T.C.
İSTANBUL GEDİK ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

ASKERİ ARAÇ TESPİTİNDE YOLO YAPAY ZEKA UYGULAMASI İLE RENK TONU DOYGUNLUĞU DEĞERİ (HSV) YÖNTEMİNİN KARŞILAŞTIRILMASI

YÜKSEK LİSANS TEZİ

Serkan GÜNDÜZ

Yapay Zekâ Mühendisliği Anabilim Dalı

Yapay Zekâ Mühendisliği Tezli Yüksek Lisans Programı

EYLÜL 2023
İSTABUL

T.C.
İSTANBUL GEDİK ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

ASKERİ ARAÇ TESPİTİNDE YOLO YAPAY ZEKA UYGULAMASI İLE RENK TONU DOYGUNLUĞU DEĞERİ (HSV) YÖNTEMİNİN KARŞILAŞTIRILMASI

YÜKSEK LİSANS TEZİ

Serkan GÜNDÜZ
(210039005)
0009-0009-5602-8321

Yapay Zekâ Mühendisliği Anabilim Dalı

Yapay Zekâ Mühendisliği Tezli Yüksek Lisans Programı

Tez Danışmanı: Dr. Öğr. Üyesi Şerife Esra DİNÇER

İstanbul 2023

T.C.
İSTANBUL GEDİK ÜNİVERSİTESİ
Lisansüstü Eğitim Enstitüsü Müdürlüğü

Jüri Tez Onay Formu

07.09.2023

LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ MÜDÜRLÜĞÜ

Bu çalışma 07.09 2023 tarihinde aşağıdaki jüri tarafından Yapay Zeka Mühendisliği Anabilim Dalı, Yapay Zeka Mühendisliği (Tezli Yüksek Lisans) Programı Yüksek Lisans Tezi olarak kabul edilmiştir.

TEZ JÜRİSİ

Dr. Öğr. Üyesi Şerife Esra DİNÇER
Danışman
İstanbul Gedik Üniversitesi

Dr. Öğr. Üyesi Feridun Cemal ÖZÇAKIR
Üye (İmza)
İstanbul Gedik Üniversitesi

Dr. Öğr. Üyesi Ümit ÖZTÜRK
Üye (İmza)
İstanbul Beykent Üniversitesi

YEMİN METNİ

Yüksek Lisans Tezi olarak sunduğum “Askeri Araç Tespitinde Yolo Yapay Zeka Uygulaması İle Renk Tonu Doygunluğu Değeri (Hsv) Yönteminin Karşılaştırılması” başlıklı bu çalışmanın, bilimsel ahlak ve geleneklere uygun şekilde tarafımdan yazıldığını, bu tezdeki bütün bilgileri akademik ve etik kurallar içinde elde ettiğimi, yararlandığım eserlerin tamamının kaynaklarda gösterildiğini ve çalışmamın içinde kullanıldıkları her yerde bunlara atıf yapıldığını, patent ve telif haklarını ihlal edici bir davranışımın olmadığını belirtir ve bunu onurumla doğrularım (07/09/2023).

Serkan GÜNDÜZ

ÖNSÖZ

İnsanoğlunun gökyüzünde kuşlar gibi uçma hayali yüzyıllardır süregelen bir çalışmayı beraberinde getirmiştir. İnsan nasıl uçabilir? Teknolojininde gelişmesi ile beraber insanoğlu pilotlu şekilde uçabilen hava araçlarını geliştirmiştir. İlkel planörler ile başlayan bu macera sıcak hava balonları, zeplinler ve uçaklar olarak ilerlemiştir. Ancak zaman içerisinde daha küçük uçabilen araçlara duyulan ihtiyaçla beraber İnsansız Hava Araçları fikri gelişmeye başlamıştır. İnsanın ağırlığını taşımak zorunda olmadığında çok daha küçük ve hafif ve dolayısıyla az maliyetli İnsansız hava araçları ortaya çıkmaya başlamıştır. Dünya savaşları ile birlikte cephe hatlarında gözlem ve saldırı yapabilecek pilotsuz hava araçlarına duyulan ihtiyaç cevap bulmaya başlar. En değerli kaynak olan insan hayatını tehlikeye atmadan uçabilen ve savaşabilen bu hava araçları ile savaş tarihinin kuralları yeniden yazılmaya başlamıştır. Düşman hatlarının ilerisine geçen bu araçlar için en temel ihtiyaç düşman unsurlarının tespiti ve sivil hedeflerden ayırt edilebilmeleridir. Bunun içinde bilgisayarlı görü sistemleri ve yapay zekanın da katkıları ile görüntü işleme ve hedef tespiti çok daha sağlıklı bir hal alabilir. Bu alanda çalışmalar yapacak araştırmacıların öncelikli görevi İnsansız Hava Araçları ile insanı tehlikeye atmadan sivil ve askeri araçları tespit edebilecek bir teknolojiye en hızlı şekilde ulaşmak olmalıdır.

Eylül 2023 Serkan GÜNDÜZ

İÇİNDEKİLER

Sayfa

ÖNSÖZ ..... **iv**
İÇİNDEKİLER ..... **v**
KISALTMALAR ..... **vii**
ÇİZELGE LİSTESİ ..... **viii**
ŞEKİL LİSTESİ ..... **ix**
ÖZET ..... **xi**
ABSTRACT ..... **xii**
1. GİRİŞ **1**
1.1 Çalışma Konusu 1
1.2 İHA'ların Sınıflandırılması 3
1.2.1 Çok küçük İHA'lar 4
1.2.2 Küçük İHA'lar 4
1.2.3 Orta İHA'lar 5
1.2.4 Büyük İHA'lar 6
1.3 İlgili Çalışmalar 6
1.4 Uçan İHA Kısıtlaması 7
1.5 Çalışmanın Amacı 8
2. METODOLOJİ VE TASARIM **9**
2.1 Görüntü İşleme 9
2.1.1 Python dili 9
2.1.2 OpenCV kitaplığı 10
2.1.3 Morfolojik işlemler 10
2.1.4 Erozyon 11
2.1.5 Genişleme 11
2.1.6 Opening 12
2.1.7 Closing 13
2.1.8 Performans değerlendirme ölçütleri 13
2.1.9 Mini bilgisayar ve kamera modülü 16
2.1.10 Görüntü işlemede kullanılan metot 17
2.1.10.1 YOLOv7 18
2.1.10.2 Bilgisayar görüşünde YOLO nedir? 19
2.1.10.3 Gerçek zamanlı nesne dedektörleri ve YOLO sürümleri 19
2.1.10.4 YOLOv7 nedir 20
2.1.10.5 Temel YOLOv7 sürümleri arasındaki farklar 21
2.1.10.6 YOLOv7 nesne algılama performansı 21
2.1.10.7 YOLOv7 uygulamaları 22
2.1.10.7.1 Güvenlik ve gözetim 22
2.1.10.7.2 Akıllı şehir ve trafik yönetimi 23
2.1.11 Nesne takibi yöntemleri 24
2.2 İHA Kontrolü 25
2.2.1 İHA bileşenleri 26
2.2.1.1 Elektrikli bileşenler 26
2.2.1.1.1 Fırçasız motorlar 26

2.2.1.1.2 A 2212 920KV fırçasız motor 28
2.2.1.1.3 Batarya 28
2.2.1.2 Elektronik bileşenler 29
2.2.1.2.1 Uçuş kontrolörü 29
2.2.1.2.2 Radiolink pixhawk 32 bit uçuş kontrol kartı + güç modülü 30
2.2.1.2.3 GPS ve pusula 31
2.2.1.2.4 Uzaktan kumanda + receiver 31
2.2.1.2.5 Uzaktan kumanda alıcısı 32
2.2.1.2.6 Telemetri alıcı-vericisi 32
2.2.1.2.7 Pil göstergesi 33
2.2.1.2.8 ESC (Elektronik hız kontrolörü) 34
2.2.1.2.9 Güç anahtarı 34
2.2.1.3 Mekanik bileşenler 35
2.2.1.3.1 Çerçeve ve pervaneler 35
2.2.4 Test ve simülasyon 36
2.2.4.1 Test sonucu ile yapılan tasarımın uyumluluğu 36
2.2.5 Güvenlik 36
2.2.5.1 Güvenlik ihtiyaçlarının karşılanması için önlemler ve çözüm yöntemler 36
2.4 Maliyet Analizi 36
3. GEREÇ VE YÖNTEM 38
3.1 Askeri Araç Fotoğraflarının Toplanması 38
3.2 Roboflow Üzerinden Set Eğitimi 40
3.3 Hedef Yakalama Çalışmaları 43
3.4 Pycharm Üzerinde Kullanılabilir Program 45
3.5 HSV(Renk Tonu Doygunluğu Değeri) ile Hedef Yakalama Çalışması 46
3.6 İHA Montajı 51
4. BULGULAR VE KARŞILAŞTIRMA SONUÇLARI 52
4.1 YOLO ve HSV açılımı 57
5. SONUÇ 59
KAYNAKLAR 61
EKLER 65
ÖZGEÇMİŞ 78

