLOv5'in mobil cihazlar ve uç sistemlerden NVIDIA gibi yüksek performanslı GPU'lara kadar farklı platformlarla uyumlu olmasını sağlamaktadır. Ayrıca, YOLOv5 önceki sürümlerin kullandığı anchor

tabanlı tahminleri sürdürmekte, ancak buna ek olarak yenilikçi bir anchor-free yaklaşım benimseyerek daha küçük nesneleri algılama yeteneğini artırmaktadır.

Faster R-CNN ve EfficientDet gibi diğer nesne algılama modelleri, İHA uygulamalarında umut vadeden alternatifler arasında yer almaktadır. İki aşamalı bir dedektör olan Faster R-CNN, karmaşık ortamlarda yüksek algılama doğruluğu ve sağlam performansıyla tanınmaktadır. Ancak, önemli ölçüde yüksek hesaplama gereksinimleri, kaynakları sınırlı İHA platformlarında uygulanabilirliğini sınırlamaktadır. Öte yandan, EfficientDet modeli, bileşik ölçeklendirme stratejisi ve gelişmiş mimarisi sayesinde hız ile doğruluk arasında dengeli bir performans sunmakta; bu yönüyle İHA tabanlı görevlerde güçlü bir alternatif olarak öne çıkmaktadır. Bu güçlü yönlere rağmen, her iki model de, YOLO'nun optimize edilmiş hafif yapısı ve gerçek zamanlı uygulamalardaki verimliliği karşısında yüksek hesaplama gereksinimleri nedeniyle zorluklarla karşılaşmaktadır. YOLO, özellikle İHA platformları üzerinde gerçek zamanlı çalışmayı sağlamak için tasarlanmış hafif bir yapı sunmaktadır.

Microsoft Common Objects in Context (MS COCO) veri seti, bilgisayarlı görü alanında temel bir kaynak olarak öne çıkmaktadır ve kapsamlı, çeşitli etiketli görüntü koleksiyonuyla tanınmaktadır. İlk kez 2014 yılında tanıtılan bu veri seti, nesne algılama, segmentasyon ve görüntü açıklama görevlerinde ilerlemeyi kolaylaştırmak amacıyla geliştirilmiştir. 330.000'den fazla görüntü ve çok çeşitli günlük sahneler içeren MS COCO, makine öğrenimi modellerinin eğitimi ve değerlendirilmesi için zengin bir kaynak sunmaktadır.

MS COCO veri setinin önemi yalnızca büyüklüğü ve çeşitliliği ile sınırlı değildir; aynı zamanda en gelişmiş nesne algılama algoritmalarının performansını değerlendirmek için bir kıstas olarak hizmet etmektedir. Araştırmacılar ve uygulayıcılar, modellerini standartlaştırılmış bir dizi zorluk karşısında test etmek ve farklı yöntemler arasındaki karşılaştırmaları kolaylaştırmak için COCO'yu kullanmaktadır. Yapılandırılmış formatı, çeşitli derin öğrenme çerçeveleriyle sorunsuz entegrasyona olanak tanımakta ve bu da onu sağlam bilgisayarlı görü uygulamaları geliştirmek için tercih edilen bir seçenek haline getirmektedir.

2. KAYNAK ARAŞTIRMASI

Derin öğrenmenin yükselişi, nesne algılama yeteneklerini önemli ölçüde ilerletmekte ve örtüşme (occlusion) ile ölçek değişimleri gibi zorlukların daha etkili bir şekilde ele alınmasını sağlamaktadır. Evrişimli Sinir Ağları CNNs gibi teknikler, sağlam nesne algılama sistemlerinin geliştirilmesinde temel bir rol oynamaktadır. Bu dönüşüm, gözetim sistemleri ve otonom araçlar gibi alanlarda geniş çapta uygulama bulmaktadır.

Geleneksel makine öğrenimi yöntemleri ile derin öğrenme tekniklerinin karşılaştırmalı bir analizi, geleneksel yöntemlerin büyük ölçüde elle tasarlanmış özelliklere dayanırken, derin öğrenme yaklaşımlarının özellik çıkarma işlemini otomatikleştirdiğini ve bu durumun nesne algılama görevlerinde daha yüksek doğruluk ve verimlilik sağladığını göstermektedir (Bochkovskiy ve ark. 2020).

YOLO gerçek zamanlı nesne algılama çerçevesi olarak ön plana çıkmaktadır. Bu mimari, görüntüler içindeki nesnelerin aynı anda tespiti ve sınıflandırılmasını mümkün kılmakta ve etkileyici hızlar ile doğruluk seviyelerine ulaşarak otonom sürüş ve video gözetimi gibi uygulamalar için uygun hale gelmektedir (Jiang ve ark. 2022).

MS COCO veri seti, nesne algılama algoritmalarını değerlendirmek için standart bir kıstas haline gelmiştir. Çeşitli görüntü koleksiyonu ve ayrıntılı açıklamaları sayesinde, araştırmacıların modelleri etkili bir şekilde eğitmelerine olanak tanımakta ve yeni algoritmaların test edilmesi için kapsamlı bir platform sunarak alanın gelişimine katkıda bulunmaktadır.

MS COCO gibi yerleşik veri setlerine dayalı sürekli kıyaslamalar, yeni algoritmaların performansını değerlendirmek için hayati önem taşımaktadır. Bu uygulama, farklı yöntemler arasındaki karşılaştırmayı kolaylaştırmakla kalmayıp iyileştirme gerektiren alanları vurgulayarak yenilikleri teşvik etmektedir.

MS COCO veri seti, nesne algılama modellerinin, özellikle de çeşitli YOLO sürümlerinin eğitimi ve değerlendirilmesinde önemli bir rol oynamaktadır. 80'den fazla nesne

kategorisini ve ayrıntılı segmentasyon maskelerini kapsayan geniş açıklamaları ile MS COCO, model performansını değerlendirmek için sağlam bir kıstas sunmaktadır. Veri setinin geniş görüntü yelpazesi, YOLO modellerinin farklı ortamlar genelinde daha iyi genelleme yapmasını sağlamaktadır. Son literatür, büyük ölçekli veri setlerinin kullanılmasının, nesne algılama algoritmalarının sağlamlığını ve doğruluğunu artırmadaki önemini vurgulamaktadır (Lin ve ark. 2015).

Nesne algılama, güvenlik ve kamu güvenliği açısından hareketli nesnelerin video akışlarında tespit edilmesinin gerekli olduğu gözetim sistemlerinde kritik bir rol oynamaktadır. Arka plan çıkarımı gibi teknikler, karmaşık ortamlarda nesne takibinin doğruluğunu artırmak için yaygın olarak kullanılmaktadır (Abba ve ark. 2024).

Sağlık sektöründe nesne algılama algoritmaları, hastalıkların teşhisinde tıbbi görüntülerin analizinde uygulanmaktadır. Örneğin, derin öğrenme modelleri, röntgen veya MR görüntülerindeki anormallikleri tespit ederek sağlık profesyonellerinin zamanında kararlar almasına yardımcı olmaktadır (Albuquerque ve ark. 2025).

Tarımda, nesne algılama teknolojileri mahsul sağlığını izlemek ve zararlıları tespit etmek için giderek daha fazla kullanılmaktadır. Nesne algılama algoritmaları ile donatılmış İHA'lar (İnsansız Hava Araçları), geniş alanları verimli bir şekilde tarayarak çiftçilere tarlaları hakkında değerli bilgiler sağlamaktadır (Pavate ve ark. 2025).

Gelişmelere rağmen, gerçek zamanlı nesne algılama, değişen çevre koşulları altında küçük veya örtük nesneleri tespit etmek gibi zorluklarla karşı karşıya kalmaktadır. Süregelen araştırmalar, bu senaryolarda performansı artırmak için algoritmaları geliştirmeyi hedeflemektedir (YINGJIE ve ark. 2020).

Ayrıca, özellikle küçük nesnelerin tespiti ve olumsuz aydınlatma koşullarında performansın optimize edilmesi gibi belirli zorlukların üstesinden gelmek için YOLO çerçevesine yönelik çeşitli değişiklikler araştırılmıştır.

Nesne algılama araştırmalarında ortaya çıkan eğilimler arasında zayıf denetimli öğrenme, çoklu alan algılama ve model eğitimini ve performansını artırmak için Üretici Adversarial Ağların (G.A.N'ler) entegrasyonu yer almaktadır (Lim ve ark. 2024).

YOLO algoritması, geliştirilmesinden bu yana önemli bir evrim geçirmiştir ve YOLOv4 ile YOLOv7 gibi sürümleri, algılama doğruluğu ve hızında önemli iyileştirmeler göstermektedir. Araştırmalar, bu sürümlerin çeşitli çevresel koşullarda araç algılama doğruluklarında %93'ı aşan sonuçlara ulaştığını ve bu durumun, Akıllı Ulaşım Sistemleri (Intelligent Transport System, ITS) etkinliklerini ortaya koyduğunu göstermektedir (Omar ve ark. 2023).

Mimari geliştirmeler, anchor kutularının entegrasyonu ve gelişmiş özellik çıkarma teknikleri de dahil olmak üzere, YOLO'nun gerçek zamanlı algılama yaparken yüksek hassasiyetini korumasına katkıda bulunmaktadır (Diwan ve ark. 2023).

YOLO algoritmalarının İHA çerçevelerinde kullanımı, hava gözetimi, tarım ve afet yönetimi gibi alanlarda verimli gerçek zamanlı nesne algılama ihtiyacının artması nedeniyle ilgi görmektedir. Çalışmalar, YOLO tabanlı sistemlerin tarım alanlarının izlenmesi, trafik senaryolarında araçların tespiti ve arama-kurtarma operasyonlarında nesnelerin belirlenmesi gibi uygulamalardaki etkinliğini vurgulamaktadır (David ve ark. 2020).

Elma tespiti görevi için YOLOv5 ve YOLOv3'ün performanslarının karşılaştırıldığı kapsamlı bir analizde, YOLOv5'in çeşitli metriklerde üstün performans sergilediği ortaya konmuştur. Araştırmada, YOLOv5'in %96,5 doğruluk, %97,2 geri çağırma ve %96,9 F1 skoruna ulaştığı, buna karşılık YOLOv3'ün bu metriklerin tümünde daha düşük değerlere sahip olduğu görülmüştür. Ayrıca, YOLOv5'in hem Yanlış Negatif Oranı (FNR) hem de Yanlış Pozitif Oranı (FPR) açısından önemli azalmalar gösterdiği tespit edilmiştir. Bu bulgular, YOLOv5 tarafından temsil edilen derin evrişimsel ağlardaki ilerlemeleri ve hassasiyet ile verimliliği artırma potansiyelini vurgulamaktadır (Tatiana ve ark. 2021).

Acil durum senaryolarında İHA'lar için uygun iniş noktalarının belirlenmesi amacıyla üç farklı YOLO nesne algılama algoritması sürümünün performansını değerlendirmiştir. YOLOv3, Darknet53 omurgasıyla hızı ile öne çıkmakta, ancak görece düşük doğruluk sergilemektedir. Buna karşılık, CSPDarknet53 omurgasını içeren YOLOv4, YOLOv3'e kıyasla hem doğruluk hem de algılama hızında önemli iyileştirmeler sağlamaktadır. Bu gelişmeler, modelin özellikleri daha etkili bir şekilde işlemesini ve çıkarmasını mümkün kılan Spatial Pyramid Pooling (SPP) ve Path Aggregation Network (PANet) gibi gelişmiş özelliklerin entegrasyonuna bağlanmaktadır (Nepal ve ark. 2022).

3. MATERYAL ve YÖNTEM

3.1. İHA Uygulamalarında YOLO Algoritmalarının Entegrasyonu

Metodoloji, hız, doğruluk ve operasyonel verimlilik arasındaki dengeyi gözetirken, ortaya çıkan zorlukları aşmak amacıyla en uygun donanım ve algoritma yapılandırmalarının belirlenmesine odaklanmaktadır.

Bu yapılandırılmış yaklaşım, İHA sisteminin güçlü, yüksek verimli ve gerçek dünya uygulamalarının gereksinimlerini etkin biçimde karşılayabilecek kapasitede tasarlanmasını temin etmektedir.

İHA uygulamalarında yüksek hız ile doğru nesne tespiti arasında bir denge kurmak, çeşitli tavizler ve zorluklarla başa çıkmayı gerektirmektedir. YOLO algoritmalarının İHA sistemlerine entegrasyonu, doğrudan bir süreç olmayıp; donanım kapasitesi, hesaplama gereksinimleri ve çevresel kısıtlamalar gibi faktörlerin titizlikle değerlendirilmesini zorunlu kılmaktadır. Bu faktörler, YOLO'nun İHA tabanlı uygulamalardaki uygulanabilirliği, performansı ve verimliliği üzerinde belirleyici bir etkiye sahiptir.

3.2. YOLO Algoritmasının Diğer Nesne Tespit Algoritmalarına Kıyasla Üstünlükleri

3.2.1. Hız ve Gerçek Zamanlı İşlem Yeteneği

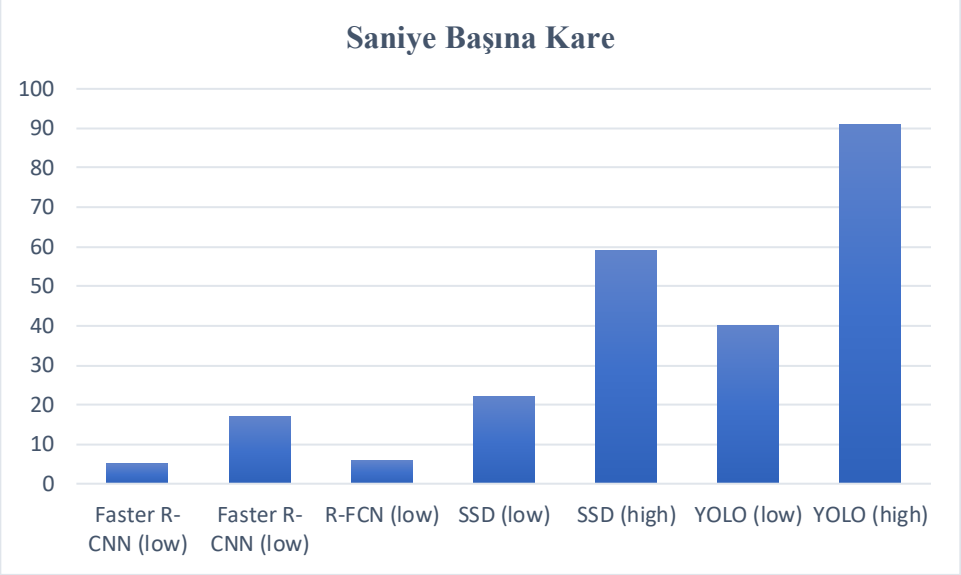
• Tek İleri Yayılım (SFP)

YOLO algoritması, görsel verileri yalnızca tek bir derin sinir ağı geçişiyle işleyerek, hem nesne sınıflarına ait olasılıkları hem de sınırlayıcı kutuların konumlarını aynı anda tahmin edebilmektedir. Şekil 3.1'de gösterildiği gibi, bu birleşik yaklaşım, bölge önerisi ve sınıflandırma işlemlerini birbirinden ayrı adımlar şeklinde yürüten iki aşamalı yöntemlere (örneğin R-CNN ve Faster R-CNN) göre belirgin bir hız üstünlüğü sunmaktadır.

• Yüksek Kare Hızı (FPS)

Safaldin ve arkadaşlarının (2024) çalışmasında ifade edildiği üzere, modern GPU mimarileri üzerinde saniyede 30 ile 79 kare, hatta daha yüksek hızlarda çalışabilen YOLO algoritması, gerçek zamanlı nesne tespiti için güçlü bir çözüm sunmaktadır. Bu yüksek

hesaplama kapasitesi, özellikle zaman duyarlılığı ön planda olan otonom sürüş, video tabanlı gözetim sistemleri ve robotik uygulamalar gibi alanlarda kritik bir avantaj sağlamaktadır.



Şekil 3.1. Farklı çözünürlüklere sahip giriş görüntüleri kullanılarak Daha Hızlı R-CNN, R-FCN, SSD ve YOLO modellerini uygulayan saniyede işlenen karelerin (FPS) karşılaştırılması.

3.2.2. YOLO Mimari Verimlilik

• Izgara Tabanlı Tahmin (GBP)

YOLO algoritması, girdiyi belirli bir ızgara yapısına ayırarak her hücre için birden fazla sınırlayıcı kutu ve bunlara karşılık gelen sınıf olasılıklarını tahmin etmektedir. Bu yaklaşım, görselin bağlamsal bütünlüğünü dikkate alarak yanlış pozitiflerin azaltılmasında önemli bir katkı sağlamaktadır.

• Çoklu Ölçekli Özellik Entegrasyonu

YOLO'nun üçüncü sürümünden itibaren, özellik piramidi ağları (FPN) ile çapa kutuları (anchor boxes) gibi ileri düzey yapılar entegre edilmekte ve bu sayede farklı boyutlardaki nesnelerin tespit kapasitesi optimize edilmektedir. Böylelikle hem küçük hem de büyük ölçekli nesnelerin algılanmasında anlamlı bir doğruluk artışı sağlanmaktadır.

3.2.3. Doğruluk ve Hız Dengesi

YOLO'nun erken sürümleri (v1 ve v2), hız öncelikli mimarileri sebebiyle doğruluk açısından sınırlı kalmaktadır; ancak sonraki gelişmiş sürümlerde (v4, v5 ve v7/v8), mozaik veri artırımı (mosaic augmentation), CSPDarknet tabanlı omurga ağları ve öz-dikkat (self-attention) mekanizmaları gibi ileri derin öğrenme tekniklerinin uygulanmasıyla, çağdaş en başarılı modellerle rekabet edebilecek düzeyde doğruluk elde edilmektedir.

3.3. Elde edilen YOLOv4 ve YOLOv5 puan sonuçları

Submit

Here are your submissions to date (✔ indicates submission on leaderboard):

#	SCORE	FILENAME	SUBMISSION DATE	STATUS	✔	
1	0.743	detections_test-dev2017_yolov4_results.json.zip	02/23/2021 15:34:25	Finished		+
2	0.682	detections_test-dev2017_yolov5_results.zip	03/05/2021 13:29:59	Finished		+
3	0.822	yolov5x_predictions.zip	11/06/2021 18:03:32	Finished		+
4	0.743	detections_test-dev2017_yolov4_results.json.zip	11/07/2021 09:58:52	Finished		+

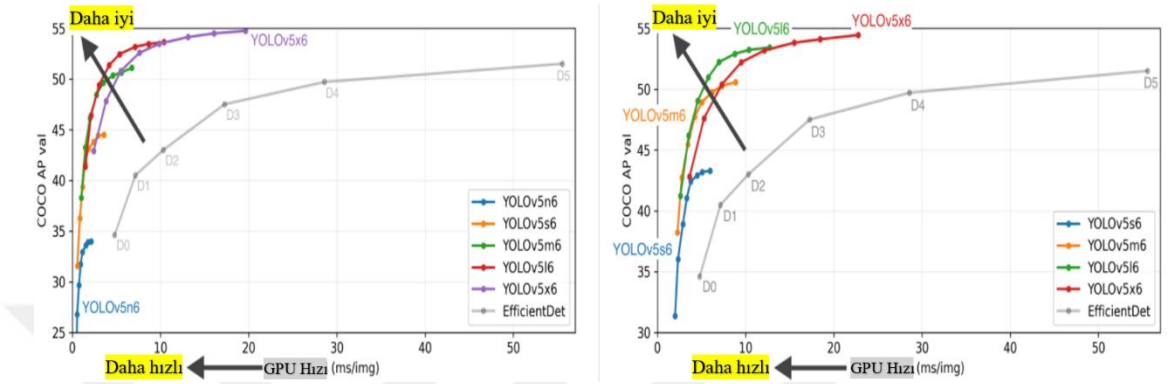
Şekil 3.2. MS COCO sunucusu kullanılarak elde edilen YOLOv4 ve YOLOv5 puan sonuçları (Sonuçlar COCO sunucularına kaydedilir).

```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.441
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.651
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.488
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.271
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.474
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.537
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.341
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.564
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.606
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.417
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.646
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.743
```

Şekil 3.3. YOLOv4 için genel performans sonuçları.

Şekil 3.2 ve Şekil 3.3'te sunulan değerlendirme testi, NVIDIA GPU ve 14 GB RAM'e sahip Google Research Colab sunucuları kullanılarak iki kez gerçekleştirildi, YOLOv5

hala geliştirme aşamasında olduğundan ve topluluktan günlük güncellemeler aldığından test tekrarlanmıştır. Sonuç olarak YOLOv4 skor sonucunun değişmediğini ancak 8 içinde yeni güncellemelerle testi tekrarladıktan sonra MS COCO metriklerine göre YOLOv5 skorunun 0.682'den 0.822'ye çıktığını görünmektedir.



Şekil 3.4. YOLOv5_v5'e karşı YOLOv5_v6'ya yapılan geliştirmenin sonuçları (Ultralytics. 2024)

Şekil 3.4, YOLOv5 üzerinde gerçekleştirilen temel iyileştirmelerin çıktısını ortaya koymaktadır. Bu bağlamda, modelde yapılan başlıca güncellemeler şunlardır:

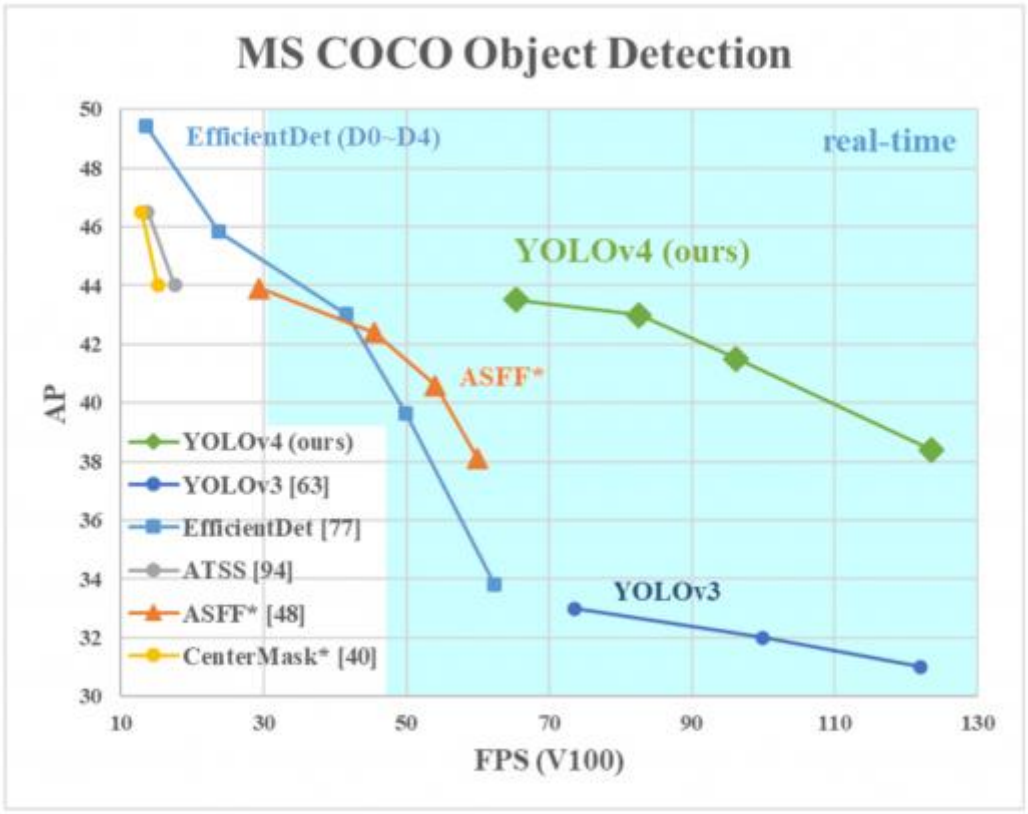
- Roboflow Entegrasyonu: YOLOv5 modellerini doğrudan herhangi bir Roboflow veri kümesinde eğitin.
- YOLOv5n 'Nano' modeller: YOLOv5s (7.5M param) altındaki yeni daha küçük YOLOv5n (1,9M param) modeli, ultra hafif mobil çözümler için ideal olan 2,1 MB INT8 boyutuna dışa aktarır.
- TensorFlow ve Keras Dışa Aktarma: TensorFlow, Keras, TFLite, TF.js model dışa aktarımı artık python export.py kullanılarak tamamen entegre edilmiştir --include save_model pb tflite tfjs.
- OpenCV DNN: YOLOv5 ONNX modelleri artık hem OpenCV DNN hem de ONNX Runtime ile uyumludur.
- Model Mimarisi: Güncellenen omurgalar biraz daha küçük, daha hızlı ve daha doğru.
- Gelişmiş dışa aktarma yeteneği için Focus() öğesinin eşdeğer bir Conv(k=6, s=2, p=2) katmanıyla değiştirilmesi.
- Azaltılmış işlemler için SPP() katmanının yeni SPPF() yedeği.

- P3 omurga katmanında azalma C3(), daha iyi hızlar için 9'dan 6'ya tekrar eder.
- Yeniden sıralama SPPF()'yi omurganın sonuna yerleştirir.
- Son C3() omurga katmanında kısayolun yeniden tanıtılması.
- Arttırılmış karıştırma ve kopyala-yapıştır artırımı ile güncellenmiş hiperparametreler.

YOLO algoritmasının çalışma prensibi, bir görüntünün giriş olarak alınması ve bu görüntünün $S \times S$ boyutlarında bir ızgaraya bölünmesi esasına dayanmaktadır. Her bir ızgara hücresi, m adet sınırlayıcı kutu (bounding box) üretmekle sorumludur. Bu sınırlayıcı kutuların her biri için algoritma, ilgili kutunun ait olduğu nesne sınıfının olasılık değeri ile konumsal ofset bilgilerini (merkez koordinatları, genişlik ve yükseklik) üretir. Daha sonra, belirlenen bir eşik değerinin üzerinde sınıf olasılığına sahip olan sınırlayıcı kutular seçilerek, ilgili nesnenin görüntü içerisindeki konumu tespit edilmektedir.

YOLO algoritması, gerçek zamanlı uygulamalarda kullanılabilecek düzeyde yüksek işlem hızına sahiptir ve literatürdeki diğer nesne algılama yöntemlerine kıyasla oldukça hızlıdır (yaklaşık saniyede 45 kare işleyebilmektedir). Bununla birlikte, algoritmanın önemli bir sınırlaması, görüntü içerisindeki küçük boyutlu nesneleri algılamada zorluk yaşamasıdır. Örneğin, bir kuş sürüsünü tespit etmekte yetersiz kalabilmektedir. Bu durum, YOLO'nun kullandığı uzamsal (spatial) bölütleme yaklaşımının sınırlı çözünürlük sağlamasından kaynaklanmaktadır; zira her bir ızgara hücresi yalnızca belirli sayıda nesne tespit edebilmektedir ve bu durum, özellikle küçük nesnelerin tespitinde performans kaybına neden olmaktadır.

YOLO v4, aynı zamanda Darknet'e dayanmaktadır ve hem hız hem de doğruluk açısından en hızlı ve en doğru dedektörleri geçerek Tesla V100'de 65 FPS'lik gerçek zamanlı hız ile birlikte COCO veri kümesinde yüzde 43,5'lik bir AP değeri elde etmiştir. YOLO v3 ile karşılaştırıldığında, Ortalama Doğruluk (AP) ve Saniyedeki Kare Sayısı (FPS) metriklerinde, sırasıyla %10 ve %12 oranında artış gözlemlenmiştir.



Şekil 3.5. YOLOv4'ün hızı ve doğruluğu (Alexey ve ark. 2020)

YOLOv4 algoritması, son teknoloji BoF (Bag of Freebies) ve BoS (Bag of Specials) yaklaşımlarının katkılarından faydalanmaktadır.

Bag of Freebies (BoF), yalnızca eğitim aşamasında kullanılan ve çıkarım (inference) süresini etkilemeden dedektörün doğruluk performansını artıran yöntemleri kapsamaktadır. Bu teknikler, yalnızca eğitim sürecinin hesaplama maliyetini artırmakta; ancak çıkarım sırasında ek bir yük oluşturmamaktadır.

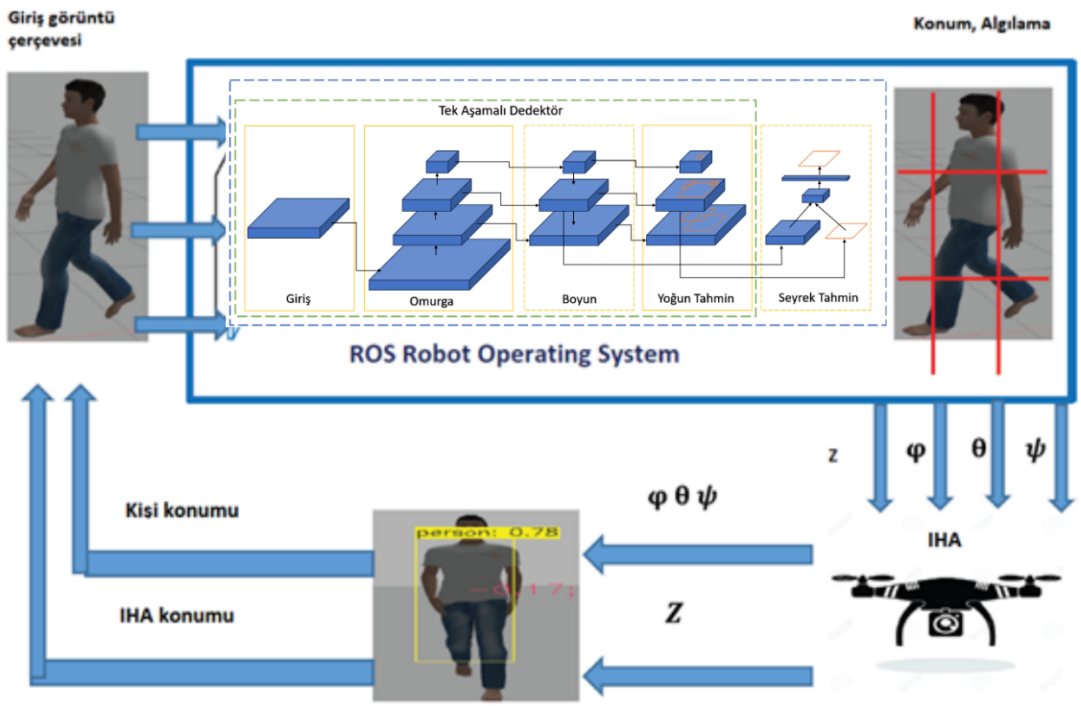
Buna karşılık, Bag of Specials (BoS) ise çıkarım aşamasında sınırlı bir hesaplama maliyeti artışına neden olmakla birlikte, nesne algılama doğruluğunu belirgin ölçüde iyileştiren yapılar içermektedir. Şekil 3.4'te, YOLOv4'ün söz konusu iki yaklaşımı bir araya getirerek hem yüksek verimlilik hem de üstün doğruluk sunan bir nesne tespit modeli oluşturduğu gösterilmektedir.

YOLOv5, önceki sürümlerden farklı olarak orijinal Darknet mimarisi üzerine inşa edilmemiş, bunun yerine PyTorch tabanlı olarak geliştirilmiş bir modeldir. YOLOv4 ile benzer şekilde, YOLOv5 mimarisinde CSP (Cross-Stage Partial) temelli bir omurga (backbone) ve PANet (Path Aggregation Network) tabanlı bir boyun (neck) yapısı bulunmaktadır. Modeldeki temel iyileştirmeler arasında mozaik veri artırımı (mosaic data augmentation) ve otomatik öğrenilen sınırlayıcı kutu çapaları (auto-learning anchor boxes) yer almaktadır. Mozaik veri artırımı, dört farklı görüntünün tek bir kompozit görüntüde birleştirilmesiyle modelin farklı ölçeklerdeki nesnelere karşı daha yüksek genelleme kabiliyeti göstermesini sağlamakta; otomatik çapa öğrenimi ise, sınır kutusu çapalarının eğitim süreci boyunca veri kümesine adaptif biçimde optimize edilmesini mümkün kılarak manuel ayarlama ihtiyacını ortadan kaldırmakta ve doğruluğu artırmaktadır. Tüm bu iyileştirmeler sayesinde YOLOv5, hem eğitim verimliliği hem de gerçek zamanlı nesne tespit performansı açısından yüksek başarı göstermektedir.

Derin öğrenme tabanlı izleyiciler (trackers), genellikle Kesişme Üzerinden Birlik (Intersection over Union – IoU) oranına dayalı olarak tahmin edilen hedef konumu etrafında pozitif ve negatif örnekler oluşturmaktadır. Bu örnekler, hedef nesnenin konumuna olan yakınlıklarına göre sınıflandırılmakta; yüksek IoU değerine sahip örnekler pozitif, düşük IoU değerine sahip olanlar ise negatif olarak etiketlenmektedir. Model, bu örnekler aracılığıyla eğitilmekte olup, hedefin daha doğru ve tutarlı bir şekilde izlenmesini sağlamaktadır.

Yeni çerçevede hedefin konumunu tahmin edebilmek için, genellikle bir önceki çerçevede tahmin edilen hedef konumu etrafında çeşitli aday örnekler (candidate samples) oluşturulmaktadır. Bu adaylar, hedefe ait olma olasılığına göre puanlanmakta ve en yüksek puana sahip örnek, hedefin yeni çerçevedeki konumu olarak belirlenmektedir. Modelin güncellenmesinde kullanılacak örneklerin seçiminde ise genellikle IoU (Intersection over Union) oranına dayalı bir eşikleme stratejisi uygulanmaktadır. Bu bağlamda, IoU değeri %70'in üzerinde olan örnekler pozitif, %30'un altında olan örnekler ise negatif olarak kabul edilmekte ve model bu pozitif-negatif örnekler üzerinden eğitilmeye devam edilmektedir. Bu yaklaşım, modelin takip doğruluğunu ve kararlılığını artırmak amacıyla yaygın olarak kullanılmaktadır.

CNN tabanlı izleyiciler, genellikle hafif (lightweight) derin sinir ağları kullanmaktadır; bunun temel nedeni, büyük ölçekli CNN modellerinin son katmanlarının anlamsal bilgileri (semantic features) daha iyi temsil etmesine rağmen, bu katmanların nesne izleme (object tracking) görevleri için yeterince uygun olmamasıdır. Büyük ağların son katmanları, nesne sınıflandırma ve algılama gibi görevlerde yüksek performans gösterse de, izleme problemlerinde kritik olan uzamsal ayrıntıları (spatial details) yeterince hassas şekilde temsil edemezler. Öte yandan, CNN'lerin erken (ön) katmanları, nesnenin konum ve biçim gibi düşük düzeyli özelliklerini koruma konusunda daha başarılıdır ve bu nedenle izleme görevlerinde daha avantajlıdır. Ayrıca, gerçek zamanlı takip sistemlerinde hesaplama maliyetleri ve işlem hızı kritik öneme sahip olduğundan, bu tür uygulamalarda daha küçük ve verimli mimarilerin tercih edilmesi gerekmektedir.



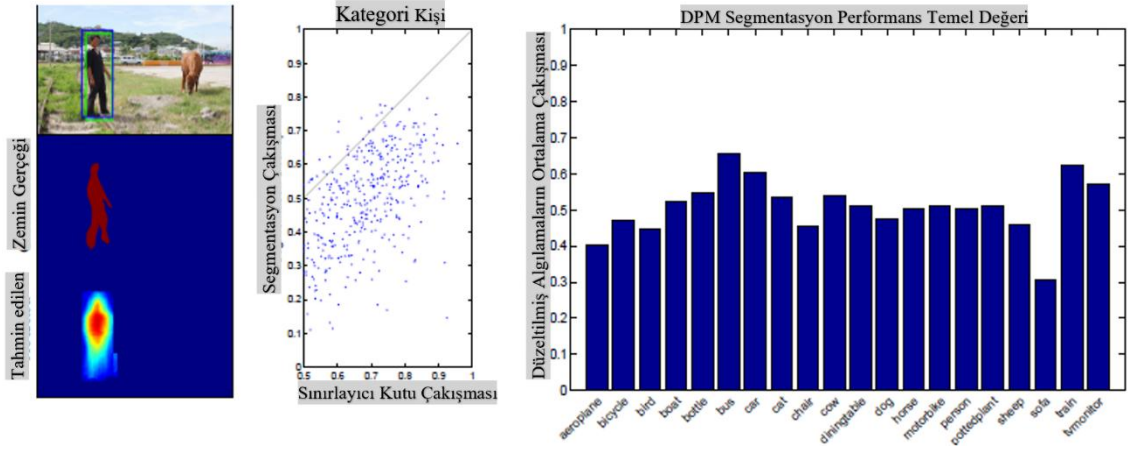
Şekil 3.6. Önerilen yaklaşımın blok şeması

MS COCO veri kümesi, 91 adet yaygın nesne kategorisi içermekte olup, bu kategorilerden 82'si 5.000'den fazla örnekle etiketlenmiş durumdadır. Veri kümesi, toplamda 328.000 görüntü ve bu görüntüler üzerinde yaklaşık 2.500.000 etiketli örnek

(annotated instance) içermektedir. Kapsadığı geniş nesne yelpazesi, çeşitli sahne bağlamları ve yüksek kaliteli açıklamalar sayesinde MS COCO, nesne algılama, sınıflandırma ve segmentasyon gibi bilgisayarla görme alanındaki görevlerin değerlendirilmesi ve karşılaştırılması için yaygın olarak kullanılan, araştırmaya son derece elverişli bir benchmark veri kümesi konumundadır.



Şekil 3.7. (a) ikonik nesne görüntüleri, (b) ikonik sahne görüntüleri ve (c) ikonik olmayan görüntüler örneği (Lin ve ark. 2015).



Şekil 3.8. Nesne algılama ve kutular örtüşse de ayrıntılar kaybolabilir; kişi örneği segmentasyonun zorluğunu, MS COCO'daki ortalama örtüşme ise problemin boyutunu göstermektedir (Lin ve ark., 2015).

3.4. YOLO versiyonlarını karşılaştırma:

YOLO ağının V1'den V5'e ana iyileştirme önlemleri:

- YOLO: Şebeke bölümü (grid division) tespitten, güven kaybından sorumludur;
- YOLO V2: K-araçları eklenmiş çapa, iki aşamalı eğitim, tam evrişimli ağ;
- YOLO V3: FPN kullanarak çok ölçekli algılama;

- YOLO V4: SPP, MISH etkinleştirme işlevi, veri geliştirme Mozaik/Karıştırma, GIOU (Generalized Intersection over Union) kayıp işlevi;
- YOLO V5: Model boyutunun esnek kontrolü, Hardswish etkinleştirme işlevinin uygulanması ve veri geliştirme.