KISALTMALAR

CCW : Counterclockwise-saat yönünün tersi

CW : Clockwise-saat yönü

HSV : Renk Tonu Doygunluğu Değeri

İHA : İnsansız Hava Aracı

Li-Po : Lityum polimer

mAh : Miliamper / saat

YOLO : Yalnızca Bir Kez Bakarsınız

ÇİZELGE LİSTESİ

Sayfa

Çizelge 2.1: Karmaşıklık Matrisi 13
Çizelge 2.2: Temel Nesne Dedektörleri YOLOR Ve Yolov4'ün Yeni Yolov7 İle Karşılaştırılması 22
Çizelge 2.3: F450, F550 ve TF680H4 Gövdelerinin Teknik Özellikleri 25
Çizelge 2.4: Telemetre Sinyal Tablosu 33
Çizelge 2.5: İHA Sisteminin Ayrıntılı Maliyet Analizi 37
Çizelge 4.1: Literatürde Nesne Takibi İçin Yaygın Kullanılan Yöntemler 53
Çizelge 4.2: Derin Öğrenmede Kullanılan Kütüphaneler 56
Çizelge 4.3: Yolo ile Hedef Yakalama Performansı 57
Çizelge 4.4: HSV(Renk Tonu Doygunluğu Değeri) İle Hedef Yakalama Performansı 58

ŞEKİL LİSTESİ

Sayfa

Şekil 1.1: Tarihte Belgelenen İlk İHA Olan Güvercin 2
Şekil 1.2: Leonardo da Vinci'nin Hava Vidası Tasarımı 2
Şekil 1.3: Predatör İHA 3
Şekil 1.4: Aksungur İHA 3
Şekil 1.5: Çok Küçük İHA Örneği 4
Şekil 1.6: Küçük İHA Örneği "HAVELSAN BAHA İHA" 5
Şekil 1.7: Orta İHA Örneği "Eagle Eye" 5
Şekil 1.8: Büyük İHA Örneği "MQ-9 Reaper" 6
Şekil 2.1: Sistem Çalışma Prensibinin Blok Diyagramı 9
Şekil 2.2: Erozyon 11
Şekil 2.3: Genişleme 12
Şekil 2.4: Opening 12
Şekil 2.5: Closing 13
Şekil 2.6: Algoritmanın Akış Şeması 14
Şekil 2.7: Mini PC'nin Fotoğrafı (Raspberry Pi 3 Model B+) 16
Şekil 2.8: Mini PC'nin Kamera Kartı 16
Şekil 2.9: YOLOv7 Algoritması 18
Şekil 2.10: Halka Açık Yerlerde Kalabalık Tespiti İçin Görüntü İşleme 19
Şekil 2.11: Yolov7, Viso Suite Üzerine İnşa Edilmiş, İnşaatta Bir Bilgisayarla Görme Uygulaması 20
Şekil 2.12: Uçak Tespiti İçin YOLO V7 Kullanan Bilgisayar Görüş Sistemi 21
Şekil 2.13: Diğer Gerçek Zamanlı Nesne Dedektörleriyle Karşılaştırma 21
Şekil 2.14: Derin Öğrenme İle Yüz Algılama 23
Şekil 2.15: YOLO Modellerini Kullanarak Perakende Mağazalarında Kişi Tespiti . 23
Şekil 2.16: Otomotiv İmalatında Nesne Tespiti İçin Bilgisayar Görüşü 23
Şekil 2.17: Kullanılan 2212 920KV Fırçasız Dc Motor Modeli 27
Şekil 2.18: Pervane Dönüş Yönleri 27

Şekil 2.19: Kullanılan 5300 mAh 3S Li-po Pil 28
Şekil 2.20: Pixhawk Devre Şeması Temsili 29
Şekil 2.21: Pixhawk Uçuş Kontrolörü Üstten Görünüm ve Bağlantı Noktalarının Görünümü..... 29
Şekil 2.22: Sistem Devre Şeması..... 30
Şekil 2.23: GPS Modülü..... 31
Şekil 2.24: 2.4 GHz'lik 6 Kanallı Kumanda Verici ve Alıcı..... 31
Şekil 2.25: Uzaktan kumanda alıcısı 32
Şekil 2.26: Kanal Bağlantıları 32
Şekil 2.27: Telemetri Modülü Alıcı-Vericileri..... 32
Şekil 2.28: Batarya Voltaj Seviyesi Göstergesi..... 34
Şekil 2.29: Kullanılan 30A'lik ESC Modeli..... 34
Şekil 2.30: Güç Düğmesi..... 34
Şekil 2.31: Montaj Aşamaları..... 35
Şekil 2.32: 1045 İHA Pervanesi Seti Cw/ccw – Siyah..... 35
Şekil 3.1: Uydudan Askeri Araç Görüntüleri 39
Şekil 3.2: Uydudan Askeri Araç Görüntüleri 40
Şekil 3.3: İşlenen Görüntü Örneği..... 41
Şekil 3.4: İşlenen Görüntü Örneği..... 41
Şekil 3.5: İşlenen Görüntüler ile Train-Test-Valid için Oluşturulan Dosya..... 42
Şekil 3.6: Çalışmada Kullanılan Askeri Araç Fotoları 42
Şekil 3.7: Kişisel Roboflow Sayfası 43
Şekil 3.8: İşlenen Görüntüler İle Yakalanan Askeri Araçlar..... 43
Şekil 3.9: Roboflow Ana Sayfası Kişisel Bilgisayar Ekran Görüntüsü 44
Şekil 3.10: YOLO Versiyonları ile Tespit Örneği..... 46
Şekil 3.11: HSV(Renk Tonu Doygunluğu Değeri) ile Tespit Kodları 50
Şekil 3.12: HSV (Renk Tonu Doygunluğu Değeri) ile Tespit Örneği 50
Şekil 3.13: HSV (Renk Tonu Doygunluğu Değeri) ile Tespit Örneği 50
Şekil 3.14: İHA Montaj Aşamaları..... 51
Şekil 4.1: Performans karşılaştırması YOLOv7 - YOLOR - YOLOX - YOLOv5 ... 54
Şekil 4.2: HSV(Renk Tonu Doygunluğu Değeri) Uzaydaki Renk Kodları Aralığı 55
Şekil 4.3: Discriminative Deep Appearance Modelinin Genel Ağ Yapısı 56

ASKERİ ARAÇ TESPİTİNDE YOLO YAPAY ZEKA UYGULAMASI İLE RENK TONU DOYGUNLUĞU DEĞERİ (HSV) YÖNTEMİNİN KARŞILAŞTIRILMASI

ÖZET

Günümüzün yükselen trentlerinden yapay zekanın yanısıra tüm sektörlerde olduğu gibi askeri alanda da İHA kullanımı hızla artmaktadır. Ayrıca yapay zeka ve insansız hava araçları, orman yangınlarının başlangıç aşamasında tespiti, sınır kaçakçılığı, düzensiz göçmenlerin tespiti içinde sıkça tercih edilmeye başlanmıştır.

Bilinirliğine paralel olarak yapay zekâ teknolojisi gelişmekte olan bir alandır. Birçok alanda yaygın olarak kullanılmaktadır. İnsan hatalarını en aza indirdiği ve daha uygun maliyetli olduğu için bu teknolojiye olan talep günden güne artmaktadır. Ortaya çıkan ürünün artan talebi ile ilgili bilimsel çalışmalar olmasına rağmen, arama yapıldığında tespit ve takip sistemleri ile ilgili çok fazla çalışma bulunmamaktadır.

Bu tezin önceki çalışmalardan ayrılan en önemli özelliği hedefin tanımlanması sırasında veri setlerinden yararlanılmasıdır. Önceki çalışmalarda araç takibi yapılacaksa hedef boy ve en ölçüleri ayarlanarak bir dikdörtgen olarak tanımlanmış ve takibi bu şeklinde yapılmıştır. Bu çalışmalarda araç tespiti mümkün ancak sivil ve askeri araç ayırımı mümkün değildir. Tez çalışmasında ise olası hedefin İHA tarafından imha edilecek olması sebebiyle sivil araç ile düşman askeri araçların birbirinden ayrılması gerekmektedir. Çok yukarılardan bakıldığında bile askeri araçların seçilebileceği eğitilmiş bir algoritma ile hedef tespiti planlanmaktadır. Çalışma sonunda ikinci kullanıcaya sunulabilecek ticari bir ürün olarak askeri araç tespit edebilen "Ağırlık Dosyalarının" üretimi hedeflenmektedir.