Gerçek dünya senaryolarında YOLOv4 ve YOLOv5 arasında kapsamlı karşılaştırmalar gerçekleştirilmiştir. Orijinal YOLO mimarisi, 24 evrişim katmanından oluşmaktadır. Ancak, test sonuçları birbirine çok yakın nesneler veya nesne gruplarında zayıf performans göstermektedir. Bu düşük performansın temel nedeni, her ızgara hücresinde yalnızca iki kutunun tahmin edilmesidir.

Karşılaştırmalı olarak incelendiğinde, GoogLeNet ve VGG16 mimarileri hesaplama karmaşıklığı açısından farklılık göstermektedir. GoogLeNet yaklaşık 8,25 milyar işlem gerektirirken, VGG16 için bu değer 30,69 milyar işlem olup, bu durum GoogLeNet'in daha verimli olduğunu ortaya koymaktadır. YOLOv4, mevcut alternatiflere kıyasla daha hızlı ve daha doğru bir nesne algılama performansı sunmaktadır. Buna karşılık, YOLOv5 çoklu ağ mimarilerinin kullanımıyla daha esnek bir yapıya sahip olup, hafif model boyutu sayesinde farklı platformlara kolaylıkla uyarlanabilmektedir. Doğruluk açısından ise YOLOv5, YOLOv4 ile benzer seviyede sonuçlar elde etmektedir.

Bununla birlikte, YOLOv4'e kıyasla daha az yenilikçi olması nedeniyle YOLOv5 hakkında bazı çekinceler bulunmaktadır; ancak bu model, aşağıda sıralanan önemli avantajlar ve belirli performans iyileştirmeleri sunmaktadır:

- PyTorch tabanlı olması: PyTorch çerçevesi, kullanıcı dostu yapısı sayesinde veri kümelerinin eğitilmesini kolaylaştırmakta ve üretim ortamına entegrasyonu, YOLOv4'te kullanılan Darknet çerçevesine kıyasla daha pratik hale getirmektedir.
- Kod okunabilirliği ve entegrasyon kolaylığı: YOLOv5'in açık ve anlaşılır kod yapısı, farklı bilgisayarlı görü teknolojileriyle entegrasyonu kolaylaştırmaktadır.
- Modelin ortam kurulumu kolaydır, eğitimi yüksek hızda gerçekleştirilebilir ve toplu işlem yürütme ile gerçek zamanlı sonuçlar elde edilebilir.

3.5. Performans ölçümü

YOLOv5 Versus YOLOv3 for Apple Detection makalesinde algoritma performansını değerlendirmek için geleneksel karışıklık matrisi gösterimini kullanırdı (Tatiana ve ark. 2021).

- TP (True Positive) - algoritma tarafından doğru olarak tespit edilen elma sayısı;
- FP (False Positive) - algoritma tarafından yanlış olarak tespit edilen elma sayısı, tip I hataların, görüntüdeki arka plan nesneleri nedeniyle olabilir, algoritma tarafından elmalarla karıştırılabilir;
- FN (False Negative) - tip II hataların sayısı, yani algılanmayan elmalar.
- TP, FP ve FN kullanarak Recall F'yi hesaplayabiliriz Recall F $= \frac{TP}{TP+FN}$
- False NegativeRate $FNR = 1 - Recall\ F$
- Doğruluk $F = \frac{TP}{TP+FP}$
- $IoU = \frac{Örtüşme\ Alanı}{Birleşim\ Alanı}$

YOLOv4 ve YOLOv5 arasındaki bu karşılaştırmalı analiz, iki farklı metodoloji kullanılarak yapılmaktadır. Bu çalışma için MS COCO veri seti, kapsamlı anotasyonları, geniş nesne kategorisi yelpazesi ve çevresel koşullardaki geniş çeşitlilik nedeniyle seçilmektedir. Bu özellikler, diğer veri setlerine kıyasla nesne tespiti modellerinin değerlendirilmesi için ideal bir seçim oluşturmaktadır. İHA platformu, gerçek dünya uygulamaları için uygunluğunu sağlamak ve pratik dağıtım sırasında karşılaşılabilecek potansiyel zorlukları belirlemek amacıyla titiz bir değerlendirme sonrasında seçilmektedir. Bu değerlendirme, küçük nesnelerin tespitindeki zorluklar ve İHA'nın hava perspektifinin getirdiği benzersiz zorluklar gibi kritik faktörleri ortaya koymaktadır. Bu bulgular, sistemin gerçek dünya koşullarında sağlam ve güvenilir performans sergilemesi için önemli bir rol oynamaktadır. Bu seçimler, genelleme yeteneği, hesaplama uygulanabilirliği ve gerçek dünya geçerliliği arasında bir denge kurmak amacıyla yapılmaktadır; bunlar, arama ve kurtarma (AKG) gibi İHA tabanlı uygulamaları geliştirmek için hayati öneme sahip olan unsurlardır

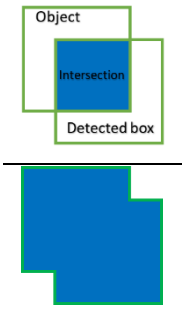
YOLO'nun her iki sürümünün performans değerlendirmesi, Ortalama Doğruluk (AP) ve

Ortalama Geri Çağırma (AR) gibi yaygın olarak kabul gören metrikler kullanılarak gerçekleştirilmiştir. Bu metrikler, tespit edilen nesnelerin doğruluğunu ve algoritmanın farklı Kesişim/Birleşim Oranı (IoU) eşiklerinde doğru pozitifleri tanıma kapasitesini ölçmekte olup, Microsoft, Facebook ve Common Visual Data Foundation (CVDF) gibi önde gelen araştırma kuruluşları tarafından da standart bir referans olarak benimsenmiştir. Bu nedenle, çalışmada güçlü ve güvenilir bir performans değerlendirmesi sağlamak amacıyla bu metodoloji tercih edilmiştir.

Değerlendirme, MS COCO veri setinden seçilen 20.288 gerçek dünya görüntüsü kullanılarak yapılmıştır. Analiz kapsamında, İHA uygulamalarında kritik öneme sahip araba, otobüs ve kişi sınıfları incelenmiş ve her iki YOLO sürümünün bu hedefleri tespit etme performansı karşılaştırılmıştır. MS COCO veri seti çok geniş bir nesne kategorisi yelpazesi sunsa da, mutfak eşyaları veya mobilya gibi İHA senaryolarıyla doğrudan ilişkili olmayan sınıfların dahil edilmesi çalışmanın odaklanmasını zayıflatacak ve hesaplama verimliliğini azaltacaktır. Bu nedenle yalnızca İHA uygulamaları açısından anlamlı ve kritik nesne sınıfları üzerinde karşılaştırmalı bir analiz gerçekleştirilmiştir.

Bu çalışma genel olarak hedef tespiti modelini doğrulamak için model göstergelerini kullanmaktadır; Ortalama Hassasiyet (AP) ve Ortalama Hatırlama (AR) metrikleri, algoritma performansını değerlendirmek için kullanılmaktadır. Bu göstergeler, farklı değişkenlere bağlıdır: IoU, alan ve maxDets. Çizelge 4.1–4.3'te görülebileceği gibi, Kesişim/Birleşim Oranı IoU, anlamına gelir ve Şekil 3.9'de formüle edilebilir ve görselleştirilebilir.

$$IoU = \frac{\text{Örtüşme Alanı}}{\text{Birleşim Alanı}} =$$



Şekil 3.9. Kesişim/Birleşim Oranı (IoU) metriği

Bu, tahmin edilen pozisyonun, nesnenin gerçek pozisyonu ile ne kadar iyi örtüştüğünü gösteren bir metriktir. Hassasiyet ve hatırlamayı değerlendirmek için kullanılır. Örneğin, 0.50, 0.50 IoU'ya sahip tespitlerin sayılacağı anlamına gelir. Dolayısıyla IoU arttıkça, hassasiyet düşmesine neden olmaktadır.

Gözlemlenen performans farkları, farklı IoU eşiklerinde YOLOv5'in genel olarak daha fazla esneklik ve güvenilirlik sağladığını göstermektedir. Düşük ve orta IoU eşiklerinde üstün hassasiyeti ve dayanıklılığı, onu gerçek zamanlı yanıt verme ve yüksek tespit doğruluğu gerektiren İHA uygulamaları için özellikle uygun hale getirmektedir; örneğin trafik izleme ve tarımsal değerlendirmeler gibi. YOLOv4 rekabetçi kalmakla birlikte, daha katı eşiklerdeki sınırlamaları daha belirgin hale gelmekte olup, hassas yer tespiti gerektiren görevlerde kullanımını etkileyebilir. Bu farklar göz önünde bulundurularak, uygulayıcılar, İHA uygulamalarının özel gereksinimlerine dayalı olarak uygun model ve IoU eşiklerini seçebilmektedirler.

3.6. Metodoloji için Anahtar Değerlendirmeler

• Sınırlı Batarya Ömrü

İHA sistemleri batarya ile çalıştığından, YOLO gibi hesaplama açısından yoğun algoritmalar enerji tüketimini önemli ölçüde artırabilir ve bu durum uçuş süresinin kısalmasına yol açar. Gerçek zamanlı görevlerin etkin bir şekilde yürütülebilmesi için, enerji tüketiminin optimize edilmesi ve operasyonel sürenin yeterli düzeyde tutulması kritik bir gerekliliktir.

• İşlem Gücü ve Bellek

YOLO algoritmasının İHA platformlarında etkin bir şekilde uygulanabilmesi, gerçek zamanlı performansın korunması için özel donanım çözümleri veya kapsamlı optimizasyonlar gerektirmektedir. Kullanılan donanımın hesaplama kapasitesi, tespit hızı ve doğruluk üzerinde doğrudan bir etkiye sahiptir. Bu çalışmada, farklı işlem kapasiteleri ve güç tüketim seviyelerine sahip üç ayrı donanım seçeneği karşılaştırılmıştır.

• **Yük Sınırlamaları**

İnsansız hava araçları, genellikle sınırlı taşıma kapasitesine (payload) sahiptir ve bu durum, üzerlerine entegre edilecek donanım bileşenlerinin seçiminde kritik bir faktör oluşturmaktadır. YOLO gibi derin öğrenme tabanlı nesne algılama algoritmalarının İHA'larda uygulanabilmesi, genellikle ek kamera sistemleri ve işlem birimleri (örneğin GPU'lar veya gömülü yapay zeka modülleri) gerektirmektedir. Ancak bu tür bileşenlerin eklenmesi, İHA'nın toplam ağırlığını artırarak hem uçuş süresi hem de manevra kabiliyeti gibi performans kriterleri üzerinde olumsuz etkilere yol açabilmektedir. Bu durum, özellikle küçük boyutlu ve hafif sınıf İHA'lar için daha belirgin olup, sistemin genel verimliliğini ve görev başarımını doğrudan sınırlayabilmektedir.

• **Veri İletimi**

İHA'lardan yere istasyonlarına işlenmek üzere iletilen yüksek çözünürlüklü video veya görüntü akışları, iletişim bant genişliğini zorlayabilir ve gecikmeye neden olabilir. Gerçek zamanlı nesne tespiti ve karar alma için verimli veri iletiminin sağlanması gereklidir.

• **Gerçek Zamanlı İşleme**

İnsansız hava araçlarında, gerçek zamanlı YOLO işleme yeteneği, çoğunlukla sınırlı yerleşik hesaplama kaynakları ile kısıtlanmaktadır. Bu tür sistemler, genellikle düşük güçlü işlemciler veya sınırlı GPU kapasitesine sahip gömülü platformlar üzerinde çalıştığından, derin öğrenme tabanlı algoritmaların yüksek hesaplama gereksinimlerini karşılamakta zorlanabilmektedir. Bu kısıtlama, sistemin tepki süresini doğrudan etkileyerek, özellikle hızlı nesne hareketlerinin veya aniden değişen çevresel koşulların söz konusu olduğu dinamik senaryolarda görev başarımını düşürebilmektedir. Dolayısıyla, gerçek zamanlı uygulamalarda yüksek doğrulukla birlikte düşük gecikme sağlayacak şekilde verimli model tasarımı ve optimizasyon teknikleri kritik önem taşımaktadır.

• **Nesne Boyutu ve Mesafe**

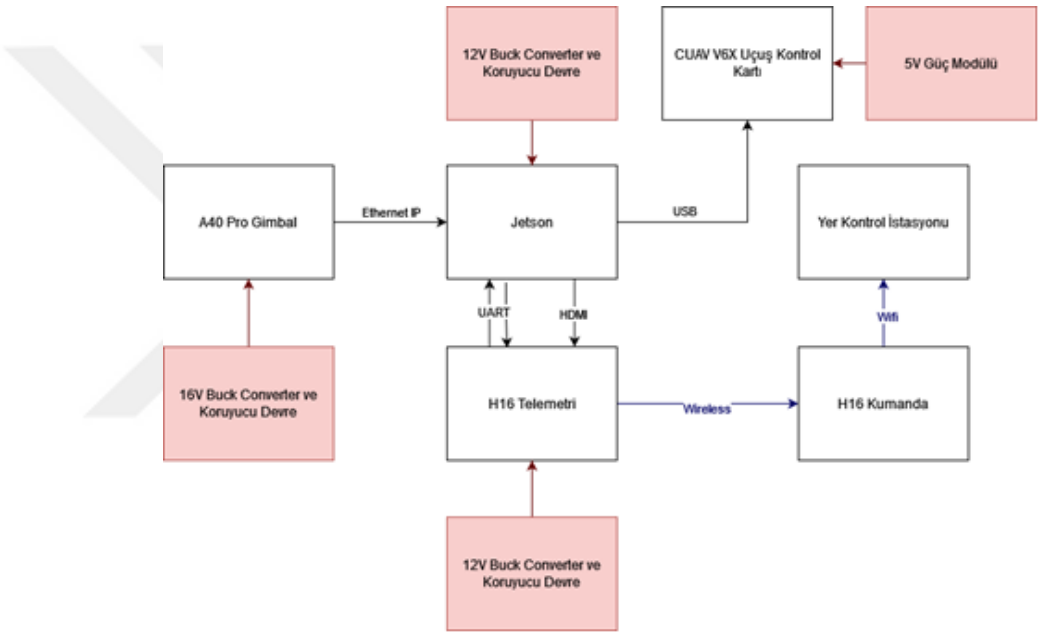
YOLO algoritmaları, küçük veya uzak nesneleri tespit etmekte zorluk yaşayabilir, bu da arama-kurtarma veya yaban hayatı izleme gibi İHA uygulamaları için önemli bir

sınırlamadır. Bu zorluğu aşmak için algoritmanın ve kamera ayarlarının ince ayar yapılması gerekmektedir.

• Maliyet Etkileri

YOLO'nun İHA'lara uygulanması, donanım, yazılım ve eğitim yatırımlarını içerir, bu da operasyonel maliyetleri artırabilir. Şekil 3.24'ta gösterildiği gibi, daha yüksek donanım performansı genellikle artan maliyetlerle ilişkilidir.

3.7. Kullanılan Cihazlar ve Ekipmanlar

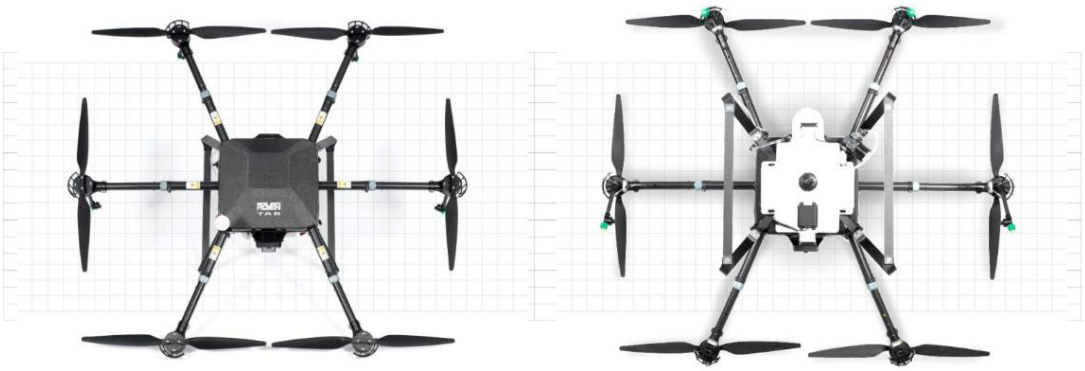


Şekil 3.10. Elektronik yerleşimi

3.7.1. İHA platformu

Bu araştırmada kullanılan hava platformu, Şekil 3.11'de gösterildiği üzere, endüstriyel denetim, hava haritalama ve gözetleme gibi yüksek taşıma kapasitesi gerektiren uygulamaları desteklemek amacıyla tasarlanmış ağır yük sınıfı bir çok rotorlu insansız hava aracıdır (İHA). Araç, dayanıklı ve modüler bir gövde yapısına sahiptir. Açık konfigürasyonda 1755 × 1520 × 690 mm boyutlarına sahip olan sistem, katlanabilir yapısı sayesinde 950 × 880 × 550 mm ölçülerine kadar kompakt hâle getirilebilmektedir. Yanlara doğru gerçekleşen katlanma mekanizması, hızlı ve güvenli şekilde kol

toparlanmasına olanak tanımakta, bu da taşınabilirlik ve sahada konuşlandırma verimliliğini artırmaktadır.



Şekil 3.11. IHA platformu

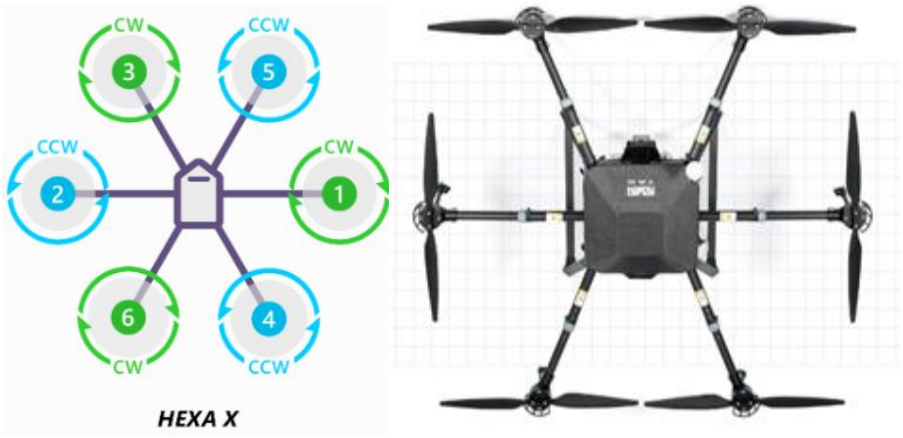
İHA Şekil 3.11'de gösterilmektedir, motorlar arası 1620 mm çapraz (diagonal) mesafesiyle geniş ve dengeli bir uçuş zarfı sunmakta olup, yüksek hassasiyet gerektiren görevler için uygun aerodinamik yapı sergilemektedir. Platform, 12S yapılandırmalı, 16.000 mAh kapasiteye sahip lityum polimer (LiPo) batarya sistemiyle çalışmakta ve uzun süreli görevler için gerekli enerjiyi sağlamaktadır. Boş durumdaki kalkış ağırlığı (iki batarya dahil) 19 kg'dır ve platform, 16 kg'a kadar faydalı yük taşıma kapasitesine sahiptir. Bu yüksek taşıma kapasitesi, gelişmiş sensörler, gimbal sistemleri ve yardımcı donanımların uçuş performansından ödün vermeden entegre edilebilmesine olanak tanımaktadır.



Şekil 3.12. Uçuş testleri

Tam yük altında, Şekil 3.12'de gösterildiği üzere, sistem yaklaşık 12 dakika boyunca kesintisiz uçuş gerçekleştirebilmekte olup, bu süre hızlı müdahale gerektiren görevler için yeterlidir. Maksimum uçuş irtifası, yasal düzenlemelere uygun olarak, yazılım tabanlı sınırlama ile yer seviyesinden 400 metre ile sınırlandırılmıştır. Ayrıca, yer kontrol sistemi açık alan koşullarında 10.000 metreye kadar kontrol menzili sunarak uzun mesafeli operasyonlar için güvenilir bir iletişim altyapısı sağlamaktadır.

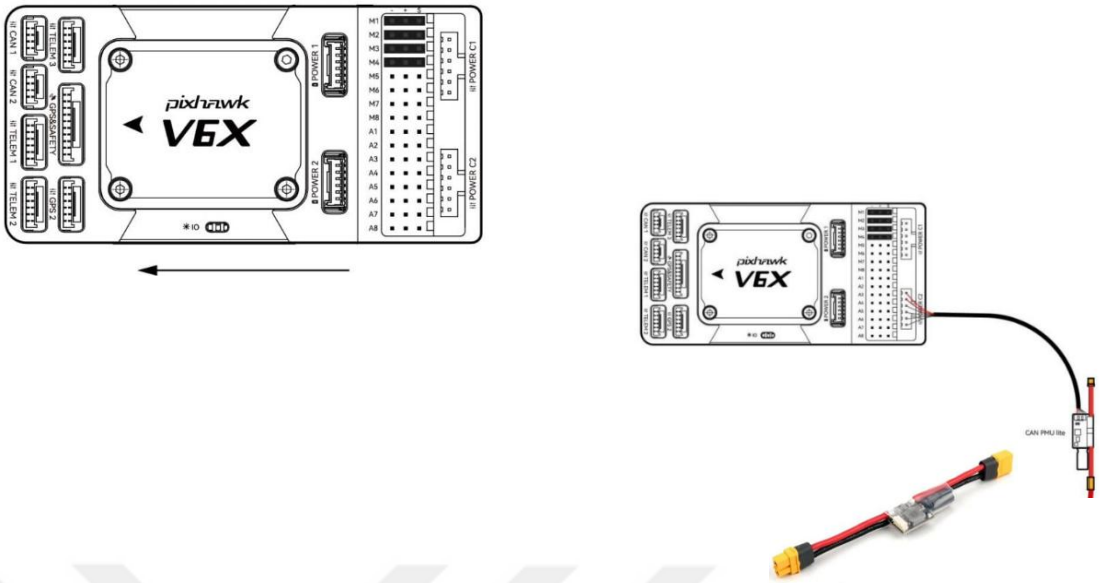
Şekil 3.12'de görüldüğü üzere, itici sistem 30 inç boyutunda katlanabilir kompozit pervaneler kullanmakta olup; bu yapı, hem aerodinamik verimliliği artırmakta hem de taşıma ve depolama kolaylığı sağlamaktadır. Genel olarak, İHA'nın mekanik tasarımı, güç sistemi ve yüksek taşıma kapasitesi, onu stabilite, dayanıklılık ve modüler entegrasyon gerektiren hassas hava uygulamaları için ideal bir platform hâline getirmektedir.



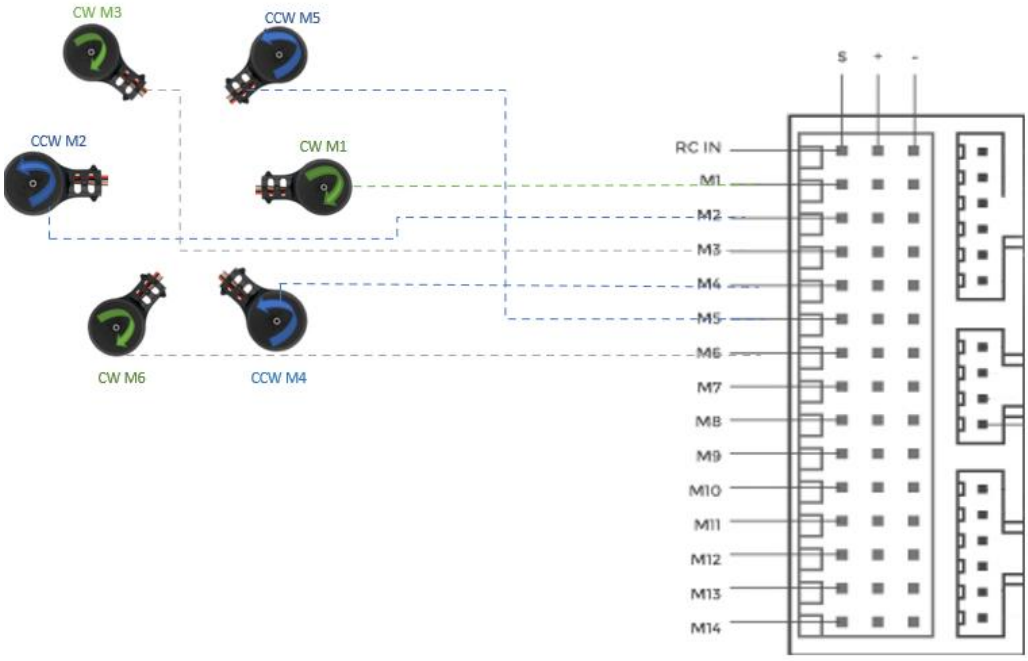
Şekil 3.13. Motor Numaraları ve Dönüş Yönleri

3.7.2. Uçuş Kontrol Sistemi

İnsansız hava araçları veya drone'lar, Şekil 3.14 - Şekil 3.18 ve Şekil 3.25'ta gösterildiği üzere, stabil uçuş sağlamak ve çeşitli görevleri yerine getirmek için birden çok bileşenin entegrasyonuna dayanan karmaşık sistemlerdir. Bu bileşenler arasında en kritik olanlardan biri, drone'un "beyni" olarak kabul edilen uçuş kontrol sistemidir (FC). Uçuş kontrol sistemi, farklı sensörlerden gelen verileri işleyerek pilotun komutlarını motor hareketlerine dönüştürür ve böylece aracın dengeli ve kontrollü uçuşunu sağlar. Bu açıklamada, drone uçuş kontrol sistemlerinin çalışma prensipleri, insansız hava araçlarının kontrol mekanizmaları ve ilgili bileşenler ile sistemler arasındaki temel farklar ele alınacaktır.



Şekil 3.14. Pixhawk CUAV V6X (Pixhawk. 2025)



Şekil 3.15. CUAV V6X Motor Bağlantı Pinleri (Pixhawk. 2025)

Bir İnsansız Hava Aracı'nın (İHA) merkezi yönetim birimi olan uçuş kontrolcüsü, aşağıdaki kritik işlevleri yerine getirmektedir:

- Stabilizasyon ve Denge Kontrolü

Uçuş kontrolcüsü, üzerinde bulunan atalet ölçüm birimi (IMU), barometre ve diğer sensörlerden alınan gerçek zamanlı verileri işleyerek platformun üç eksenli (roll, pitch, yaw) açısal konumunu ve dikey eksendeki yerden yüksekliğini sürekli izlemektedir. Bu verilere dayanarak, aerodinamik bozucu etkilere (rüzgâr, türbülans) karşı koymak ve kararlı bir uçuş duruşu (attitude) sağlamak amacıyla motor devirlerini dinamik olarak ayarlayan dengeleyici algoritmalar (PID kontrolcüler) yürütmektedir.

- Hareket Komut İşleme ve Aktüasyon

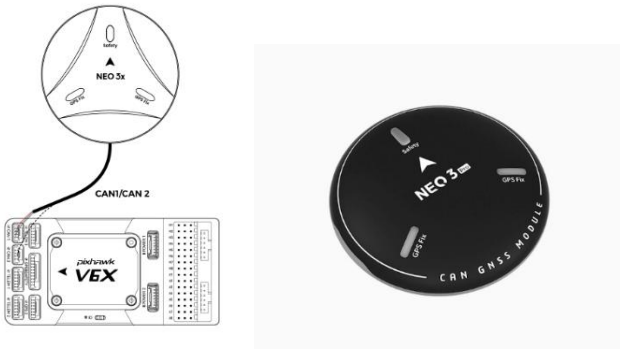
Kontrolcü, pilotun uzaktan kumanda ünitesinden veya yer istasyonu yazılımından gelen ham girdi sinyallerini (kumanda çıkışları) alıp yorumlayarak, İHA'nın istenen yönde ve eksende hareket etmesini sağlayacak şekilde motorlara iletilen elektriksel kumanda sinyallerine dönüştürmektedir. Bu kapsamda, itki kuvveti (throttle), yunuslama (pitch), yatış (roll) ve sapma (yaw) eksenlerindeki hareketlerin koordineli kontrolü gerçekleştirilmektedir.

- Telemetri Veri Yönetimi ve İletimi

Uçuş kontrolcüsü, platformun anlık durum bilgilerini (durumsal farkındalık) içeren telemetri verilerinin (örn. GNSS koordinatları, barometrik irtifa, yer hızı, batarya gerilimi, sensör sağlık durumu) toplanmasından, paketlenmesinden ve yer istasyonuna aktarılmasından sorumludur. Bu veri akışı, operatörün durumsal farkındalığını ve karar destek yeteneğini artırmaktadır.

- Güvenlik ve Otonom Acil Durum Yönetimi

Kontrolcü, gömülü güvenlik protokolleri aracılığıyla çeşitli arıza senaryolarına karşı özerk karşılık verme yeteneğine sahiptir. Bu kapsamda; kumanda sinyali kaybı (fail-safe), kritik batarya eşiğine ulaşılması, jeo-çit (geofence) ihlali gibi durumlarda otomatik olarak "Eve Dönüş" (Return-to-Home - RTH) prosedürünü başlatmak, sabit irtifada beklemek (auto-hover) veya kontrollü iniş (auto-land) gerçekleştirmek gibi otonom karar mekanizmaları bulunmaktadır.



Şekil 3.16. Neo3 Pro GPS alıcısı (Pixhawk. 2025)

3.7.3. Haberleşme Sistemi

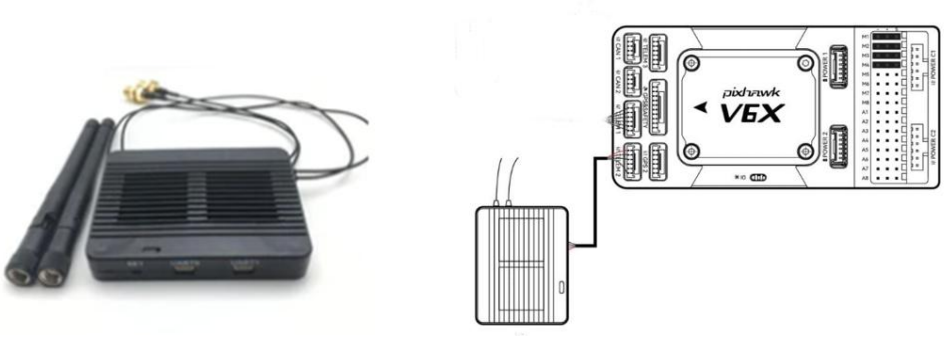
İHA kontrol birimleri, hava aracının uçuşunun, manevralarının ve çeşitli işlevlerinin kullanıcı tarafından etkin bir şekilde yönetilmesini sağlayan temel sistemlerdir. Bu kontrol sistemleri; elde taşınabilir vericiler, akıllı telefonlar ya da tablet tabanlı uygulamalar gibi farklı biçimlerde sunularak, kullanıcıya kullanım esnekliği ve kolaylığı sağlamaktadır.

Kontrol birimleri, kullanıcıdan gelen komutları yüksek doğrulukla İHA'ya iletmek amacıyla çeşitli sensörler ve kontrol algoritmaları kullanarak bu girdileri anlamlı hareket komutlarına dönüştürmektedir. Böylece İHA'nın yön, hız, irtifa ve görev işlevleri üzerinde hassas kontrol sağlanmaktadır.

Bununla birlikte, kontrol biriminin kapsama menzili, İHA'nın uzaktan kontrol edilebileceği maksimum mesafeyi belirler. Bu menzil, hem operasyonel verimliliğin korunması hem de iletişim güvenliğinin sağlanması açısından kritik öneme sahiptir.



Şekil 3.17. Skydroid H16 Pro Uzaktan Kumanda (Skydroid. 2025)

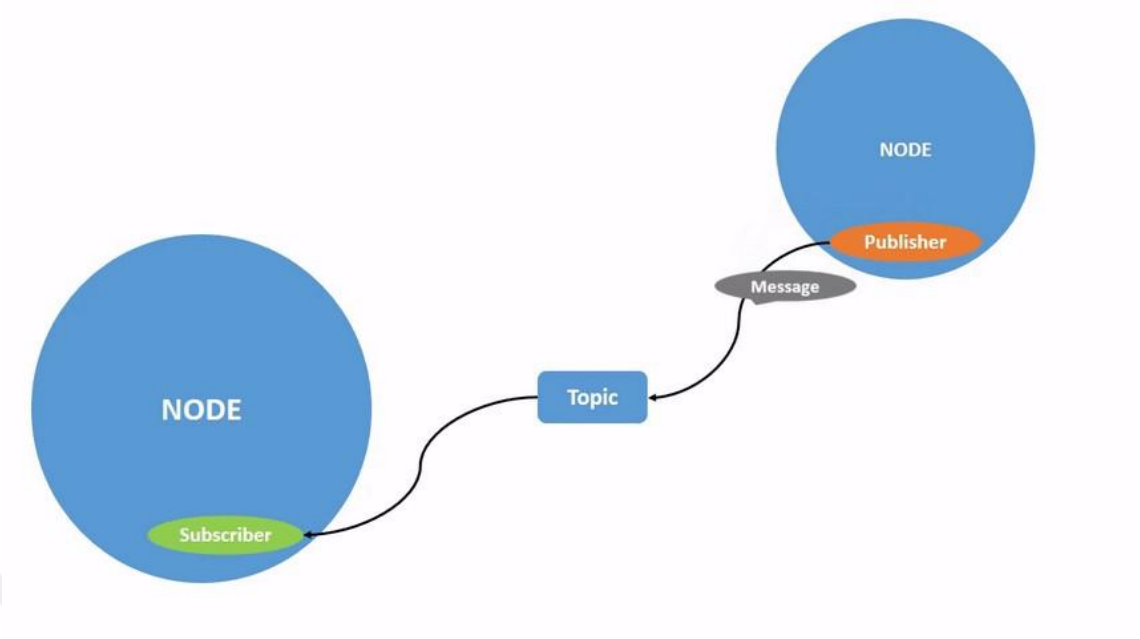


Şekil 3.18. Telemetri ve RF Kontrol sinyallerini alan ve gönderen Transmitter/receiver donanımı

3.7.4. Robot İşletim Sistemi (ROS)

Robot Operating System (ROS), farklı yazılım modüllerinin belirli görevleri yerine getirmek amacıyla, “topic” ve “message” yapıları üzerinden modüler bir şekilde haberleşmesini sağlamaktadır. UDP tabanlı bir iletişim mimarisine sahip olan ROS, temel mantık açısından MQTT protokolü ile benzerlik göstermektedir.

ROS altyapısında her bir yazılım bileşeni "node (düğüm)" olarak adlandırılmaktadır. Her bir node, belirli bir işlevi yerine getirmekte ve diğer node'lar ile veri alışverişinde bulunabilmektedir. Örneğin, sistemde yer alan bir YOLO node’u, yalnızca tespit edilen nesnelerin konum bilgilerini ve sınır kutularını içeren verileri bir topic üzerinden yayınlamakta (publish etmektedir). Buna karşılık, sistemde yer alan bir başka node olan gimbal takip birimi (gimbal tracker node), aynı topic’e abone olmakta (subscribe etmekte) ve aldığı verilere göre hedef nesneyi fiziksel olarak takip etmektedir.



Şekil 3.19. ROS Bağlantı Şeması (ROS. 2024)

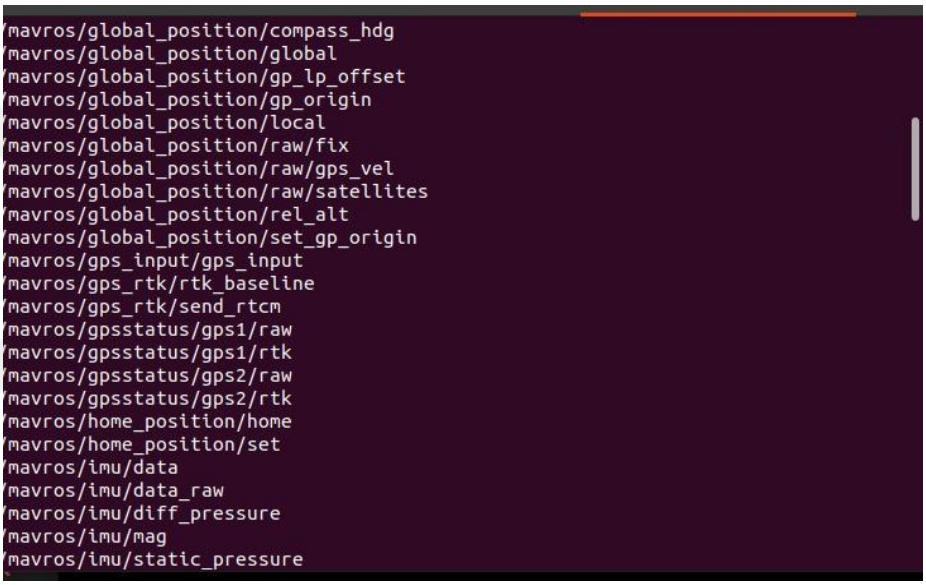
Bu mimari yaklaşım, Şekil 3.19'de ROS Mekanizması'nda da gösterildiği üzere, birden fazla node'un aynı veriye eş zamanlı olarak erişebilmesini mümkün kılmakta ve böylece sistem genelinde yüksek seviyede modülerlik sağlanmaktadır. Aksi durumda, örneğin YOLO verilerinin başka bir yazılıma aktarılması için verilerin elle paketlenmesi, farklı bir soket protokolü üzerinden gönderilmesi ve hedef yazılım tarafından tekrar çözülmesi (decode edilmesi) gerekebilmektedir. Bu ise hem geliştirme sürecini uzatmakta hem de sistemde hata yapma riskini artırmaktadır.

ROS'un sağladığı bir diğer temel avantaj, bu publish/subscribe tabanlı yapı sayesinde farklı programlama dilleri ile geliştirilmiş yazılım bileşenlerinin birlikte çalışabilmesini mümkün kılmasıdır. Örneğin, nesne algılama algoritması olan YOLOv8 Python dili ile geliştirilmişken, kontrol yazılımları C++ ile yazılmış olabilir. ROS bu bileşenler arasında kesintisiz bir veri alışverişine imkân tanımaktadır.

Ek olarak, ROS mimarisi, donanım soyutlama (hardware abstraction) ilkesi doğrultusunda, simülasyon ortamlarından gerçek donanımlara geçişi son derece kolaylaştırmaktadır. Örneğin, bir drone'u simülasyon ortamında hareket ettirmek için "cmd_vel" topic'ine hız komutları gönderiliyorsa, aynı yapı gerçek uçuş donanımı olan

Pixhawk ile çalışırken de geçerli olmakta, böylece kodda herhangi bir değişiklik yapmaksızın geçiş mümkün hâle gelmektedir.