Hedeflerin tespiti ve sınıflandırmasında hazır eğitilmiş veri setleri üzerinden çalışan Yolov7 ve Yolov8 Algoritması tercih edilmiştir. Hız ve kesinlik açısından gelenekselleşmiş görü sistemlerine göre yüksek performans ve verim elde edilebilen bu yöntem ile hedef tanıma ve tespiti sağlanmaya çalışılmıştır.

Bu tezin amacı, bu araştırma alanına katkıda bulunmak için, istenilen askeri hedefi tespit edebilen etkili bir insansız hava aracı üretmenin yanısıra hedefi tespit edebilecek en etkili görüntü işleme yöntemini bulabilmek için HSV (Renk Doygunluğu Değeri) ile yolo'nun karşılaştırılması amaçlanmıştır .

Tez sonunda görüntü işleme donanımlarına sahip bir insansız hava aracı ve özgün bir ağırlık dosyası üretılmesi planlanmaktadır.

Anahtar Kelimeler: *İnsansız hava araçları, askeri araç tespiti, hedef tespiti, RoboFlow, Yolo, HSV*

COMPARISON OF YOLO AI APPLICATION AND COLOR SATURATION VALUE (HSV) METHOD IN MILITARY VEHICLE DETECTION

ABSTRACT

In addition to artificial intelligence, which is one of the rising trends of today, the use of UAVs in the military field is increasing rapidly as in all sectors. In addition, artificial intelligence and unmanned aerial vehicles have started to be preferred frequently for the detection of forest fires at the initial stage, border smuggling, and detection of irregular migrants.

In parallel with its popularity, artificial intelligence technology is an emerging field. It is widely used in many fields. The demand for this technology is increasing day by day as it minimizes human errors and is more cost effective. Although there are scientific studies on the increasing demand for the product, there are not many studies on detection and tracking systems when searching.

The most important feature of this thesis that distinguishes it from previous studies is the use of data sets during the definition of the target. In previous studies, if the vehicle was to be tracked, the target was defined as a rectangle by adjusting the height and width dimensions, and it was followed in this way. Vehicle detection is possible in these studies, but it is not possible to distinguish between civil and military vehicles. In the thesis study, it is necessary to separate the civilian vehicle from the enemy military vehicle, since the possible target will be destroyed by the UAV. Target detection is planned with a trained algorithm where military vehicles can be selected even when viewed from very high. At the end of the study, it is aimed to produce "Weight Files" that can detect military vehicles as a commercial product that can be offered to the second user.

In the detection and classification of targets, Yolov7 and Yolov8 Algorithms working on ready-trained data sets were preferred. With this method, which can achieve high performance and efficiency compared to traditional vision systems in terms of speed and precision, target recognition and detection have been tried to be achieved.

The aim of this thesis, in order to contribute to this field of research, is to produce an effective unmanned aerial vehicle that can detect the desired military target, as well as to compare HSV (Color Saturation Value) and yolo in order to find the most effective image processing method to detect the target.At the end of the thesis, it is planned to produce an unmanned aerial vehicle with image processing equipment and an original weight file.

Keywords: *Unmanned aerial vehicles, military vehicle detection, target detection, RoboFlow, Yolo, HSV*

1. GİRİŞ

1.1 Çalışma Konusu

İnsansız Hava Araçları ile yapılan çalışmalar uzun yıllar öncesine dayanmaktadır. Çok farklı tanımlamaların olduğu kaynaklarda genel olarak pilotsuz kullanım ön plana çıkmaktadır. İnsansız hava araçları kısaca İHA'lar aracın içinde bir insan pilot veya yolcu olmadan uçabilir kabiliyete sahiptir. Kontrol fonksiyonları ise araç içerisinde yerleşik veya uzaktan kumandalı olabilir, (Valavanis, 2007). Sektör ve teknoloji temelli kaynaklar takip edildiğinde çok değişik görünümlerde insansız hava araçlarına raslamak mümkündür. Sinema sektörü ise hayal gücünü kullanarak insan ve yük taşıyabilen İHA'ları ekrana taşımakta, bu yol ilede ideal işleve ve görünüme sahip araçlar konusunda gerçek dünyadaki sektör temsilcilerine yol göstermektedir.

İHA sektörünün genişlemesine paralel olarak, yapay zekâ teknolojiside son yıllarda otonom sistemler ile beraber gelişmeye başlamıştır. Otonom sistemler her alana yayılmıştır ve bu eğilim hızla artmaktadır.

Hükümetler tarafından yakından takip edilen savunma sanayi, arama kurtarma ve izleme çalışmaların temel alanlarını oluşturmaktadır.

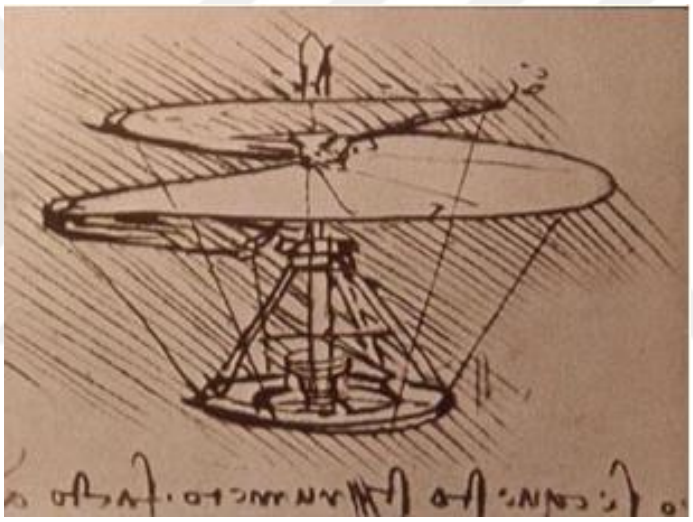
İnsansız Hava Araçları tarım veya endüstri ile ilgili alanların taranması ve haritalama, fotoğrafçılık veya hobi amaçlı olarak yaygın şekilde kullanılmaktadır. Otonom sistemler alanındaki gelişmeler önemli oranda iş gücünün azalması, hata olasılığının azalması ve olası kaza risklerini en aza indirilmesi olarak kendini göstermektedir (Keipour ve diğ., 2019).

Bu aşamada İnsansız hava Araçlarının tarihsel gelişimine köşe taşı olabilecek birkaç gelişmeden bahsetmekte fayda olacaktır. İlk uçan makine tasarımı yaklaşık 2500 yıl önce antik çağda Yunanistan ve Çin'de planlandı (Valavanis, 2007). 425 yılında güvercin denilen mekanik bir kuş yapılmıştır. Antik Yunan matematikçi Archytas'ın Uçan güvercin tasviri resimde gösterilmiştir.

Şekil 1.1: Tarihte Belgelenen İlk İHA Olan Güvercin

Kaynak: (Valavanis ve Vachtsevanos, 2015)

1483'te Leonardo Da Vinci, hava vidası adı verilen bir uçak tasarlamıştır.



Şekil 1.2: Leonardo da Vinci'nin Hava Vidası Tasarımı

Kaynak: (Valavanis ve Vachtsevanos, 2015)

22 Ağustos 1849'da İHA ilk kez askeri amaçla kullanılmıştır. 1883'te Douglas Archibald anemometreyi uçurtmaya entegre ederek ve rüzgârın hızını ölçmeyi başarmıştır. Kameralar ilk defa 1887'de Bay Archibald tarafından uçurtmalara takılır, bu nedenle dünyanın ilk keşif İHA'ları kabul edilmektedir (Fahlstrom ve Gleason, 2012). Elmer Sperry, Jiroskop kullanması otonom kontrolünün başlangıcı olarak bilinir ki bu bir uçağın otomatik olarak yönlendirilmesi için kullanılır (Nonami ve diğ., 2010). Bu insansız hava aracını kontrol etmek için radyo kontrol teknikleri kullanılılır.

İnsansız hava araçları için Queen Bee (Kraliçe Arı) adının 'drone' (erkek arı) teriminin kullanılmasına yol açtığı söylenir. Bu ilk dronlar aynı zamanda Hedef

uçağı olaraktan kullanılmışlardır. 1950'lerin ortalarında, SD-1 olarak adlandırılan veya bilinen adıyla ilk keşif uçağı MQM-57 Falconer geliştirilmiştir. Günümüzde ülkemizde milli ve yerli sermaye ve teknoloji kullanılarak geliştirilen Aksungurun fotoğrafı Şekil 1.4'te verilmiştir.

Şekil 1.3: Predatör İHA

Kaynak: (Howstuffworks, 2019)

Şekil 1.4: Aksungur İHA

Kaynak: (defensnews, 2011)

1.2 İHA'ların Sınıflandırılması

Literatürde birçok sınıflandırma yöntemi vardır. Bu çalışma için sınıflandırma, ölçülere göre daha kullanışlıdır.