MAVROS, ArduPilot tabanlı uçuş kontrol yazılımları ile Robot Operating System (ROS) arasında haberleşmeyi sağlayan bir ROS paketidir. Bu arayüz, ROS ortamında oluşturulan mesajları alarak bunları MAVLink protokolüne dönüştürmekte ve uçuş kontrolcüsüne iletmektedir. Aynı şekilde, uçuş kontrolcüsünden alınan MAVLink tabanlı veriler de ROS mesaj formatına dönüştürülerek ROS altyapısındaki diğer düğümlerin erişimine sunulmaktadır.



Şekil 3.20. Mavros Kütüphanesi tarafından yayın yapılan konu başlıkları

MAVROS, ROS içinde yaygın olarak kullanılan konular (topics) üzerinden haberleşmeyi mümkün kılmaktadır. Örneğin, bir hava aracına hız komutu vermek için standartlaştırılmış bir ROS topic'i olan /mavros/cmd_vel kullanıma sunulmaktadır. Bu konuya yayın (publish) yapılan hız vektörleri, MAVROS tarafından otomatik olarak ilgili MAVLink komutlarına çevrilmekte ve uçuş kontrol birimine aktarılmaktadır.

Benzer şekilde, uçuş kontrolcüsünden gelen telemetri verilerine erişmek için ilgili konulara abone (subscribe) olunması yeterlidir. Örneğin, drone'un mevcut pozisyon, hız

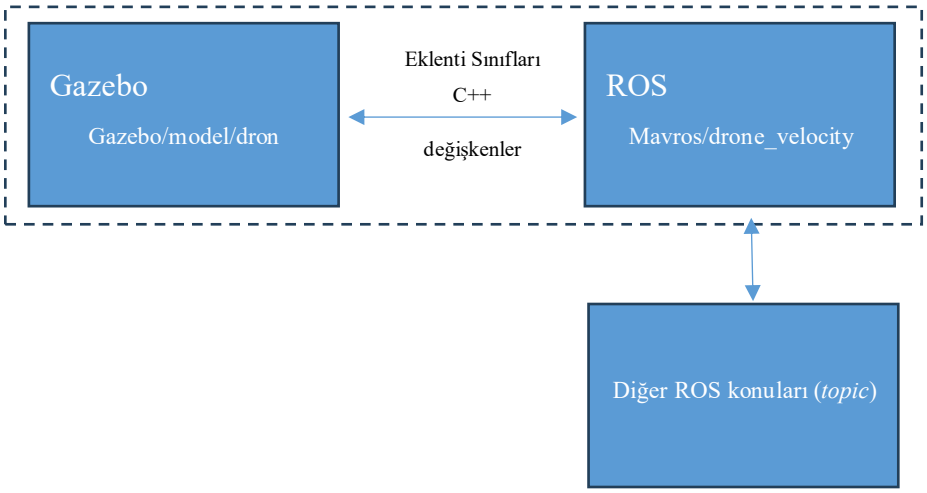
ya da sistem durumu gibi bilgileri, MAVROS tarafından sağlanan çeşitli konular üzerinden gerçek zamanlı olarak elde edilebilmektedir.

Bu yapı sayesinde, kullanıcılar uçuş kontrol sistemine doğrudan müdahale etmeksizin, yalnızca ROS içerisinden standart veri yayın ve abonelik işlemleri ile karmaşık görevleri kolaylıkla gerçekleştirebilmektedir. Böylece, ArduPilot ile çalışan insansız hava araçları ROS tabanlı sistemlere entegre edilmekte, hem yazılım geliştirme süreci hızlanmakta hem de modülerlik ve taşınabilirlik sağlanmaktadır.

Gazebo, benzer şekilde ROS (Robot Operating System) altyapısında olduğu gibi bir publish/subscribe (yayınla/abone ol) haberleşme mekanizması kullanmakta, ancak varsayılan olarak doğrudan ROS ile bağlı çalışmamaktadır. ROS ile Gazebo arasında doğrudan veri alışverişini sağlamak amacıyla özel olarak geliştirilmiş eklentiler (plugins) kullanılmaktadır.

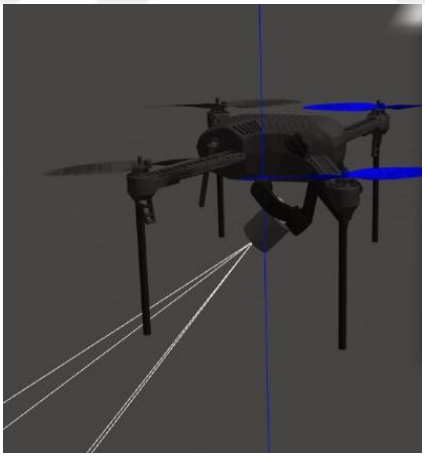
Bu eklentilerden biri olan ros_camera_plugin, Gazebo ortamındaki bir kameranın verilerine abone olmakta ve bu görüntü verilerini belirli bir ROS konusuna (örneğin /camera/image_raw) yayınlamaktadır. Böylece, simülasyon ortamında yer alan bir kamera, fiziksel bir kamera ile aynı veri akışını sağlayarak ROS içinde kolaylıkla kullanılabilmektedir.

Ancak, Gazebo ortamında elle oluşturulmuş özel modellerin – bu projede kullanılan gimbal, hareketli kişi (walking human) ve araçlar gibi – ROS ile haberleşebilmesi için özel Gazebo–ROS eklentilerinin yazılması gerekmektedir. Bu eklentiler, modelin hareket verilerini ROS mesajlarına dönüştürmekte ya da ROS üzerinden gelen komutları modele aktarmaktadır.



Şekil 3.21. ROS eklenti diagramları

Bu entegrasyon süreci ve veri akış yapısı, Şekil 3.21'de Gazebo–ROS Eklenti Mekanizması üzerinde şematik olarak gösterilmiştir. Söz konusu mimari sayesinde, simülasyon ortamındaki modellerin ROS ile çift yönlü etkileşimi mümkün hâle gelmekte ve gerçek donanım üzerinde test edilmeden önce tüm sistem bileşenlerinin simülasyon tabanlı doğrulaması gerçekleştirilebilmektedir.



Şekil 3.22. Gazebo Gimbal Model Eklentisi

Gimbal takip algoritmasının test edilmesi amacıyla, Gazebo simülasyon ortamında üç eksende dönebilen bir gimbal modeli gerçeklenmiştir. Bu model, fiziksel sistemin davranışlarını simüle etmek ve algoritmanın doğruluğunu değerlendirmek için kullanılmıştır.

Gazebo ortamında nesne modellemeleri, SDF (Simulation Description Format) adı verilen, XML tabanlı tanımlama dosyaları aracılığıyla gerçekleştirilmektedir. Bu format, silindir, küre, küp gibi basit geometrik nesnelerin yanı sıra, daha karmaşık yapılar içeren modellerin tanımlanmasına da olanak tanımaktadır.
SDF dosyalarında, her bir nesnenin veya bileşenin:

- Hangi gövdeye (link) rijit olarak bağlı olduğu,
- Hangi eksen(ler) etrafında dönebileceği (eklemler – *joints*),
- Diğer bileşenlerle olan bağıl konum ilişkileri

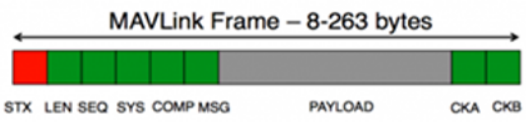
açıkça tanımlanmaktadır. Bu tanımlamalar, hem görsel simülasyonun doğruluğunu artırmakta hem de fiziksel hareketlerin dinamik olarak hesaplanmasını mümkün kılmaktadır.

Bu kapsamda geliştirilen üç serbestlik derecesine (3-DOF) sahip gimbal modeli, her eksendeki dönme hareketini temsil eden üç ayrı eklem (örneğin pitch, yaw, roll) ile yapılandırılmıştır. Eklemler arasında uygun bağlantılar (joint), gövdeler (link), atalet ve kütle parametreleri ile birlikte tanımlanarak, modelin dinamik simülasyonu sağlanmıştır. Bu yapı sayesinde, gimbal'ın gerçek donanımda sergileyeceği davranışlar simülasyon ortamında doğru şekilde temsil edilmekte ve ROS ile entegre takip algoritmaları bu sanal ortamda test edilebilmektedir.

3.7.5. MAVLink

MAVLink (Micro Air Vehicle Link), hafif yapılı, açık kaynak kodlu bir iletişim protokolüdür. Temel olarak insansız hava araçları ile yer kontrol istasyonları (GCS) arasında veri ve komut iletimini sağlamak amacıyla geliştirilmiştir. Bu protokol, aynı zamanda İHA bünyesindeki farklı bileşenler arasında da iletişimi mümkün kılar. MAVLink, telemetri verilerinin, komutların ve sistem durum bilgilerinin gerçek zamanlı olarak aktarılmasını sağlayarak ArduPilot ve PX4 gibi birçok İHA sisteminde temel bir ara yazılım katmanı (middleware) işlevi görmektedir.

MAVLink mesaj yapısı, protokolün güvenilirliğini ve düşük bant genişliğinde verimli çalışmasını sağlayacak şekilde sabit ve kompakt bir biçimde tasarlanmıştır. Her MAVLink mesajı, belirli alanlardan oluşan standart bir çerçeve (frame) yapısına sahiptir. MAVLink v1.0 için bu çerçeve; başlık baytı (start-of-frame), yük boyutu (payload length), paket sırası (packet sequence), sistem kimliği (system ID), bileşen kimliği (component ID), mesaj kimliği (message ID), veri yükü (payload) ve CRC (döngüsel artıklık kontrolü) alanlarını içerir. MAVLink v2.0 sürümünde ise bu yapıya ek olarak; bayraklar (incompatibility and compatibility flags), 24-bit mesaj kimliği ve opsiyonel olarak imza (signature) alanı eklenmiştir. Veri yükü alanı, uygulamaya bağlı olarak sabit ya da değişken uzunlukta olabilir ve taşıdığı bilgiler, mesaj türüne göre önceden tanımlanmış bir biçimde kodlanır. CRC alanı ise iletim sırasında oluşabilecek hataların algılanmasını sağlayarak veri bütünlüğünü garanti altına alır. Bu yapı sayesinde MAVLink, düşük bant genişliğine sahip haberleşme kanallarında dahi düşük gecikmeli, güvenilir ve tutarlı bir mesajlaşma altyapısı sunmaktadır.



Bayt İndeksi	İçerik	Değer	Açıklama
0	Paket başlangıç imzası	v1.0: 0xFE (v0.9: 0x55)	Yeni bir paketin başlangıcını gösterir.
1	Yük uzunluğu	0 - 255	Yük verisinin uzunluğunu belirtir.
2	Paket sırası	0 - 255	Her bileşen kendi gönderim sırasını sayar. Paket kayıplarını tespit etmeye yarar.
3	Sistem Kimliği	1 - 255	GÖNDEREN sistemin kimliği. Aynı ağ üzerindeki farklı MAV'leri ayırt etmeye yarar.
4	Bileşen Kimliği	0 - 255	GÖNDEREN bileşenin kimliği. Aynı sistemdeki farklı bileşenleri ayırt etmeye yarar (ör. IMU, otopilot).
5	Veri	(0 – 255) bayt	Mesajın verisi. Mesaj kimliğine bağlıdır.
(n+7) (n+8)	Sağlama toplamı (düşük bayt, yüksek bayt)	ITU X.25/SAE AS-4 hash, Paket başlangıç imzası hariç, bayt 1...(n+6). MAVLINK_CRC_EXTRA içerir. Aynı paketin farklı sürümlerini ayırt eder.	

Şekil 3.23. MAVLink mesaj yapısı (MAVLink. 2024)

MAVLink, sınırlı donanım kaynaklarına ve düşük bant genişliğine sahip iletişim ortamlarına yönelik olarak optimize edilmiş, seri ve ikili (binary) yapıda çalışan bir telemetri protokolüdür. Bu protokol, WiFi, 900 MHz radyo frekansı veya herhangi bir seri bağlantı gibi farklı taşıyıcı katmanlar (transport layers) üzerinden çalışabilmektedir. Her bir MAVLink mesajı belirli bir biçimsel yapıya sahiptir ve bu yapıda; Sistem Kimliği (System ID), Bileşen Kimliği (Component ID), Mesaj Kimliği (Message ID) ve veri yükü (payload) gibi temel alanlar bulunmaktadır. Sistem Kimliği, her bir hava aracını veya yer kontrol istasyonunu benzersiz olarak tanımlarken; Bileşen Kimliği, aynı sistem üzerindeki uçuş kontrol birimi, kamera gibi farklı alt bileşenlerin ayırt edilmesini sağlar. MAVLink protokolünün günümüzde kullanılan iki ana sürümü mevcuttur: v1.0 ve v2.0.

MAVLink 2.0 sürümü, v1.0 ile geriye dönük uyumluluğu koruyacak şekilde geliştirilmiş olup; daha büyük veri paketlerinin iletimine olanak tanıyan genişletilmiş mesaj yapısı, artırılmış güvenilirlik ve gelişmiş güvenlik mekanizmaları gibi önemli iyileştirmeler sunmaktadır. Açık kaynak kodlu ve modüler yapısı sayesinde MAVLink, farklı kullanıcı gereksinimlerine göre kolayca özelleştirilebilmekte ve çeşitli platformlara entegre edilebilmektedir. Bu yönüyle MAVLink, amatör kullanıcı uygulamalarından ticari ve askeri düzeydeki İHA sistemlerine kadar oldukça geniş bir uygulama alanı sunmaktadır. MAVLink, insansız hava araçlarının kontrolünü mümkün kılan yapısal bir mesajlaşma protokolü sunarak, drone ile yer kontrol istasyonu veya yardımcı bilgisayarlar arasında komutlar ve telemetri verilerinin iletimini sağlar. Protokol, konum, hız, batarya durumu gibi akışkan telemetri verilerinin yanı sıra kalkış, iniş, görev noktası (waypoint) takibi gibi kontrol ve yönlendirme komutlarını da desteklemektedir. MAVLink, özellikle komut mesajları üzerinden drone davranışının kontrolünü sağlayan kapsamlı bir komut protokol katmanına sahiptir. Örneğin, *MAV_CMD_NAV_TAKEOFF* (kalkış), *MAV_CMD_NAV_LAND* (iniş) ve *MAV_CMD_DO_REPOSITION* (yeniden konumlandırma) gibi komutlar doğrudan hava aracının eylemlerini tetiklemek amacıyla kullanılmaktadır. MAVLink mesajlarının büyük bir bölümü, veri iletiminde "en iyi çaba" (best-effort) yaklaşımını kullanırken, kritik komutlar için daha üst katmanda tekrar gönderim ve onaylama mekanizması içeren protokol yapısı sayesinde güvenilir iletim sağlanmaktadır.

İHA sistemleri ayrıca, düzenli olarak gönderilen *heartbeat* (Mesaj Kimliği 0) mesajları aracılığıyla aktiflik ve bağlantı durumu hakkında sürekli bilgi sunar; bu sayede yer kontrol istasyonu hava aracının durumunu anlık olarak izleyebilir. Bir İHA'nın MAVLink üzerinden kontrol edilmesi süreci temel olarak dört adımdan oluşmaktadır. İlk olarak, yer kontrol istasyonu veya yardımcı bilgisayar, seri bağlantı, TCP/UDP protokolü ya da radyo frekansı aracılığıyla İHA'nın MAVLink arayüzüne bağlanır. Daha sonra, motorların aktif hâle getirilmesi, kalkış, görev noktalarına yönelme, uçuş modlarının değiştirilmesi ve iniş gibi işlemleri gerçekleştirecek komut mesajları gönderilir. Üçüncü aşamada, İHA GPS konumu, irtifa, hız, batarya seviyesi ve çeşitli sensör verilerini içeren telemetri bilgilerini sürekli olarak geri iletir. Bu telemetri verileri, hem operatörlerin hem de otonom sistemlerin durumu izlemesine ve komutları gerektiği şekilde uyarlamasına

imkân tanır. Son aşamada ise, geliştiriciler sıklıkla *pymavlink* (Python) gibi kütüphaneleri kullanarak MAVLink mesajlarını programatik olarak üretip iletirler; bu da özel kontrol uygulamaları veya QGroundControl gibi yer istasyonu yazılımları ile entegrasyon sağlamaktadır.

Hareket komutlarına örnek olarak, yerel koordinat sisteminde belirli bir pozisyona gitmek için kullanılan *SET_POSITION_TARGET_LOCAL_NED* ve küresel GPS koordinatına yönelme amacıyla kullanılan *SET_POSITION_TARGET_GLOBAL_INT* komutları verilebilir. Bunun yanı sıra, *COMMAND_LONG* yapısında iletilen *MAV_CMD_COMPONENT_ARM_DISARM* (motorları aktifleştir/devre dışı bırak) ve *MAV_CMD_NAV_TAKEOFF* (kalkış başlat) gibi komutlar da temel kontrol için sıkça kullanılmaktadır. Özetle MAVLink, hafif yapılı ve standartlaştırılmış iletişim yapısıyla İHA'ların esnek ve güvenilir bir şekilde kontrol edilmesini sağlar. Mesaj tabanlı yapısı sayesinde operatörler ve otonom sistemler, uçuşu yönetebilir, durumu izleyebilir ve karmaşık görevleri farklı haberleşme altyapıları üzerinden güvenli biçimde gerçekleştirebilir.

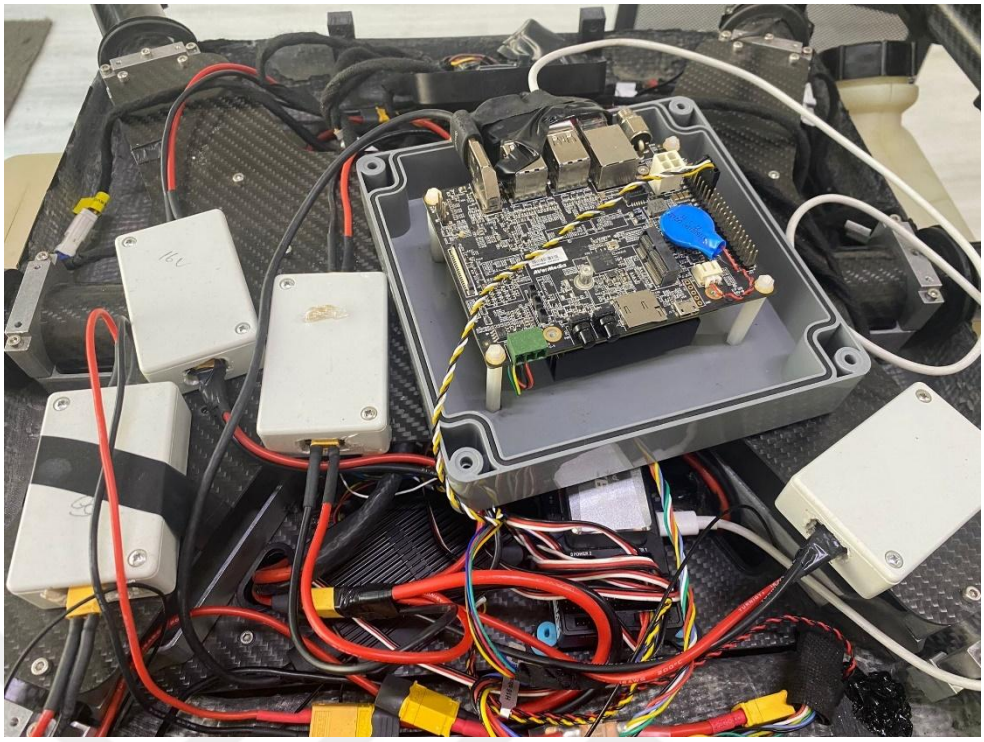
3.7.6. Gömülü Yapay Zekâ İşlemci Cihazı

Yapay zekâ işlem performansının karşılaştırılmasında yaygın olarak kullanılan bir metrik olan TOPS (Tera Operations Per Second), bu çalışmada değerlendirme ölçütü olarak dikkate alınmıştır. TOPS, bir yapay zekâ hızlandırıcısının tam yük altında saniyede gerçekleştirebildiği işlem (operasyon) sayısını ifade eden, sadeleştirilmiş fakat kıyaslama açısından işlevsel bir göstergedir. Başka bir ifadeyle, bir donanımın belirli bir süre içerisinde çözebileceği matematiksel işlem miktarını yansıtır. Örneğin, 5 TOPS değerine sahip bir hızlandırıcı ile 15 TOPS değerine sahip başka bir hızlandırıcı karşılaştırıldığında, teorik olarak ikinci sistemin birincisinden üç kat daha yüksek işlem kapasitesine sahip olduğu sonucuna varılabilir.