1.2.1 Çok küçük İHA'lar

Çok küçük İHA'ların genişlikleri 50 cm uzunluğa kadar gider. İHA'ların çoğu bu sınıfta sayılabilir.

Şekil 1.5: Çok Küçük İHA Örneği

Kaynak: (defensnews, 2011)

Daha çok hobi amaçlı ve sportif kullanıma uygun çok küçük İHA lar Sivil Havacılık Genel Müdürlüğünün meskun mahallerde uçuş kriterlerine uygun büyüklük ve ağırlığa sahiptirler. Otonom uçuş, araç takip ve havalandığı bölgeye geri dönme gibi gelişmiş pek çok özelliğe sahip olmalarının yanısıra ekonomik fiyatlara sahip olmaları ilede amatör ve yarı profesyonel kullanıcıların temin edebilmeleri açısından tercih edilen cihazlardır. Dar alan uçuşlarına uygun boyutları ile kapalı spor salonları gibi etkinlik alanlarında video çekim kabiliyetlerine sahiptirler.

1.2.2 Küçük İHA'lar

50 cm'den büyük, Bayraktar, HAVELSAN BAHA ve Raven B bunlara örnek olabilir. Daha çok havadan gözlem amacıyla kullanılmaya uygun yapısal özelliklere sahip olan küçük İHA'lar ile trafik durum gözlemi, orman yangınlarının önlenmesi için denetim ve gözlem, film ve belgesel çekimleri, savaşlarda kamikaze saldırılar, sınır hatlarının güvenliği, tren ve boru hatlarının gözlem ve denetimi, kaçak avlanmaların önlenmesi ve denetimi amaçları ile kullanılmaya uygundur.

Şekil 1.6: Küçük İHA Örneği "HAVELSAN BAHA İHA"

Kaynak: (defensnews, 2011)

1.2.3 Orta İHA'lar

İHA tek kişi tarafından taşınabilecek büyüklükte, hafif hava araçlarına karşı ise küçük ölçüdedir. Bu İHA'ların kanat açıklıkları yaklaşık 5–10 metre ve İHA'lar 200 kg civarında faydalı yük taşıyabilir. Orta İHA'lar çoğunlukla askeri amaçlı olarak kullanılmaktadır. Bayraktar TB2 bu sınıfın ülkemizde ençok tanınan modelidir. Yurt dışında gerek kabiliyetleri gerekse nuadillerine göre ekonomik avantajları sebebiyle tercih edilen askeri sınıfta bir SİHA dır.

Şekil 1.7: Orta İHA Örneği "Eagle Eye"

Kaynak: (defensnews, 2011)

1.2.4 Büyük İHA'lar

Yukarıdaki kategorilerden daha büyük olan diğer tüm İHA'lara Büyük İHA denir.

Şekil 1.8: Büyük İHA Örneği "MQ-9 Reaper"

Kaynak: (defensnews, 2011)

Tamamen askeri amaçlar için kullanılan bu sınıftaki askeri amaçlar çok yüksek düzeyde faydalı yük taşıma kapasitesine sahiptirler. Özellikle kanat uzunluklarının uygun olması sebebiyle kanat altlarına monte edilebilen MAM tipi askeri teczizatlar ile savaşlarda aktif rol alabilmektedir.

1.3 İlgili Çalışmalar

Literatür taramasında otonomi ve İHA sistemleri konusuyla ilgili birçok projenin hayata geçirilmiş olduğu görülmektedir.

2003 'de Motorlu Araç Plaka Görüntülerinden Karakter Ayrıştırma ve Tanıma konusunda çalışmalar yapılmıştır (Oral ve diğ., 2003). Bu çalışmada Motorlu taşıt plakalarından karakter ayrıştırma ve tanıma işlemleri için bir dizi görüntü işleme algoritmasının yanı sıra geriye yayılım algoritması kullanan yapay Sinir Ağları (YSA)'ndan yararlanılmıştır.

Bu çalışmada sistem performansını belirlemek için üç parametreye; karakter ayrıştırabilme yeteneği, karakter tanıma başarısı ve plaka tanıma kabiliyetini ölçmeye yönelik testler yapılmıştır. Sistemin karakter ayrıştırma başarısı %99.7 olarak belirlenmiştir. Karakter tanıma başarısı da %96.7 değerine sahiptir ve oldukça yüksektir. YSA'larının daha zengin bir eğitim seti ile eğitilmesi ile bu başarı oranı

daha da yukarılara çekilebilir. Karakter tanıma becerisinin arttırılması %86.6 olan plaka tanıma başarısını doğrudan etkileyecek ve sistemin pratik olarak kullanılabilirliğini arttıracağı öngörülmüştür ve 2019 da (Kumar ve diğ., 2019) derin öğrenme kullanarak otonomi teknolojisi üzerinde çalışmalar yapmıştır.

2019 yılında yapılan bir akademik çalışmada 4 motorlu bir insansız hava aracının ve kara aracının üretim aşamaları inşa edilmiştir (Çiftçi ve diğ., 2019) . Bu çalışmada dikey iniş ve kalkış yapabilen uzaktan kumanda ile yönlendirilebilen bir insansız hava aracının çalışma prensipleri işlenmiştir. Bu çalışmada İnsansız hava aracı (İHA) hava akımı ve itici kuvvetlerinden yararlanarak uçabilen yerden kumanda edilen (RC-Radio Controlled veya Remote Controlled, uzaktan kumandalı) yâda otonom yani belli bir uçuş planı üzerinden otomatik hareket eden, uçuş için içerisinde bir pilota ihtiyaç duymayan hava aracı olarak tanımlanmıştır. Bu çalışmada yer istasyonundan havalanan insansız hava aracı (projede sözü geçen araç quadcopterdır) ilgili bölgeye giderek, havadan görüntü alarak haritalama yapar. Elde ettiği veriler ışığında en uygun rotayı hesaplanmaya çalışılır. İHA'lar için en uygun rota minimum sürede labirenti tamamlayacak yol bilgisidir. Hesaplanan rota insansız kara aracına gönderilir. Araç elindeki veriler ışığında otonom bir şekilde alandan çıkmaya çalışır.

2019 yılında Otonom çalışan faydalı bir proje hayata geçirilmiştir (Lin ve diğ., 2019), ancak bu ürün ağır, pahalı ve uçuş süresi kısa olması sebebiyle tercih edilmemiştir.

Hedef gereksinimler, yeni hedefe uyum sağlamak için kolayca değiştirilebilir. Tarama alanı, tarama rotası, tarama mesafesi adımları ve tarama hızı ayarlanabilir. Sistemin derin öğrenme için çalışmaya ve zamana ihtiyacı yoktur. Bu modellemede olası kaza sonrası mekanik aksamın hasar görmesi durumunda plastik malzemeler 3D yazıcıdan alınarak kolayca geri kazanılabilir.

1.4 Uçan İHA Kısıtlaması

Valilik, jandarma gibi kurumlardan izin alınması gerekmektedir. Prosedürler uygulandıktan sonra bu proje kapsamında yapılan uçuşlar için ilgili yerlerden ve sivil havacılık genel müdürlüğünden izinler alınmalıdır. Uçuş güvenliğinin sağlanabilmesi için insan ve araç yoğunluğunun en az olduğu, askeri veya özel resmi güvenlik

bölgelerinin dışında kalan alanlar için resmi uçuş izinleri ülkelerin ilgili mevzuatları çerçevesinde alınabilmektedir.

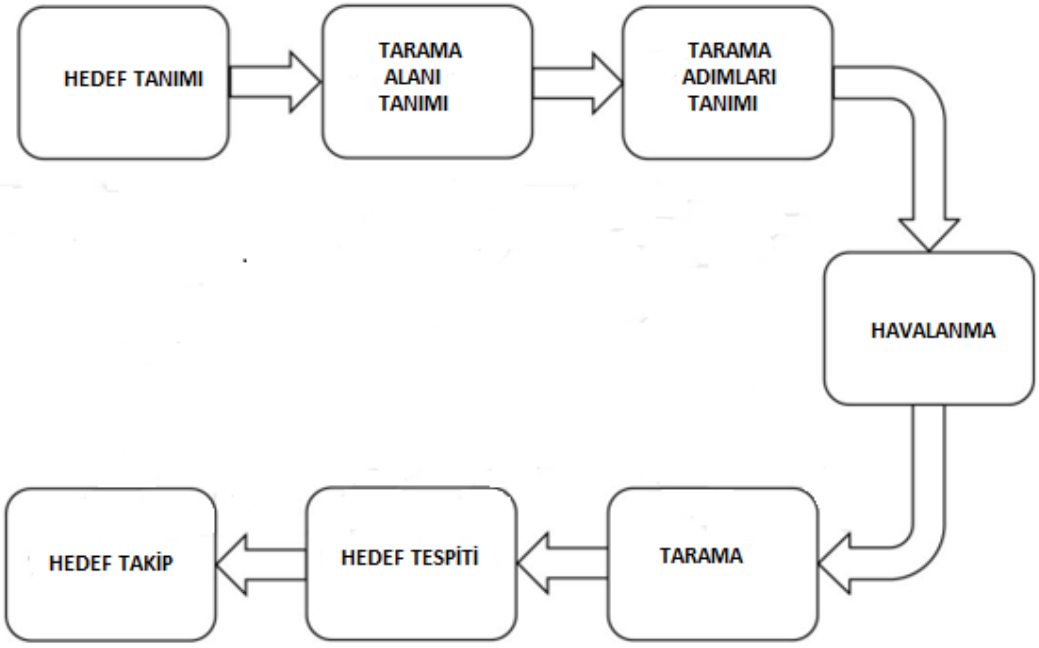
1.5 Çalışmanın Amacı

Bu tez ile hedeflenen görüntü tespiti ve görüntü yakalama yapabilen İHA'lar için ideal görüntü işleme yöntemini bulmaktır. Çalışmaların ileyleyen safhalarında işlenmiş görüntüler ile hedeflenen askeri araçları tespit edebilecek ticari bir ürüne dönüşme potansiyeline sahip özgün bir ağırlık dosyası oluşturulmaya çalışılacaktır. Tüm parçaları birleştirerek arama ve hedef tespiti yapacak belirlenen alanda belirlenen hedefi takip edecek algoritmaya sahip insansız hava aracı geliştirmeye katkıda bulunmak amaçlanmıştır.

Tezin ideal görüntü yakalama yönteminin tespiti ve İHAnın kontrolü olmak üzere iki ana konusu çalışıldı.

2. METODOLOJİ VE TASARIM

Sistemin blok diyagramı aşağıda verilmiştir. Şekil 2.1.



Şekil 2.1: Sistem Çalışma Prensibinin Blok Diyagramı

2.1 Görüntü İşleme

Görüntü işleme işlemi için bu çalışmada mini PC kullanılmıştır. Python dili bu şekilde işlemler için çok popülerdir. Görüntü işleme teknolojisi için en uygun yazılım dili olması dolayısıyla Python dili çalışma için seçilmiştir. Ayrıca, OpenCV (Açık Kaynaklı Bilgisayarla Görüntü işleme alanında yaygın olarak kullanılan Library) kütüphanesinden de yararlanılmıştır (Kumar ve diğ., 2019)

2.1.1 Python dili

Python, yorumlanmış bir üst düzey programlama dilidir, bu nedenle Python programları uzun derleme işlemi olmadan hemen çalışabilir.

Diğer programlama dillerine kıyasla yaklaşık %80 oranında kod uzunluğu azdır. Çoğu Python programları herhangi bir değişiklik yapılmadan birçok bilgisayar platformunda çalıştırılabilir. Linux ve Windows arasındaki taşımalarda Python kodu

çok kolaylık sağlamaktadır. Python'un önceden oluşturulmuş ve taşınabilir işlevleri olan geniş bir kitaplığa sahiptir.

2.1.2 OpenCV kitaplığı

OpenCV, programlama dilleri için bir kütüphane görevi türüdür, Bradski ve Kaehler (2008). Açık kaynak kodludur ve görüntü işleme alanında yaygın olarak kullanılır. OpenCV bilgisayar görme altyapısı için kullanım kolaylığı sağlar.

OpenCV (Açık Kaynak Bilgisayarlı Görme Kütüphanesi), açık kaynaklı bir bilgisayarlı görme ve makine öğrenimi yazılım kütüphanesidir. OpenCV, bilgisayarlı görü uygulamalarına ortak bir altyapı sağlamak ve makine algısının ticari ürünlerde kullanımını hızlandırmak amacıyla oluşturulmuştur. (opencv.org,2023)

Kütüphanede, hem klasik hem de son teknoloji ürünü bilgisayar görüşü ve makine öğrenimi algoritmalar bulunmaktadır. Bu algoritmalar, yüzleri algılamak ve tanımak, nesneleri tanımlamak, videolardaki insan eylemlerini sınıflandırmak, kamera hareketlerini izlemek, hareketli nesneleri izlemek, nesnelerin 3 boyutlu modellerini çıkarmak, stereo kameralardan 3 boyutlu nokta bulutları oluşturmak, yüksek çözünürlük elde etmek için görüntüleri birleştirmek için kullanılabilir. tüm bir sahnenin görüntüsünü alma, görüntü veri tabanından benzer görüntüleri bulma, flaş kullanılarak çekilen görüntülerdeki kırmızı gözleri kaldırma, göz hareketlerini takip etme, manzarayı tanıma ve artırılmış gerçeklikle kaplamak için işaretçiler oluşturma vb. işlemler için, kütüphane şirketler, araştırma grupları ve devlet kurumları tarafından yaygın olarak kullanılmaktadır (opencv.org,2023).

Kütüphaneyi kullanan Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota gibi köklü şirketlerin yanı sıra OpenCV'yi yaygın olarak kullanan Applied Minds, VideoSurf ve Zeitera gibi birçok startup mevcuttur, (opencv.org,2023).

2.1.3 Morfolojik işlemler

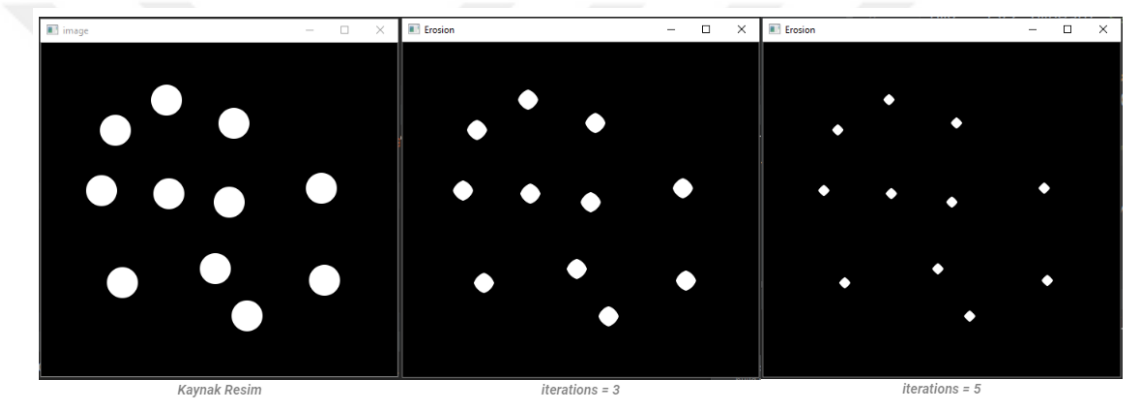
Görüntüler üzerinde morfolojik (şekil bilimsel) işlemler yaparken kullanacak bazı kavramlar : Erozyon, Dilation, Opening,.. olarak sıralanabilir.

Bu özellikleri kullanabilmek için cv2.erode(), cv2.dilate(), cv2.morphologyEx() gibi belli başlı kod metinlerine ihtiyaç olacaktır.

Morfolojik işlemler, resimlerin piksel ve kanal verileri üzerindeki bir takım temel işlemleri ifade etmektedir. Bu işlemleri yaparken binary resimler üzerinde çalışılmıştır. Tüm morfolojik işlemlerde yaptığımız temel işlem girdi olarak alınan resmi, bir Kernel (çekirdek) ile işlemektir. Erozyon, dilation, opening, closing gibi birçok morfolojik işlem vardır.

2.1.4 Erozyon

Erozyon, aşınmak ve aşındırmak gibi anlamlara gelir. Resim üzerinde ön plandaki şekli aşındırmak için kullanılan yöntemdir. Bilindiği gibi toprak erozyonundan farklı değildir. Bu yöntem bir resme uyguladığında ön plandaki şekil aşınır ve zayıflar.