Özellikler	Orin Nano 4GB	Orin Nano 4GB (Super)	Orin Nano 8GB	Orin NX 8GB	Orin NX 8GB (Super)	Orin NX 16GB
GPU Çekirdek	512 CUDA Çekirdek 16 Tensor Çekirdek	512 CUDA Çekirdek 16 Tensor Çekirdek	1024 CUDA Çekirdek 32 Tensor Çekirdek	1024 CUDA Çekirdek 32 Tensor Çekirdek	1024 CUDA Çekirdek 32 Tensor Çekirdek	1024 CUDA Çekirdek 32 Tensor Çekirdek
GPU Max Frekans	625 MHz	1020 MHz	625 MHz	765 MHz	1173 MHz	918 MHz
Maks YZ PERF INT8	20 TOPS (seyrek) 10 TOPS (yoğun)	34 TOPS (seyrek) 17 TOPS (yoğun)	40 TOPS (seyrek) 20 TOPS (yoğun)	70 TOPS (seyrek) 35 TOPS (yoğun)	117 TOPS (seyrek) 58 TOPS (yoğun)	100 TOPS (seyrek) 50 TOPS (yoğun)
GPU Perf (S/D)	20/10 TOPs	34/17 TOPs	40/20 TOPs	50/25 TOPs	77/38 TOPs	60/30 TOPs
CPU	6x A78 1.5 GHz	6x A78 1.7 GHz	6x A78 1.5 GHz	6x A78 2.0 GHz	6x A78 2.0 GHz	8x A78 2.2 GHz
int oran	106	118	106	130	167	167
DLA Perf (S/D)	NA	NA	NA	20/10 TOPs	40/20 TOPs	40/20 TOPs
DRAM BW	34 GB/s	51 GB/s	68 GB/s	102 GB/s	102 GB/s	102 GB/s
REFERANS GÜÇ MODU	7W | 10W	10W | 25W | MAXN SUPER	7W | 15W	10W | 15W | 20W | MAXN	10W | 15W | 20W | 40W | MAXN SUPER	10W | 15W | 25W | MAXN

Şekil 3.24. Jetson Modelleri Teknik Özellikler (NVIDIA Jetson. 2025)

Tablo verilerine göre, NVIDIA Jetson AGX Orin platformu 275 TOPS işlem gücü sunarken, Jetson Orin NX 16 GB modeli 100 TOPS, Jetson Xavier AGX 64 GB modeli ise 32 TOPS kapasitesindedir. Bu değerler ışığında, Orin NX'in Xavier AGX'e göre yaklaşık üç kat daha yüksek işlem gücüne sahip olduğu, AGX Orin'in ise Orin NX'e kıyasla yaklaşık 2,75 kat daha fazla performans sunduğu söylenebilir. Ancak, TOPS yalnızca teorik maksimum işlem kapasitesini ifade ettiğinden, gerçek uygulamalarda performansın belirlenmesinde bellek bant genişliği, güç tüketimi ve yazılım optimizasyonu gibi faktörlerin de dikkate alınması gerektiği unutulmamalıdır.



Şekil 3.25. Jetson Orin NX 16 GB modelin yerleşimi

3.7.7. Kamera ve Sabitleme Sistemi (Gimbal)

İnsansız Hava Aracı (İHA) modelinde kullanılacak kamera, hareket hâlindeki insanları tespit edebilmeyi mümkün kılmakta ve geliştirilen nesne tanıma ile takip algoritmaları sayesinde İHA'nın hedefi etkin bir biçimde takip edebilmesini sağlamaktadır. Bu bağlamda, projede tercih edilecek kameranın sahip olması gereken teknik ve işlevsel kriterler aşağıda maddeler hâlinde sunulmuştur:

- Kamera, hareket kabiliyetine sahip olmalı ve bu kapsamda bir **gimbal mekanizması** ile entegre çalışmalıdır.
- Görüntü verisi, en az **720p çözünürlükte** ve **gerçek zamanlı** olarak iletilebilir nitelikte olmalıdır.
- Kamera, veri alışverişinin sağlıklı gerçekleştirilebilmesi amacıyla **uygun bir seri haberleşme protokolü** (örneğin UART, CAN, vb.) ile donatılmış olmalıdır.
- Haberleşmenin yazılımsal olarak kolaylıkla sağlanabilmesi için **C++ veya Python** programlama dilleri ile uyumlu bir **resmî yazılım kütüphanesi** mevcut olmalıdır.

- İHA hareketlerinin hassas biçimde yönlendirilebilmesi adına, **gimbal açı ve açısal hız verileri anlık olarak elde edilebilmelidir.**
- Kameranın, belirli bir mesafeden hedef nesneyi tespit edebilmesi için **uygun bir görüş açısına (Field of View - FOV)** sahip olması gerekmektedir.
- Kamera, **optik veya dijital zoom** özelliği sayesinde görüş açısını değiştirebilir nitelikte olmalıdır.
- Gimbal sistemine yalnızca açı bilgisi değil, **kesintisiz ve kararlı takip** amacıyla **açısal hız verisi** de iletilebilmelidir.
- İlave bir avantaj olarak, kamera sisteminin **ArduPilot tabanlı uçuş kontrolcüleriyle uyumlu** çalışabilmesi, sistem entegrasyonu açısından tercih sebebi olmaktadır.
- Gerekli durumlarda, üretici firma ile **teknik destek ve mühendislik iletişimi kurulabilir** olmalıdır.

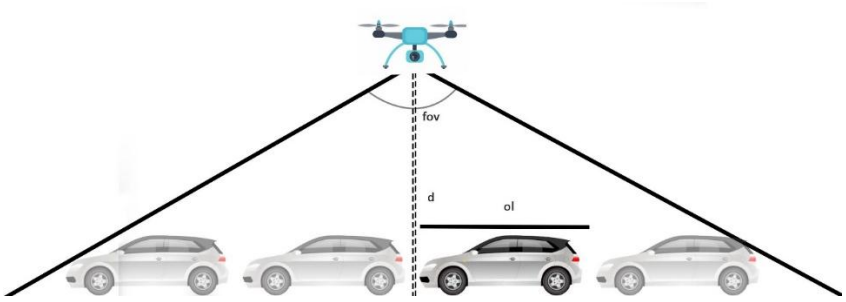
Bu çalışmada kullanılan görüntüleme sistemi, hava platformları ve insansız hava aracı (İHA) uygulamaları için özel olarak tasarlanmış, yüksek performanslı elektro-optik (EO) gimbal entegreli bir kameradır. Sistem, 1920×1080 piksel çözünürlüğe sahip tam HD görüntü sunan SONY 1/2.8″ CMOS sensör ile donatılmış olup, 2.13 megapiksel düzeyinde görüntü kalitesi sağlamaktadır. Kamera, 4.25 mm ile 170 mm arasında değişen odak uzaklığına ve F1.6 ile F4.95 arasında değişen diyafram açıklığına sahip 40× optik yakınlaştırma özelliğine ek olarak, 32× dijital yakınlaştırmayı da desteklemektedir. Bu sayede uzak hedeflerin ayrıntılı olarak görüntülenmesi, belirgin kalite kaybı yaşanmadan mümkün olmaktadır.

Sistem; otomatik görüntü sabitleme, progresif tarama modu, otomatik IR filtresi değişimi (ICR), arka ışık dengeleme ve sis azaltma (defog) gibi gelişmiş görüntüleme özelliklerine sahiptir. Bu sayede farklı çevresel koşullar altında optimum görüntüleme performansı sağlanmaktadır. Nesne tespitinde minimum mesafe 0.1 m ile 10.0 m arasında değişirken, yatay görüş açısı geniş açı konumunda 66.35°'den başlayarak telefoto ucunda 1.9°'ye kadar daralmakta ve bu özellik sistemin hem geniş alan gözetimi hem de noktasal hedef inceleme yeteneklerini desteklemektedir.

Gimbal sistemi, üç eksenli sabitleme mekanizmasına sahip olup ±0.02° hassasiyetle titreşim telafisi sağlamaktadır. Mekanik hareket sınırları ise Pitch ekseninde -60° ile +130°, Roll ekseninde ±40° ve Yaw ekseninde IP versiyonunda sınırsız olmak üzere ±300°'ye kadar uzanmaktadır. Sistem, PWM, TTL, S.BUS, TCP ve UDP protokolleri aracılığıyla çoklu kontrol girişlerini desteklemekte, böylece hassas ve esnek kontrol imkânı sunmaktadır.

Kamera, 30 Hz piksel sapma güncelleme frekansı ve 30 ms'den düşük gecikme süresi ile gerçek zamanlı nesne takibini desteklemektedir. Ayrıca yapay zekâ tabanlı otomatik yakınlaştırma ve çoklu hedef tespit özelliklerine sahiptir. Sistem, aynı anda insan ve araç gibi 10'a kadar farklı hedefi tanıyabilmekte olup; minimum %5 kontrast oranı ve en az 5×5 piksel hedef boyutu gereksinimi ile çalışmaktadır. Araç tespit doğruluğu %85'in üzerinde olup, yanlış alarm oranı %10'un altındadır.

Buna ek olarak sistem, micro-HDMI ve IP video çıkışı (RTSP/UDP) desteği sunmakta, 128 GB'a kadar TF kart ile yerel veri depolamasına olanak tanımakta ve EXIF meta verisine GPS koordinatları ile zaman bilgisini otomatik olarak ekleyebilen coğrafi etiketleme (geotagging) işlevine sahiptir. Tüm bu özellikler, sistemi İHA tabanlı gözetleme, keşif ve çevresel izleme görevleri için son derece uygun hâle getirmektedir.



Şekil 3.26. Kamera Görüş Alanı ve Takip Nesnesi

Takip edilecek hedefin ortalama boyutları ile hedefin tespit edilmesinin planlandığı mesafe biliniyorsa, kullanılacak kameranın sahip olması gereken görüş açısı (Field of View - FOV) da matematiksel olarak hesaplanabilmektedir. Bu bağlamda, İHA'nın takip edeceği nesneye olan mesafe "d", nesnenin ortalama boyutu ise "ol" olarak

tanımlanabilir. Yapay zekâ algoritmasının hedef nesneyi doğru şekilde tespit edebilmesi için, ilgili nesnenin kamera görüntüsünde görüş açısının yaklaşık dörtte birini kaplaması gerekmektedir. Bu varsayımla birlikte aşağıdaki trigonometrik denklem elde edilmektedir:

$$\frac{2ol}{d} = \tan\left(\frac{fov}{2}\right) \tag{3.1}$$

$$fov = 2\arctan\left(\frac{2ol}{d}\right) \tag{3.2}$$

Örneğin, ortalama bir kara aracı için ol=4,5m ve tespit mesafesi d=500m olarak alındığında, kameranın sahip olması gereken minimum görüş açısı yaklaşık olarak FOV≈2,06 ° olarak hesaplanmaktadır.

Belirtilen tüm teknik gereksinimler ve yapılan bu teorik hesaplama doğrultusunda, ViewPro A40 Pro model gimbal kamera tezi açısından uygun bulunmuştur. Söz konusu kamera sisteminin temel teknik özellikleri aşağıda sunulmaktadır:



Şekil 3.27. Viewpro A40 Pro (Viewpro. 2024)

Çizelge 3.1. ViewPro A40Pro Kamera Teknik Özellikleri.

Özellikler	**Değerler**
Çıkış	micro-HDMI (1080P 30fps/60fps) / IP (RTSP/UDP 720p/1080p 30fps)
Kontrol Yöntemi	PWM / TTL / S.BUS / TCP (IP çıkışı) / UDP(IPçıkışı)
Görüntü Sensörü	SONY 1/2.8" CMOS
Resim Kalitesi	Full HD 1080 (1920*1080)
Etkin Piksel	2.13MP
Optik Zoom	40x, f=4.25mm (geniş son)~170mm (tele son), F1.6~F4.95
Yatay Görüş Açısı	66.35° (geniş son) ~ 1.9°(tele son)
Yazılım Tasarımı Açı Aralığı	Pitch/Tilt: -45°〜125°, Yaw/Pan: ±290° / ±360°*N (IP çıkış)

3.8. Kalman Filtresinin Uygulanması

YOLO algoritması, nesne tespitini bir regresyon problemi olarak ele alır ve sinir ağı üzerinden yalnızca tek bir geçişle hem sınıf olasılıklarını hem de sınır kutusu (bounding box) koordinatlarını eşzamanlı olarak tahmin eder. Bu yaklaşım, yüksek işlem hızı ve uçtan uca (end-to-end) öğrenme kabiliyeti gibi avantajlar sunarak gerçek zamanlı uygulamalar için oldukça uygun bir çözüm hâline gelmiştir. Ancak, YOLO'nun ardışık video kareleri boyunca tespit edilen nesneleri birbirine bağlama yeteneği bulunmadığından, çoklu nesne takibi (MOT) senaryolarında — özellikle örtüşmeler (occlusion), hızlı hareketler veya eksik tespitler söz konusu olduğunda — çeşitli zorluklar ortaya çıkmaktadır.

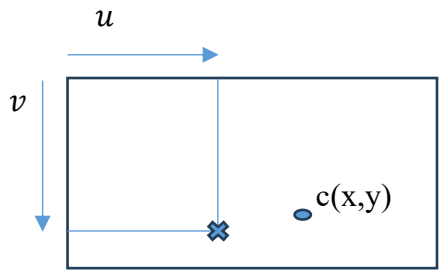
Bu sınırlamayı gidermek amacıyla, izleme uygulamalarında sıklıkla kullanılan Kalman Filtresi, YOLO ile birlikte entegre edilerek daha kararlı ve tutarlı bir takip sistemi elde edilmektedir. Kalman filtresi, doğrusal dinamik sistemlerin gürültülü gözlemler üzerinden durumunu tahmin etmek için kullanılan optimal ve yinelemeli bir Bayes filtresidir. Nesne takibi bağlamında Kalman filtresi, bir nesnenin konumunu ve hızını hem mevcut gözlemlere hem de geçmiş verilere dayanarak tahmin eder. Bu öngörü yeteneği sayesinde, nesne geçici olarak görünmediğinde ya da YOLO tarafından algılanamadığında dahi, nesnenin gelecekteki konumu makul bir doğrulukla tahmin edilebilir; bu da zamansal süreklilik ve sistem dayanıklılığı sağlar.

YOLO-Kalman tabanlı nesne takip sistemlerinde, YOLO her bir karede nesne sınıflandırması ve sınır kutusu tespiti işlevini üstlenirken, Kalman Filtresi ise her nesnenin zaman içerisindeki konum tutarlılığını sağlar. Bu entegrasyon çeşitli avantajlar sunmaktadır:

- Kalman Filtresi, tespitlerdeki ani değişimleri ve gürültüyü yumuşatarak daha stabil nesne yörüngeleri üretir;
- Geçici algılama kayıplarında nesne konumunu tahmin ederek takibin sürekliliğini sağlar;
- Zamansal tutarlılığı koruyarak her nesneye ait kimliğin (ID) kareler boyunca doğru şekilde atanmasını mümkün kılar, ki bu özellik davranış analizi ve otonom navigasyon gibi ileri düzey uygulamalar açısından kritiktir;
- Ayrıca Kalman Filtresi hesaplama açısından oldukça hafiftir ve bu yönüyle gerçek zamanlı YOLO tabanlı sistemlerle uyum içinde çalışabilir.

3.8.1. Gimbal Takip Algoritması

İlk aşamada, gimbal sisteminin hedefe yönelmesini sağlamak amacıyla, pan ve tilt eksenlerine ait hedef açıların hesaplaması gerçekleştirilmiştir. Şekilde gösterilen (u, v) vektörü, YOLO algoritmasına abone olunarak elde edilen hedef nesnenin piksel koordinatlarını temsil etmektedir. Bu koordinatlar, kameranın bilinen dikey ve yatay görüş alanı (VFOV ve HFOV) ile görüntü çözünürlüğü (genişlik ve yükseklik) kullanılarak gerçek dünya eksenlerindeki hedef açılarına dönüştürülmüştür. Böylece, pan (ψ) ve tilt (θ) eksenlerinde gimbal'in dönmesi gereken açılar belirlenmiştir. Hesaplanan bu hedef açı değerleri, bir orantı-integral (PI) kontrolörüne giriş olarak verilmekte ve bu kontrolör aracılığıyla ilgili eksenlere uygulanacak açısal hız komutları ($\dot{\theta}$ ve $\dot{\psi}$) üretilmektedir. Bu sayede, gimbal sistemi hedefin konumuna dinamik olarak tepki verebilmekte ve sürekli olarak hedefe yönelim sağlanabilmektedir.



Şekil 3.28. Hedefe Yönelim

$$e_x = c_x - u \tag{3.3}$$

$$e_y = c_y - v \tag{3.4}$$

(e_x, e_y) x ve y yönlerindeki hata vektörünü temsil etmektedir.

(c_x, c_y) Geçerli ya da ölçülen konumu değeri temsil etmektedir.

(u, v) İstenen veya hedeflenen konumu değeri temsil etmektedir.

$$e_{pan} = e_x \cdot \frac{fov_{horizontal}}{W} \tag{3.5}$$

$$e_{tilt} = e_y \cdot \frac{fov_{vertical}}{H} \tag{3.6}$$

Gimbal mekanizmasının hedefe yönelmesini sağlamak amacıyla, öncelikle pan (yatay eksen) ve tilt (düşey eksen) yönlerindeki hedef açıları hesaplanmıştır. Şekil 3.28'te gösterildiği üzere, (u,v) vektörü, YOLO düğümünden alınan ve hedef nesnenin görüntü düzlemindeki piksel cinsinden konumunu temsil etmektedir. Bu veriler, YOLO algoritmasının çıktılarına abone (subscribe) olunarak elde edilmektedir.

Hedefin görüntüdeki konumuna karşılık gelen gerçek dünya açılarını belirlemek için, kameranın dikey (vfov) ve yatay (hfov) görüş alanı değerleri ile birlikte, görüntünün genişlik ve yükseklik parametreleri kullanılmıştır. Bu parametreler yardımıyla, hedefin merkezden sapmasına karşılık gelen istenen pan ve tilt açıları trigonometrik dönüşümlerle hesaplanmıştır.