Şekil 2.2: Erozyon

Kaynak: (Güray, 2022: s5)

Erozyon genellikle küçük gürültüleri gidermek için kullanılır, (Güray,2022)

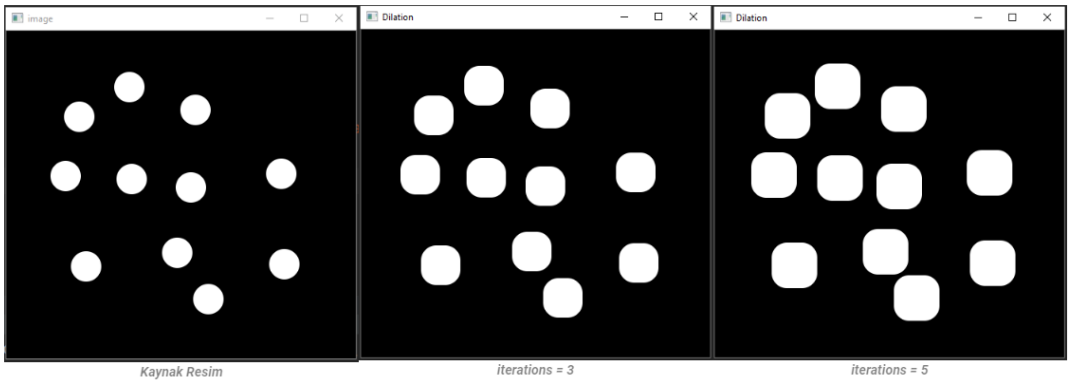
Aşındırma işlevi, belirtilen çekirdek ile görüntü üzerinde tarayarak orijinal görüntüden yeni bir görüntü oluşturur, Bradski ve Kaehler (2008).

Orijinal görüntüde, yalnızca çekirdeğin altındaki tüm pikseller 1 ise 1 olarak ayarlanacaktır, aksi takdirde ilgili piksel 0 olarak ayarlanacak ve aşınacaktır. Bu işlev, iki bağlı nesneyi ayırmak için de kullanılabilir, (GeeksforGeeks, 2019).

2.1.5 Genişleme

Sözcük anlamı genişleme, genleşmedir. Erozyon işleminin tam tersidir denilebilir. Genişletme işlemi genellikle nesnenin sınırını genişletmek için kullanılır. Genişletme işlevi, görüntü üzerinde belirtilen çekirdek ile tarama yaparak orijinal

görüntüden yeni bir görüntü oluşturur, Bradski ve Kaehler (2008). Genişletme işleminde çekirdeğin altında "1" pikseli yoksa piksel '0' olarak ayarlanır.



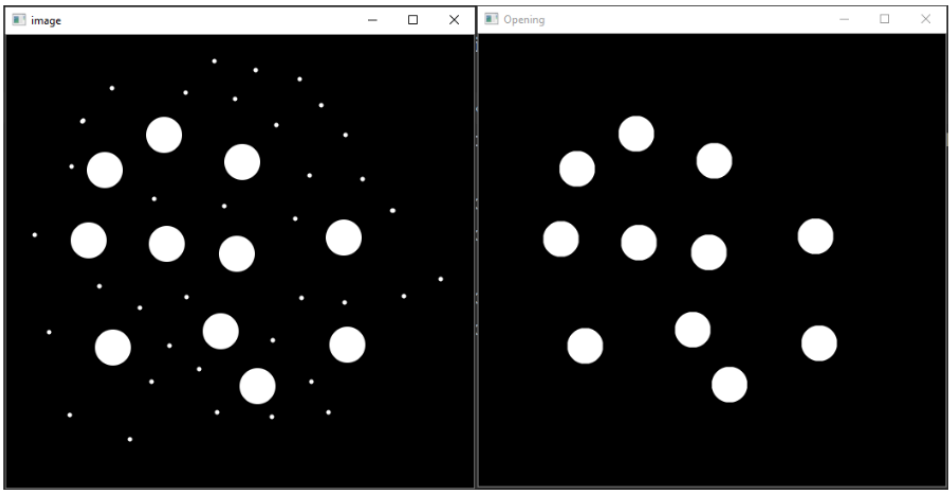
Şekil 2.3: Genişleme

Kaynak: (Kumar, 2018:s25)

Filtrelemenin amacı nesne küçülürken aşındırma fonksiyonu ile gürültüleri de ortadan kaldırmaktır. Kısaca, dilatasyon iç bükeylikleri ve erozyonu düzeltir çıkıntıları yumuşatır, Bradski ve Kaehler (2008).

2.1.6 Opening

Resim üzerindeki gürültüleri kaldırmak için faydalı bir yöntemdir. Algoritma öncekilerle aynıdır. Kütüphane ve resim projeye yükledikten sonra aşağıdaki sonuç elde edilir.

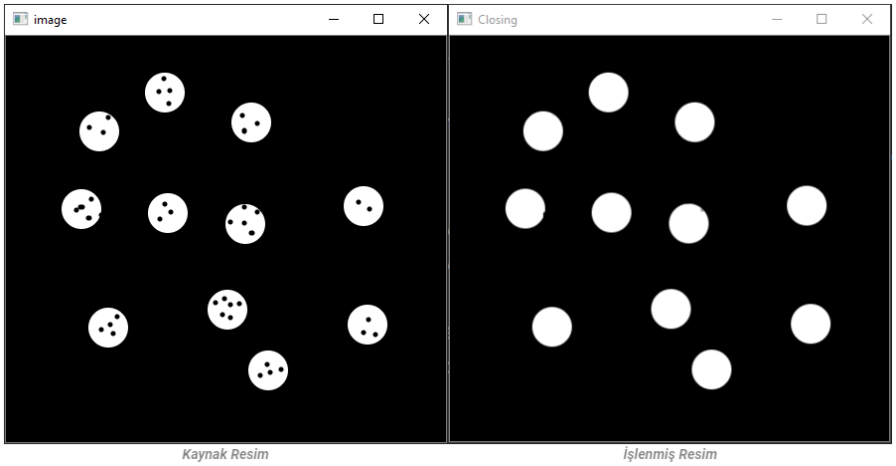


Şekil 2.4: Opening

Kaynak: (Güray,2022:s5)

2.1.7 Closing

Teknik olarak opening işleminin tam tersidir. Bu yöntemle nesnelerin dışındaki değil içindeki gürültüler temizlenir.



Şekil 2.5: Closing

Kaynak: (Güray, 2022: s5)

Öncelikle kütüphane ve resim projeye yüklenir, ayrıca bir de kernele ihtiyaç olacaktır, ardından ilgili kernel dizisi resme uygulanıp çıktılara bakılabilir.

2.1.8 Performans değerlendirme ölçütleri

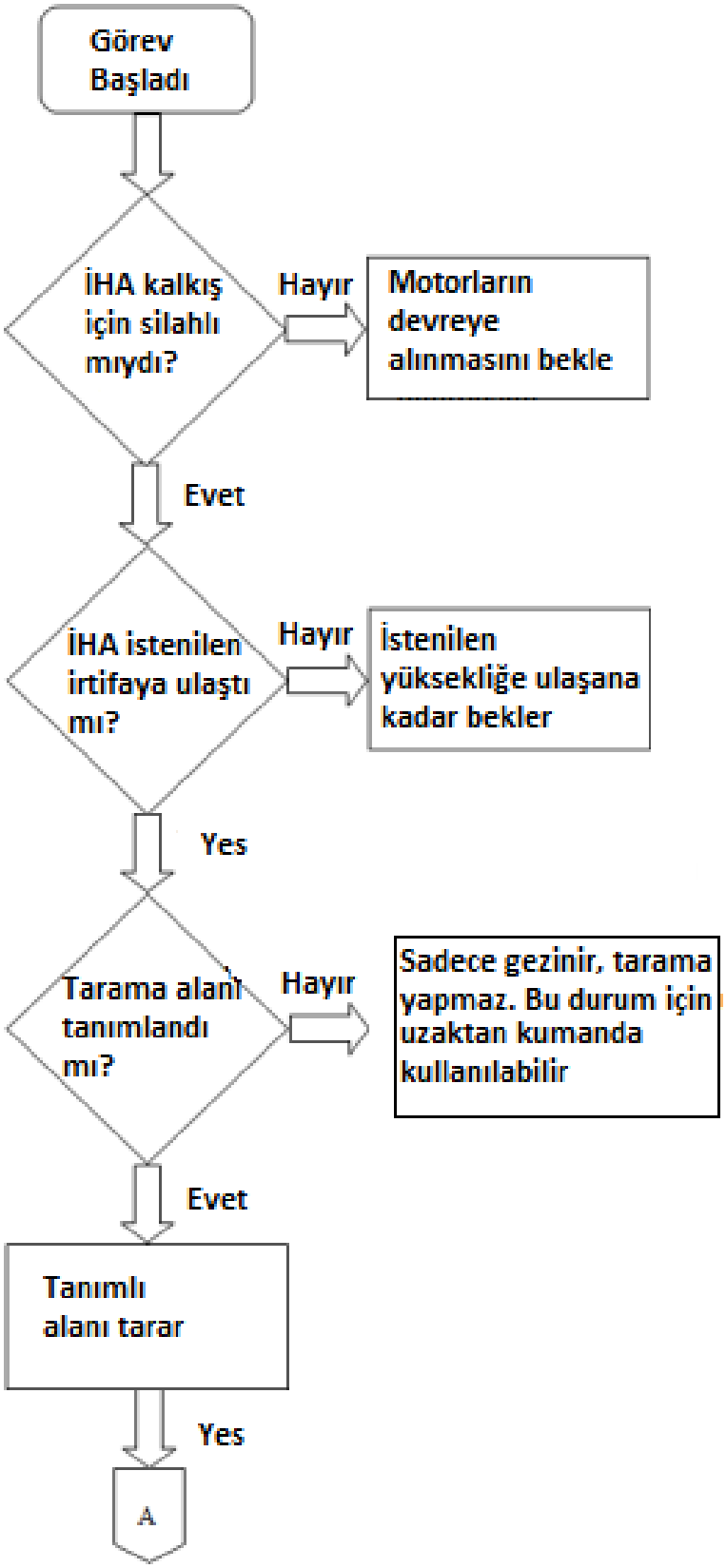
Çalışmada modelin performansı karmaşıklık matrisi (confusion matrix) kullanılarak değerlendirilmiştir. Çizelge 2.1'de gösterilen iki boyuta sahip karmaşıklık matrisi değerleri hesaplanmıştır.