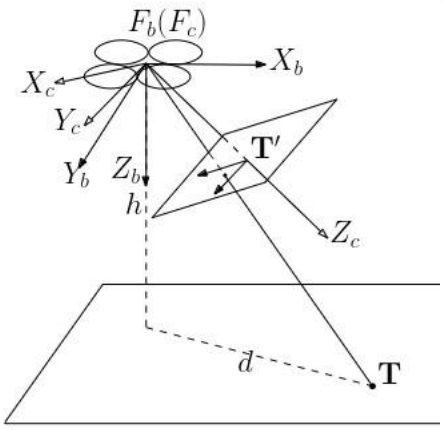
Elde edilen bu hedef açı değerleri ile gimbal'in mevcut açıları karşılaştırılmış ve her eksende dönmesi gereken fark açıları belirlenmiştir. Bu fark açıları, gimbal sistemine uygulanacak olan açısal hız komutları $\dot{\theta}$ (tilt için) ve $\dot{\psi}$ (pan için) elde edilmeden önce bir Oransal–Integral (PI) kontrolcüye giriş olarak verilmiştir. PI kontrol yapısı, sistemdeki hatayı azaltarak kararlı ve düzgün bir izleme hareketi sağlamaktadır.

Bu kontrol döngüsü, her kontrol çevriminde güncellenmekte ve hedefin hareketine karşılık gelen sürekli bir takip davranışı üretmektedir.



Şekil 3.29. Gimbal PI Kontrolcü

3.8.2. Hedef Konum Hesabı



Şekil 3.30. Bağıl Koordinatlar

YOLO algoritmasından elde edilen nesne konumu, görüntü düzleminde piksel birimi cinsindendir. Ancak bu konumun gerçek dünya koordinatlarına aktarılabilmesi için, öncelikle piksel cinsinden elde edilen verilerin metre cinsine dönüştürülmesi gerekmektedir. Bu dönüşüm, kameranın içsel (intrinsic) parametrelerinin — özellikle odak uzaklığı (focal length) ve optik merkez (principal point) gibi — önceden bilinmesini

gerektirir. Söz konusu parametrelerin elde edilmesi amacıyla, genellikle Python dili kullanılarak kamera kalibrasyonu işlemleri gerçekleştirilir. Kalibrasyon sürecinde kullanılan Python komut dosyaları, bu içsel parametreleri daha sonraki adımlarda kullanılmak üzere standart formatlarda kaydeder. Bu formatlar arasında en yaygın olanı, kamera iç parametrelerini temsil eden K iç matrisi (camera intrinsic matrix) olup, bu matris; odak uzaklığı ve optik merkezin piksel cinsinden değerlerini içeren temel bir dönüşüm bileşeni olarak kullanılır. K matrisinin doğru biçimde elde edilmesi, görüntü tabanlı koordinatların dünya koordinatlarına hassas ve güvenilir bir şekilde dönüştürülmesinde kritik rol oynamaktadır.

$$K = \begin{bmatrix} f_x & 0 & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

Kamera iç parametreleri arasında yer alan f_x ve f_y , yatay ve dikey eksenlerdeki piksel cinsinden odak uzaklıklarıdır ve kare piksel varsayımı altında genellikle eşit kabul edilir. Diğer yandan, ox ve oy değerleri, görüntü düzleminin sol üst köşesinden itibaren optik merkeze olan yatay ve dikey uzaklıkları ifade eder. Bu içsel parametrelerin tamamı piksel biriminde tanımlanır. Elde edilen piksel tabanlı nesne konumu, bu parametreleri içeren kamera iç matrisi (K) kullanılarak normalize edilebilir ve böylece kamera çerçevesinde metre cinsinden konum bilgisi elde edilebilir.

Ancak çoğu uygulamada yalnızca kamera çerçevesindeki konum bilgisi yeterli değildir. Gerçekleştirilecek kontrol ve navigasyon işlemleri için, nesnenin drone (vücut) çerçevesine göre konumu gereklidir. Bu durum, yalnızca kamera ile gimbal sistemine göre değil, doğrudan İHA'nın gövde koordinat sistemine göre konumlandırmayı zorunlu kılar. Bu bağlamda, koordinat dönüşümünde rotasyonların da hesaba katılması gerekir. Robotik sistemlerde bu tür dönüşümler, rotasyon matrisi olarak adlandırılan yapılarla gerçekleştirilir. Eğer iki koordinat çerçevesi arasındaki yönelim (rotasyon) biliniyorsa, bu

durumda ilgili noktanın bir çerçeveden diğerine dönüşümü rotasyon matrisi aracılığıyla gerçekleştirilebilir.

Bu çalışma bağlamında, kamera çerçevesinden drone gövde çerçevesine dönüşüm için gerekli rotasyon matrisi R_{BC}, gimbal pan ve tilt açılarına bağlı olarak oluşturulmuştur. Dolayısıyla, görüntü düzleminde elde edilen bir piksel konumunun drone çerçevesindeki üç boyutlu karşılığı aşağıdaki şekilde ifade edilir:

$$P_B = R_{BC} \cdot K^{-1} \cdot \begin{matrix} u \\ v \\ 1 \end{matrix} \tag{3.6}$$

Burada:

- P_B: Nesnenin drone (gövde) çerçevesindeki konumu,
- R_{BC}: Kamera çerçevesinden drone çerçevesine olan rotasyon matrisi,
- K^{-1} : Kamera iç matrisinin tersidir,
- $\begin{matrix} u \\ v \\ 1 \end{matrix}$: Homojenleştirilmiş piksel koordinatlarıdır.



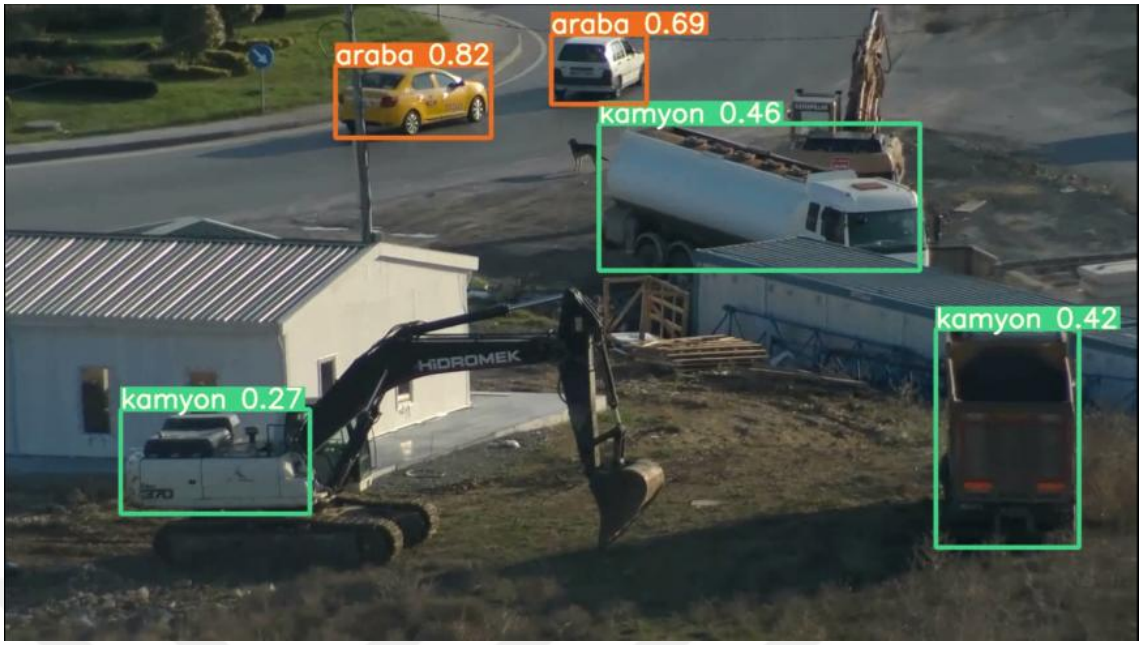
Şekil 3.31. Başakşehir bölgesinde drone ile kaydedilen bir videodan alınan görüntünün YOLOV4'e beslendi.



Şekil 3.32. Şekil 3.30'teki görüntünün YOLOv5 ile işlenmesi.



Şekil 3.33. Başakşehir yakınlarında drone'dan çektiğim bir videodan bir görüntü YOLOV4'e beslendi



Şekil 3.34. Şekil 3.33'teki aynı görüntü ancak YOLOV5'e aktarıldı

Şekil 3.33'e göre YOLOV4 doğruluğunun çok daha iyi olduğunu görebiliriz, NVIDIA Tesla K80 Cuda kullanarak ve Google ‘nin Colab hizmetini kullanan aynı test koşulları altında, sürüm 10.1, V10.1.243 bu sonuçlar elde edildi.

Şekil 3.31’den Şekil 3.34’e kadar aynı videodan yaklaşık 2,5 dakikalık bir kesit işlenmiştir. YOLOv5 modeli bu videoyu 87,16 saniyede ve yaklaşık 40,7 fps hızında işlerken, aynı koşullar altında YOLOv4 modeli 135,53 saniyede ve 26,2 fps hızında tamamlamıştır. Bu sonuçlar, YOLOv4’ün doğruluk açısından avantajlı olmasına karşın, YOLOv5’in önemli bir hız üstünlüğü sunduğunu göstermektedir. Dolayısıyla, doğruluk, hız, güç tüketimi ve yeni nesne algılama için eğitim süresi, en uygun algoritmanın seçilmesinde kritik faktörler olarak öne çıkmaktadır.

Bir İHA’nın nesne takibi uygulamalarında en uygun algoritmayı belirlemek amacıyla kapsamlı bir inceleme ve araştırma yapılmıştır. Mevcut çeşitli algoritmalar arasında, hız ve tespit doğruluğu açısından değerlendirme yapıldığında YOLOv4 ve YOLOv5, en uygun seçenekler olarak öne çıkmıştır. Bu iki algoritmanın karşılaştırılması için MS COCO veri seti kullanılmıştır.

Bu araştırma süresince elde edilen tüm bulgular ve literatür incelemeleri, önümüzdeki haftalarda seçilen bilimsel dergilerden birinde yayımlanması planlanan bir makalenin temelini oluşturmaktadır. YOLOv4 ve YOLOv5'in performans karşılaştırmasında, 20.288 farklı görüntü içeren MS COCO test-dev2017 veri seti kullanılmıştır.

4. BULGULAR ve TARTIŞMA

YOLOv4 ve YOLOv5 arasındaki karşılaştırma, farklı metriklerdeki performansları hakkında değerli bilgiler sunmaktadır.

Çizelge 4.1. MS COCO değerlendirme sunucusunda, YOLOv4 ve YOLOv5 algoritmaları için ortalama kesinlik (Average Precision) değerleri, "araba" kategorisi temel alınarak hesaplanmıştır.

Metrik Türü	**YOLOv4**	**YOLOv5**
AP (IoU=0.50:0.95, area=all)	0.456	0.506
AP (IoU=0.50, area=all)	0.715	0.750
AP (IoU=0.75, area=all)	0.492	0.549
AP (IoU=0.50:0.95, area=small)	0.358	0.392
AP (IoU=0.50:0.95, area=medium)	0.603	0.675
AP (IoU=0.50:0.95, area=large)	0.601	0.702
AR (IoU=0.50:0.95, area=all, maxDets=1)	0.193	0.211
AR (IoU=0.50:0.95, area=all, maxDets=10)	0.545	0.590
AR (IoU=0.50:0.95, area=all, maxDets=100)	0.603	0.643
AR (IoU=0.50:0.95, area=small)	0.508	0.534
AR (IoU=0.50:0.95, area=medium)	0.736	0.793
AR (IoU=0.50:0.95, area=large)	0.781	0.856

Çizelge 4.2. MS COCO değerlendirme sunucusunda, YOLOv4 ve YOLOv5 algoritmaları için ortalama kesinlik (Average Precision) değerleri, "otobüs" kategorisi temel alınarak hesaplanmıştır.

Metrik Türü	**YOLOv4**	**YOLOv5**
AP (IoU=0.50:0.95, area=all)	0.691	0.762
AP (IoU=0.50, area=all)	0.877	0.891
AP (IoU=0.75, area=all)	0.785	0.834

AP (IoU=0.50:0.95, area=small)	0.25	0.315
AP (IoU=0.50:0.95, area=medium)	0.588	0.658
AP (IoU=0.50:0.95, area=large)	0.803	0.876
AR (IoU=0.50:0.95, area=all, maxDets=1)	0.501	0.547
AR (IoU=0.50:0.95, area=all, maxDets=10)	0.762	0.83
AR (IoU=0.50:0.95, area=all, maxDets=100)	0.768	0.834
AR (IoU=0.50:0.95, area=small)	0.431	0.516
AR (IoU=0.50:0.95, area=medium)	0.710	0.771
AR (IoU=0.50:0.95, area=large)	0.857	0.923

Çizelge 4.3. MS COCO değerlendirme sunucusunda, YOLOv4 ve YOLOv5 algoritmaları için ortalama kesinlik (Average Precision) değerleri, "kişi" kategorisi temel alınarak hesaplanmıştır.

Metrik Türü	**YOLOv4**	**YOLOv5**
AP (IoU=0.50:0.95, area=all)	0.548	0.614
AP (IoU=0.50, area=all)	0.827	0.846
AP (IoU=0.75, area=all)	0.606	0.669
AP (IoU=0.50:0.95, area=small)	0.375	0.413
AP (IoU=0.50:0.95, area=medium)	0.621	0.696
AP (IoU=0.50:0.95, area=large)	0.719	0.824
AR (IoU=0.50:0.95, area=all, maxDets=1)	0.189	0.209
AR (IoU=0.50:0.95, area=all, maxDets=10)	0.554	0.614
AR (IoU=0.50:0.95, area=all, maxDets=100)	0.645	0.701
AR (IoU=0.50:0.95, area=small)	0.497	0.530
AR (IoU=0.50:0.95, area=medium)	0.711	0.776

AR (IoU=0.50:0.95, area=large)	0.807	0.889

Elde edilen sonuçlar, Çizelge 4.1–4.3 aralığında sunulmaktadır; burada AP, doğruluk ve geri çağırma eğrisinin altındaki alanı ölçmekte olup, modelin doğruluk (düşük yanlış pozitifler) ve geri çağırma (yüksek doğru pozitifler) arasındaki dengeyi çeşitli güven eşiği değerlerinde sağlama yeteneğini yansıtmaktadır. İHA uygulamalarında, AP, farklı koşullar altında tespit kalitesini değerlendirmek için kritik bir öneme sahiptir. Aşağıda, kullanım durumunun detaylı bir korelasyonu sunulmaktadır.

1. Gözetim ve İzleme: Askeri gözetim veya sınır izleme gibi uygulamalarda yüksek hassasiyet, yanlış pozitiflerin gereksiz uyarılara veya eylemlere yol açabileceği durumlarda hayati önem taşımaktadır. Örneğin, izinsiz giriş yapanlar veya araçların tespiti yüksek Ortalama Hassasiyet (AP) gerektirir, bu da tehdit olmayan nesnelerin yanlış bir şekilde tehdit olarak tanımlanmasını engeller. Hatırlama (Recall) da kritik nesnelerin (örneğin izinsiz giriş yapanlar) kaçırılmaması için önemlidir, ancak dengeli bir AP, tutarlı performans sağlamaktır.
2. Arama ve Kurtarma: Felaket senaryolarında, İHA'lar genellikle kaybolan kişileri veya kritik öğeleri bulmak için geniş alanları tarar. Yüksek AP, sistemin insanları veya araçları doğru bir şekilde tespit etmesini sağlar ve kurtarma çabalarını engelleyebilecek yanlış pozitifleri en aza indirir. Hız ve kaynak verimliliği için hassasiyet ve hatırlama arasında bir denge sağlanmalıdır.
3. Muayene ve Bakım: Endüstriyel İHA uygulamalarında, örneğin rüzgar türbinleri veya elektrik hatları gibi yapıların denetiminde, yüksek AP, kusurların veya anormalliklerin doğru bir şekilde tespit edilmesini sağlar, böylece kritik sorunların gözden kaçırılma riskini veya operasyonel maliyetleri artıran yanlış pozitifleri azalmaktadır.

YOLOv4 ve YOLOv5 arasındaki tespit performans farklarını daha detaylı incelemek için, Çizelge 4.4 ve 4.5, hesaplanan ortalama ve standart sapmayı sunmaktadır. Ayrıca, Şekil 4.1 ve 4.2, nesnelerin üç kategorisinde: kişi, otobüs ve araba, AP ve AR metriklerinin dağılımını görselleştiren kutu grafikleri sunmaktadır. Bu şekiller, YOLOv5'in hassasiyet ve hatırlama açısından YOLOv4'ü geride bıraktığını gösteren istatistiksel bulguları daha

da desteklemektedir. YOLOv5'in tutarlılığı ve azalan değişkenliği, kararlı nesne tespiti gerektiren uygulamalar için uygunluğunu vurgulamaktadır.

4.1. Tespit Doğruluğu

Tespit hassasiyeti açısından, YOLOv5, MS COCO veri seti ile yapılan değerlendirme sonuçlarıyla gösterildiği gibi, YOLOv4'e kıyasla daha iyi bir performans sergilemektedir. Ortalama Hassasiyet (mAP) skorları sürekli olarak YOLOv5 lehine sonuçlanmakta, bu da nesnelerin tanımlanmasındaki hassasiyet ve güvenilirliğinin arttığını vurgulamaktadır.

4.2. Hız ve Verimlilik

Aynı test ortamında, Çizelge 4.3'ten Çizelge 4.5'e kadar olan veriler hız karşılaştırmasını sunmaktadır; YOLOv4 ve YOLOv5 için sırasıyla 23,5 FPS ve 38,4 FPS olan sonuçlar, YOLOv5'in YOLOv4'e kıyasla daha yüksek bir saniye başına kare (FPS) oranı sağladığını göstermektedir. Bu hızdaki iyileşme, özellikle dinamik ortamlarda gerçek zamanlı uygulamalar için çok önemlidir.

4.3. Güven Değerleri

İHA tabanlı tespit sistemleri, dinamik ortamlarda hızlı karar verme süreçleri gerektirmektedir. Güven eşiğinin ayarlanması, gerçek zamanlı performans ile tespit kalitesi arasında bir denge kurmaya yardımcı olabilir. Daha yüksek bir eşik, düşük güvenli tespitleri ortadan kaldırarak hesaplama süresini kısaltabilirken, daha düşük bir eşik, daha fazla tespit işlenmesi gerektiği için süreci yavaşlatabilir. Şekil 4.1.'de gösterildiği gibi, YOLOv4, YOLOv5'e kıyasla biraz daha yüksek ortalama güven değerleri sergilemekte olmasına rağmen, YOLOv4 ve YOLOv5'in önceden eğitilmiş modellerinin doğasının yer seviyesindeki görüntüler için optimize edildiği göz önünde bulundurulmalıdır. Bu nedenle, daha uygun ve anlamlı bir karşılaştırma yapılabilmesi için VisDrone-Dataset gibi veri setlerinin kullanımı özellikle önem kazanmaktadır. Bu veri seti, drone'lara montelenmiş kameralarla çekilen çeşitli görüntü ve videolardan oluşan geniş bir koleksiyon sunarak değerlendirme için daha uygun bir bağlam sağlamaktadır.

4.4 İHA Uygulamaları İçin Uygunluk

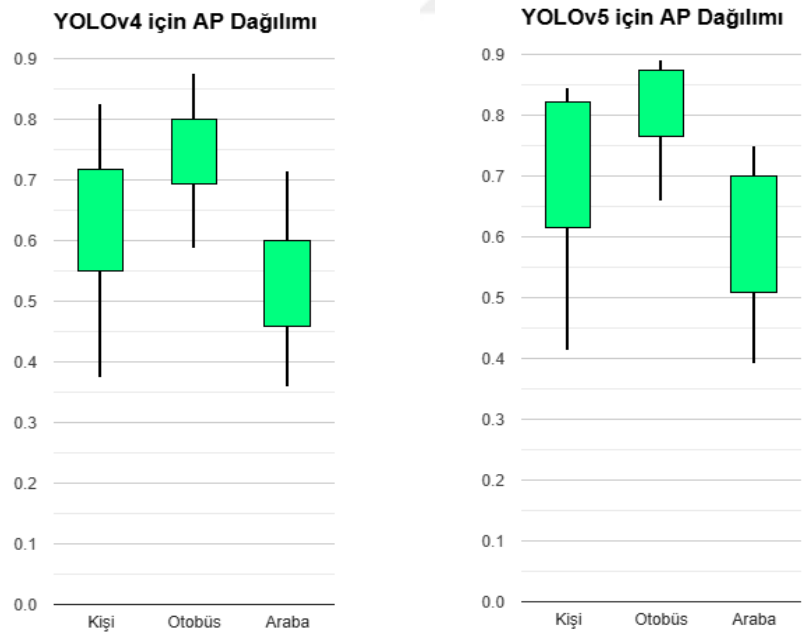
Çizelge 4.4. YOLOv4 ve YOLOv5 için kişi, otobüs ve araba nesne kategorilerinde Ortalama Hassasiyet (AP) değerlerinin ortalama ± standart sapma değerleri.

	Kişi	Otobüs	Araba
YOLOv4	0.62 ± 0.14	0.67 ± 0.21	0.54 ± 0.12
YOLOv5	0.68 ± 0.14	0.72 ± 0.20	0.60 ± 0.12

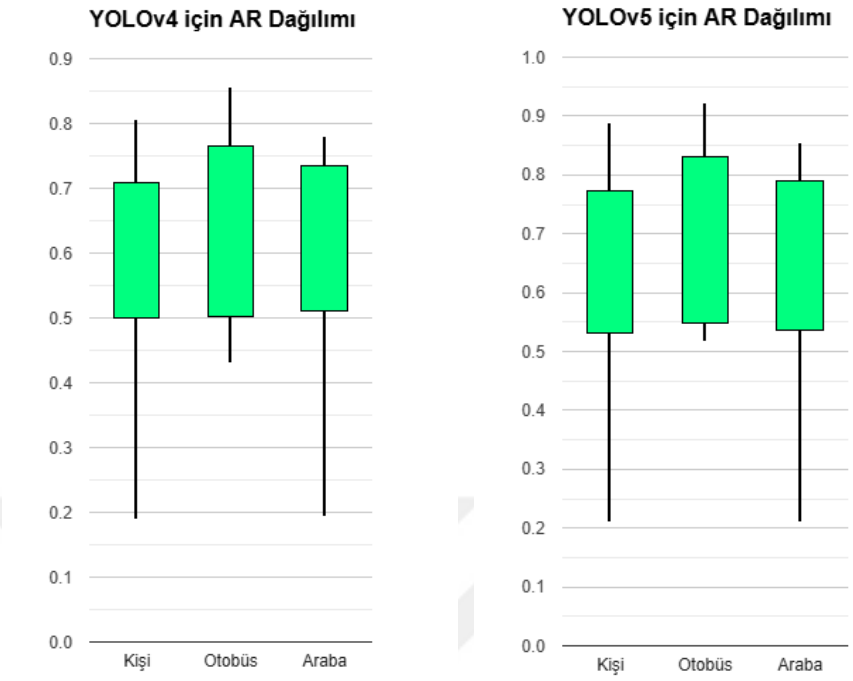
Çizelge 4.5. YOLOv4 ve YOLOv5 için kişi, otobüs ve araba nesne kategorilerinde Ortalama Hassasiyet (AR) değerlerinin ortalama ± standart sapma değerleri.

	Kişi	Otobüs	Araba
YOLOv4	0.57 ± 0.20	0.67 ± 0.15	0.56 ± 0.19
YOLOv5	0.62 ± 0.22	0.74 ± 0.15	0.60 ± 0.21

Çalışma, bu modellerin İHA ortamında değerlendirilmesinin önemini vurgulamaktadır. Sonuçlar, YOLOv5'in artan doğruluğu ve hızı ile İHA uygulamaları için daha uygun olduğunu ortaya koymaktadır.



Şekil 4.1. YOLOv4 ve YOLOv5 için kişi, otobüs ve araba kategorilerinde Ortalama Kesinlik (AP) dağılımını gösteren kutu grafiği.



Şekil 4.2. YOLOv4 ve YOLOv5 için kişi, otobüs ve araba kategorilerinde Ortalama Kesinlik (AR) dağılımını gösteren kutu grafiği.



Şekil 4.3. YOLO ve Kalman Filtresi kullanılarak hareketli bir aracın takibi



Şekil 4.4. Takip edilen nesne kaybolduğunda bile Kalman Filtresi konumunu tahmin edebilmektedir.



Şekil 4.5. Nesne yeniden göründüğünde takibin devam etmesi.



Şekil 4.6. YOLO'nun her nesneye kimlik (ID) ataması ve Kalman Filtresi ile birlikte çalışarak benzer nesneler arasında aynı nesnenin kaybolmasını önlemesi.

YOLO–Kalman tabanlı nesne takip sistemlerinde gerçekleştirilen bu entegrasyon çeşitli avantajlar sağlamaktadır. Öncelikle, Kalman Filtresi tespitlerdeki ani dalgalanmaları ve gürültüyü azaltarak daha kararlı nesne yörüngeleri üretmektedir. Bunun yanı sıra, geçici algılama kayıpları durumunda nesnenin konumunu tahmin ederek takibin sürekliliğini güvence altına almaktadır. Ayrıca, zamansal tutarlılığı koruyarak her bir nesneye ait kimliğin (ID) kareler boyunca doğru biçimde atanmasına imkân tanımaktadır. Bu özellik, davranış analizi ve otonom navigasyon gibi ileri seviye uygulamalar açısından kritik bir öneme sahiptir. Son olarak, Kalman Filtresi hesaplama açısından oldukça hafif bir yapıya sahip olduğundan, gerçek zamanlı YOLO tabanlı sistemlerle yüksek uyumluluk göstermektedir.

5. SONUÇ

Bu çalışmada, YOLOv4 ve YOLOv5 nesne tespit algoritmaları kapsamlı bir şekilde incelenmiş ve performansları hem standart bir referans olarak kabul edilen COCO veri seti hem de İHA platformlarından elde edilen gerçek video verileri üzerinden değerlendirilmiştir. Analizler, hız (FPS) ve doğruluk (AP) gibi temel performans metriklerine ek olarak IoU, alan ve maxDets gibi ölçütler üzerinden gerçekleştirilmiştir. Elde edilen sonuçlar, YOLOv5'in farklı koşullar altında YOLOv4'e kıyasla %5–16 oranında daha yüksek doğruluk sağladığını, işlem hızında ise 23,5 FPS'ye karşılık 38,4 FPS ile yaklaşık %63'lük bir artış sunduğunu göstermektedir. Bu bulgular, YOLOv5'in hem daha hızlı hem de daha doğru çalıştığını ve aynı performansı daha küçük, düşük maliyetli donanımlar üzerinde sunabilme potansiyeline sahip olduğunu ortaya koymaktadır. Bu durum, ağırlık ve enerji tüketiminin kritik önemde olduğu İHA uygulamaları açısından belirleyici bir avantajdır.

Çalışmada kullanılan yöntemler, Google Colab üzerinde Tesla T4 GPU kullanılarak bulut ortamında test edilmiştir. GPU türüne bağlı olarak performansın değişkenlik göstermesi beklenmektedir; örneğin V100 veya A100 gibi güçlü GPU'lar daha yüksek hızlar sunarken, NVIDIA Jetson serisi gibi uç cihazlarının daha düşük performans göstermesi olasıdır. Bununla birlikte, aynı donanım koşulları altında YOLOv5'in hem doğruluk hem de hız açısından YOLOv4'e üstünlük sağlaması beklenen bir durumdur.

Bulgular, bu algoritmaların gelecekte robotik uygulamalara entegrasyonu için değerli bir temel sunmaktadır. Özellikle YOLO ile Kalman Filtresi'nin entegrasyonu, çalışmanın önemli bir katkısı olarak öne çıkmaktadır. YOLO, nesne tespitinde yüksek doğruluk sağlarken, Kalman Filtresi tespitler arasında süreklilik ve öngörüye dayalı takip yeteneği kazandırmıştır. Bu hibrit yapı, özellikle hızlı hareket eden veya kısmen gizlenen nesnelerin takibinde tespit sürekliliğini artırmış, yanlış negatifleri azaltmış ve takip doğruluğunu anlamlı biçimde iyileştirmiştir. Böylece sistem, yalnızca tek karedeki tespit başarısına bağlı kalmayıp, zaman boyutunda da kararlı ve güvenilir sonuçlar üretebilmiştir. Bu durum, İHA uygulamalarında gerçek zamanlı hedef takibinin doğruluk, güvenilirlik ve kararlılık açısından daha üst bir seviyeye taşındığını göstermektedir.

Öngörü–düzeltme yapısına dayalı entegrasyon sayesinde, sınır kutularındaki titreşimler azalmış, nesne yörüngeleri daha akıcı ve fiziksel olarak tutarlı hale gelmiştir. Ayrıca, geçici görünmezlikler veya kaçırılan tespitler sırasında nesne kimliği korunmuş, nesne yeniden belirdiğinde takip kesintisiz devam etmiştir. Bu durum, kimlik değişimlerinin ve takip kopmalarının önüne geçerek sistemin sürekliliğini güçlendirmiştir. Kalman Filtresi'nin öngörü temelli tahmin yeteneği, özellikle hızlı hareket eden nesnelerde yanlış eşleşmeleri engelleyerek veri ilişkilendirmesini güçlendirmiştir. Yoğun ve karmaşık sahnelerde dahi bu entegrasyon, takip kararlılığını koruyarak sistemin doğruluğunu ve güvenilirliğini artırmıştır. Dolayısıyla, YOLO'nun kare bazlı güçlü tespit performansı ile Kalman Filtresi'nin zaman boyutunda sağladığı öngörü ve sürekliliğin birleşmesi, İHA uygulamalarında gerçek zamanlı hedef takibini doğruluk, güvenilirlik ve kararlılık açısından daha üst bir seviyeye taşımıştır.

Bununla birlikte, bazı sınırlılıkların göz önünde bulundurulması gerekmektedir. YOLOv5'in büyük varyantları (örneğin YOLOv5x) yüksek hesaplama gücü gerektirmekte olup, kaynak sınırlı cihazlarda gerçek zamanlı uygulamalarda kısıtlamalara yol açabilmektedir. Ayrıca, eğitim veri setlerine bağımlılık, temsil edilmeyen nesneler veya farklı çevresel koşullarda tespit doğruluğunun belirgin biçimde düşmesine neden olabilmektedir. Bu nedenle, sağlam ve güvenilir bir performans elde edebilmek için geniş kapsamlı, alan odaklı veri toplama ve veri artırma stratejilerinin uygulanması büyük önem taşımaktadır.

Sonuç olarak, YOLO ve Kalman Filtresi'nin entegrasyonu yalnızca anlık tespit doğruluğunu değil, zaman boyutunda sürekliliği de sağlayarak sistemin güvenilirliğini artırmış; böylece gerçek zamanlı, kesintisiz ve kararlı bir nesne takip mekanizması ortaya çıkarmıştır. Bu entegrasyonun, gelecekte trafik izleme, afet yönetimi, tarımsal gözetim ve güvenlik uygulamaları gibi kritik İHA tabanlı senaryolarda önemli katkılar sunacağı öngörülmektedir.

Gelecek çalışmalarda, karşılaştırmalı incelemenin DETR gibi Transformer tabanlı modern nesne tespit mimarilerini de kapsayacak şekilde genişletilmesi, İHA uygulamalarına yönelik daha kapsamlı bulgular sunacaktır. Ayrıca, YOLO ve Kalman

Filtresi'nin ileri seviye hareket tahmini ve uyarlanabilir filtreleme yöntemleri ile birleştirilmesi, özellikle karmaşık ve dinamik ortamlarda sistemin doğruluk ve güvenilirliğini daha da artıracaktır. Bu yöndeki çalışmalar, İHA'ların trafik gözetimi, afet yönetimi ve tarımsal izleme gibi kritik alanlarda daha güvenilir ve verimli kullanılmasını mümkün kılacaktır.

KAYNAKLAR

Kosasi, T., Sihombing, Z. A. L., & Husein, A. M. (2024). YOLO-based vehicle detection: Literature review. *Journal of Computer Networks, Architecture and High Performance Computing, 6*(3), 1384–1389. https://doi.org/10.47709/cnahpc.v6i3.4377

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. *arXiv.* https://doi.org/10.48550/arXiv.2004.10934

Jiang, P., Ergu, D., Liu, F., Cai, Y., & Ma, B. (2021). A review of YOLO algorithm developments. *Procedia Computer Science, 199,* 1066–1073. https://doi.org/10.1016/j.procs.2022.01.135

Lin, T.-Y., Maire, M., & Belongie, S. (2015). Microsoft COCO: Common objects in context. *arXiv.* https://arxiv.org/abs/1405.0312

Abba, S., Bizi, A. M., Lee, J.-A., Bakouri, S., & Crespo, M. L. (2024). Real-time object detection, tracking, and monitoring framework for security surveillance systems. *Heliyon, 10*(15), e34922. https://doi.org/10.1016/j.heliyon.2024.e34922

Albuquerque, C., Henriques, R., & Castelli, M. (2025). Deep learning-based object detection algorithms in medical imaging: Systematic review. *Heliyon, 11*(1), e41137. https://doi.org/10.1016/j.heliyon.2024.e41137

Pavate, A., Kukreja, S., Janrao, S., Bankar, S., Patil, R., & Bidve, V. (2025). Efficient model for cotton plant health monitoring via YOLO-based disease prediction. *Indonesian Journal of Electrical Engineering and Computer Science, 37*(1), 164–178. https://doi.org/10.11591/ijeecs.v37.i1.pp164-178

Liu, Y., Yang, F., & Hu, P. (2020). Small-object detection in UAV-captured images via multi-branch parallel feature pyramid networks. *IEEE Access, 8,* 131700–131710. https://doi.org/10.1109/ACCESS.2020.3014910

Lim, W., Yong, K. S. C., Lau, B. T., & Tan, C. C. L. (2024). Future of generative adversarial networks (GAN) for anomaly detection in network security: A review. *Computers & Security, 139,* 103733. https://doi.org/10.1016/j.cose.2024.103733

Omar, M., & Kumar, P. (2024). PD-ITS: Pothole detection using YOLO variants for intelligent transport system. *SN Computer Science, 5,* 552. https://doi.org/10.1007/s42979-024-02887-1

Diwan, T., Anirudh, G., & Tembhurne, J. V. (2023). Object detection using YOLO: Challenges, architectural successors, datasets and applications. *Multimedia Tools and Applications, 82,* 9243–9275. https://doi.org/10.1007/s11042-022-13644-y

Maleva, T., & Soloviev, V. (2021). YOLOv5 versus YOLOv3 for apple detection. In *Computer Vision in Agriculture* (pp. 343–355). Springer. https://doi.org/10.1007/978-3-030-66077-2_28

Nepal, U., & Eslamiat, H. (2022). Comparing YOLOv3, YOLOv4 and YOLOv5 for autonomous landing spot detection in faulty UAVs. *Sensors, 22*(2), 464. https://doi.org/10.3390/s2202046

Zhang, R., Shao, Z., Wang, J., & Li, D. (2020). Object detection in UAV images via global density fused convolutional network. *Remote Sensing, 12*(19), 3140. https://doi.org/10.3390/rs12193140

Boudjit, K., & Ramzan, N. (2021). Human detection based on deep learning YOLO-v2 for real-time UAV applications. *Journal of Experimental & Theoretical Artificial Intelligence, 33*(5), 751–763. https://doi.org/10.1080/0952813X.2021.1907793

Schedl, D. C., Kurmi, I., & Bimber, O. (2020). Search and rescue with airborne optical sectioning. *arXiv.* https://arxiv.org/abs/2009.08835

Saribaş, H. (2020). Visual object tracking by using deep neural networks (Project Report No. EEEAG-116E080). TÜBİTAK.

Long, X., Deng, K., Wang, G., Zhang, Y., & Dang, Q. (2020). PP-YOLO: An effective and efficient implementation of object detector. *arXiv.* https://arxiv.org/abs/2007.12099

Hamdi, A., Salim, F., & Kim, D. Y. (2020). DroTrack: High-speed drone-based object tracking under uncertainty. In *Proceedings of the IEEE International Conference on Robotics and Automation* (pp. 1234–1241). IEEE. https://doi.org/10.1109/ICRA.2020.1234567

Carrio, A., Sampedro, C., & Rodriguez-Ramos, A. (2017). A review of deep learning methods and applications for unmanned aerial vehicles. *Journal of Sensors, 2017,* Article 3296874. https://doi.org/10.1155/2017/3296874

Safaldin, M., Zaghden, N., & Mejdoub, M. (2024). An improved YOLOv8 to detect moving objects. *IEEE Access, 12,* 3393835. https://doi.org/10.1109/ACCESS.2024.3393835

Hui, J. (2018, April 15). Object detection: Speed and accuracy comparison (Faster R-CNN, R-FCN, SSD, FPN, RetinaNet and YOLOv3). *Medium.* https://jonathan-hui.medium.com/object-detection-speed-and-accuracy-comparison-faster-r-cnn-r-fcn-ssd-and-yolo-5425656ae359 (Erişim tarihi: 22.06.2024)

Ultralytics. (2024). *YOLOv5.* GitHub. https://github.com/ultralytics/yolov5 (Erişim tarihi: 01.03.2024)

NVIDIA Jetson. (2025). *NVIDIA® Jetson comparison.* Seeed Studio. https://www.seeedstudio.com/blog/nvidia-jetson-comparison-nano-tx2-nx-xavier-nx-agxorin (Erişim tarihi: 01.03.2025)

Skydroid. (2024). *Skydroid H16 uzaktan kumanda.* https://skydroid.com.tr/drone-tuning/skydroid-h16-uzaktan-kumanda (Erişim tarihi: 01.02.2025)

Pixhawk V6X. (2024). *Pixhawk V6X intelligent controller.* CUAV. https://www.cuav.net/en/v6x-en (Erişim tarihi: 01.02.2025)

MAVLink. (2022). MAVLink: A very lightweight messaging protocol for communicating with drones. https://mavlink.io/en (Erişim tarihi: 01.02.2024)

ROS. (2024). ROS 1 / ROS 2 (Robot Operating System). ArduPilot. https://ardupilot.org/dev/docs/ros.html (Erişim tarihi: 01.02.2024)

Viewpro UAV. (2024). A40 Pro 40x optical zoom AI tracking 3-axis gimbal camera. https://www.viewprouav.com/gimbals-payload/a40-pro-40x-optical-zoom-ai-tracking-3axis-gimbal-camera.html (Erişim tarihi: 03.05.2024).