Çizelge 2.1: Karmaşıklık Matrisi

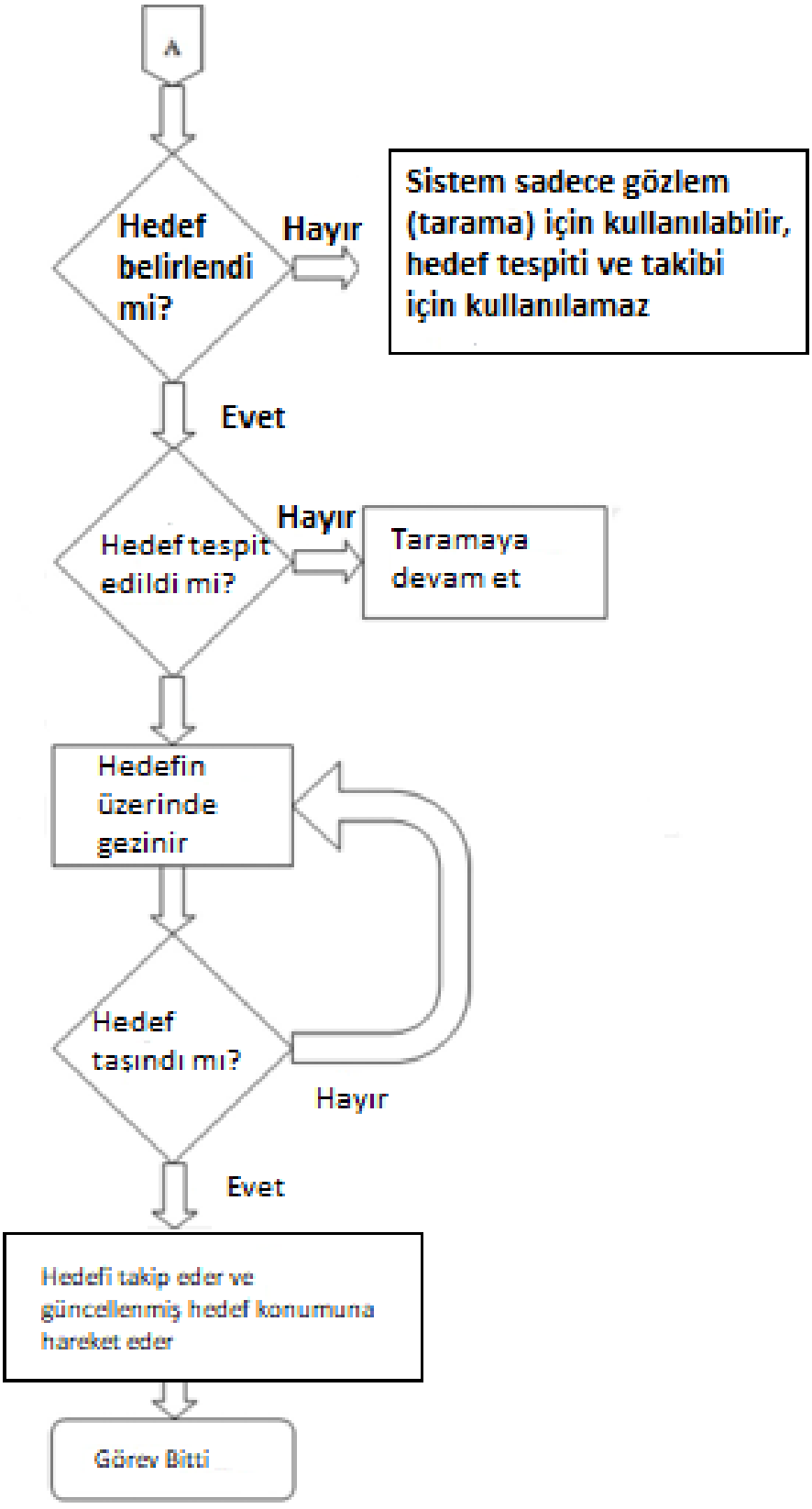
		TAHMİN EDİLEN DEĞER		
GERÇEK DEĞER	POZİTİF	DOĞRU POZİTİF (True positive-TP)	YANLIŞ POZİTİF (Falsepositive-FP)	POZİTİF
	NEGATİF	YANLIŞ NEGATİF (Falsenegative-FN)	DOĞRU NEGATİF (True negative-TN)	NEGATİF

Kaynak: (Tiwari, M. ve Singhai, R. 2017: s.48)

Farklı hedefi olan her şey parametreler ile kolayca değiştirilebilir ve sistem başka bir göreve uyarlanabilir.



Şekil 2.6: Algoritmanın Akış Şeması



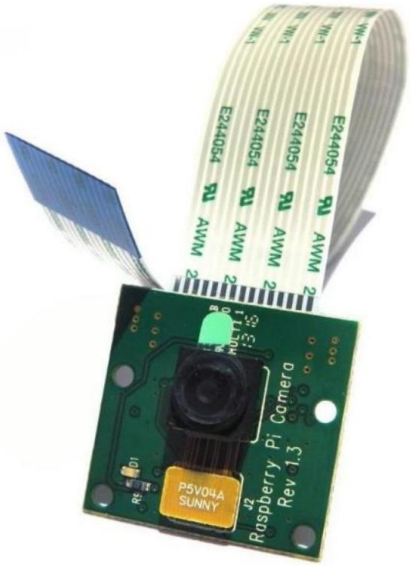
Şekil 2.6: (Devam) Algoritmanın Akış Şeması
Kaynak: (Kumar, 2018:s25)

2.1.9 Mini bilgisayar ve kamera modülü

Görüntü işleme ve komut için üretim aşamalarında Raspberry Pi 3 Model B+ kullanılabilir. Bilgisayar çok küçük boyutlu, hafif ve düşük maliyetli olmasına rağmen iyi performansa sahiptir. Mini PC'nin fotoğrafı Şekil 2.4'te verilmiştir.

Şekil 2.7: Mini PC'nin Fotoğrafı (Raspberry Pi 3 Model B+)

Kaynak: (Raspberry pi, 2019)



Şekil 2.8: Mini PC'nin Kamera Kartı

Kaynak: (Pisupply, 2019)

Kamera modülünün özellikleri:

- 5MP Omnivision 5647 Kamera Modülü
- Kamera Seri Arabirimi - Doğrudan Raspberry Pi'ye Takılır
- Boyut: 20x25x9mm
- Ağırlık 3g

2.1.10 Görüntü işlemede kullanılan metot

Çalışmada YOLOv7 kullanılarak görüntü işleme ve hedef yakalama uygulamaları gerçekleştirilmiştir.

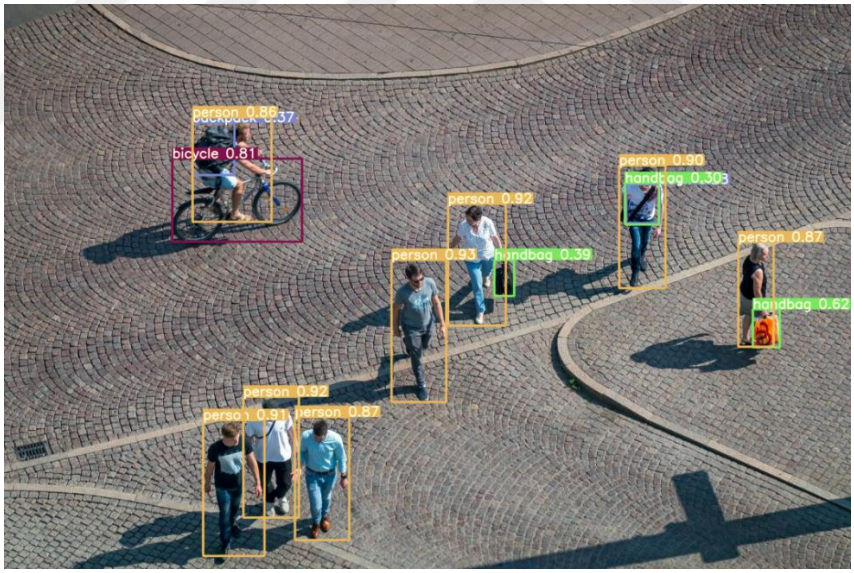
İHA'lar yerden açılacak ateşlerden korunabilecek yüksekliklerden uçarken askeri araçları tespit edilebilmesi için, kullanacak algoritmanın bu araçları daha önceden tanıyor olması gerekir. Bunun içinde insansız hava aracında kullanacak görüntü işleme algoritmasının eğitebileceği veri setlerine ihtiyaç duyulmuştur. Bu setleri elde edebilmenin yollarından biri, uygun yükseklikten çekilmiş yeterince askeri araç fotoğrafının veri setinde olmasıdır. Ancak bu tür fotoğraflar genellikle askeri gizliliğe sahip oldukları için açık kaynaklar üzerinden elde edilmesi güçtür. Bu çalışmada kullanılan fotoğraflar, Google Earth gibi uygulamalardan veya internetteki haber veya askeri kanallardan elde edilebilmiştir.

Bu tezin önceki çalışmalardan ayrılan en önemli özelliği hedefin tanımlanması sırasında veri setlerinden yararlanılmasıdır. Önceki çalışmalarda araç takibi yapılacaksa hedef boy ve en ölçüleri ayarlanarak bir dikdörtgen olarak tanımlanmış ve takibi bu şeklinde yapılmıştır. Bu çalışmalarda sivil ve askeri araç ayırımı mümkün değildir. Ancak bu çalışmada hedef İHA tarafından imha edilecek olması sebebiyle sivil araç ile düşman askeri araçların birbirinden ayrılması gerekmektedir. Çok yukarılardan bakıldığında bile askeri araçların seçilebileceği eğitilmiş bir algoritma ile hedef tespiti planlanmaktadır. Hedeflerin tespiti ve sınıflandırmasında hazır eğitilmiş veri setleri üzerinden çalışan YoloV7 Algoritması tercih edilmiştir. Hız ve kesinlik açısından gelenekselleşmiş görü sistemlerine göre yüksek performens ve verim elde edilebilen bu yöntem ile hedef tanıma ve tespiti sağlanmaya çalışılmıştır.

2.1.10.1 YOLOv7

Bu bölümde, YOLOv7'nin nasıl çalıştığına ve onun günümüzde mevcut olan nesne algılama algoritmaları arasındaki yerine değinilecektir, (J.Redmon & A. Farhadi, 2018).

- YOLO Gerçek Zamanlı Nesne Algılama
- Gerçek zamanlı nesne algılama nedir?
- Bilgisayar görüşünde YOLO nedir?
- Uçta nesne algılamayı verimli bir şekilde çalıştırma
- YOLOv7 nedir
- YOLOv7 Nesne Algılama Performansı
- YOLOv7 Mimarisi
- YOLOv7 uygulamaları



Şekil 2.9: YOLOv7 Algoritması

Kaynak: Viso.ai (2022)

YOLOv7 algoritması, hız ve doğruluk açısından önceki nesne algılama modellerine ve YOLO sürümlerine göre avantajlı özeliklere sahiptir. Diğer sinir ağlarına göre birkaç kat daha ucuz donanım gerektiren ve önceden eğitilmiş herhangi bir ağırlık olmaksızın küçük veri kümelerinde eğitilebilir yapısı mevcuttur, (Şahin,O., Özer, S. 2022)

2.1.10.2 Bilgisayar görüşünde YOLO nedir?

YOLO, “Sadece Bir Kez Bakarsınız” anlamına gelen, popüler bir gerçek zamanlı nesne algılama algoritmaları ailesidir. Joseph Redmon, Ali Farhadi ve Santosh Divvala tarafından oluşturulan YOLO piyasaya sürüldüğünde, bu mimari diğer nesne algılayıcılara göre çok daha hızlı ve gerçek zamanlı bilgisayarla görme uygulamaları için son teknoloji ürünlerden biri haline gelmiştir, (docs.ultralytics,2023)

Her biri performans ve verimlilikte önemli bir artış sağlayan farklı YOLO sürümleri ve çeşitleri zamanlarda kullanıcıya önerilmiştir.

Resmi olmayan ticari ağlarında diğer tüm sayıları atlayarak doğrudan YOLOv4'ten v7'ye geçmesini beklenmektedir.

Şekil 2.10: Halka Açık Yerlerde Kalabalık Tespiti İçin Görüntü İşleme

Kaynak: Viso.ai (2022)

2.1.10.3 Gerçek zamanlı nesne dedektörleri ve YOLO sürümleri

Son zamanlarda yaygın olarak kullanılmaya başlanılan gerçek zamanlı nesne algılayıcıları temel olarak YOLO ve FCOS'a (Tam Evrişimli Tek Aşamalı Nesne Algılama) dayanmaktadır. Performans açısından başarı gösteren nesne dedektörleri şunlardır:

- Redmon ve diğerleri tarafından tanıtılan YOLOv3 modeli 2018'de
- Bochkovskiy ve diğerleri tarafından yayınlanan YOLOv4 modeli. 2020'de
- YOLOv4-tiny modeli, 2021'de yayınlanan araştırma

- 2021'de yayınlanan YOLOR (Yalnızca Bir Temsil Öğrenirsiniz) modeli
- 2021 yılında yayınlanan YOLOX modeli
- 2021'de yayınlanan NanoDet-Plus modeli
- 2022'de yayınlanan endüstriyel bir nesne dedektörü olan PP-YOLOE
- 2022'de Ultralytics tarafından yayınlanan YOLOv5 model v6.1
- 2022'de yayınlanan YOLOv7 modeli, (docs.ultralytics,2023)

2.1.10.4 YOLOv7 nedir

Temmuz 2022'de Chien-Yao Wang, Alexey Bochkovskiy ve Hong-Yuan Mark Liao tarafından YOLOv7: gerçek zamanlı nesne dedektörleri için yeni ve son teknoloji ürünü ayarlar" adlı resmi bir YOLOv7 makalesi yayınlar, (Şahin,O., Özer, S. 2022)

YOLO, evrişimsel sinir ağlarını (CNN) kullanarak nesne tespiti yapması için geliştirilmiş bir algoritmadır. Açılımı “You Look Only Once”, “Yalnızca Bir Kez Bak” anlamına gelmektedir. Çalışmasının temel prensibi görüntünün tek seferde nöral bir ağdan geçiriliyor olmasıdır.

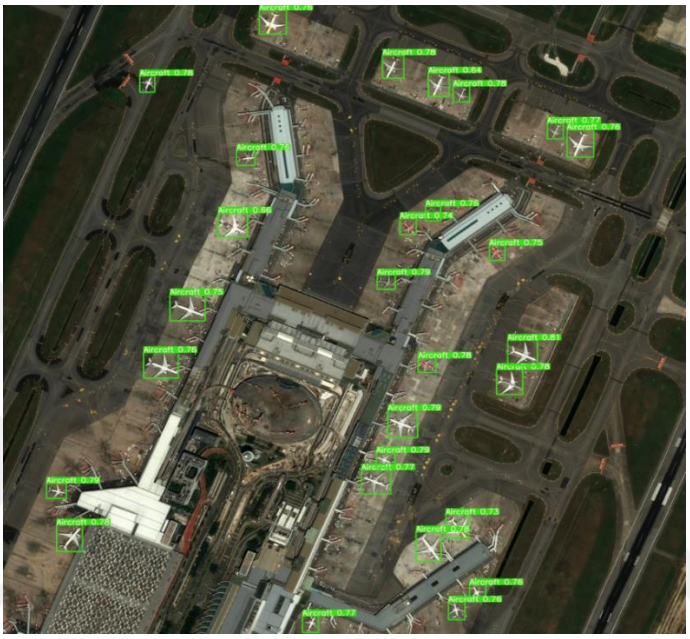
YOLO mimarisinin temelinde CNN mantığı yatmaktadır. Görüntü ilk olarak evrişim katmanından geçerek filtreleme işlemleri uygulanır. Diğer bir ifade ile görüntünün özniteliklerini çıkararak nesneleri sınıflandırır, (burakzzd, 2023).

Şekil 2.11: Yolov7, Viso Suite Üzerine İnşa Edilmiş, İnşaatta Bir Bilgisayarla Görme Uygulaması

Kaynak: Viso.ai (2022)

2.1.10.5 Temel YOLOv7 sürümleri arasındaki farklar

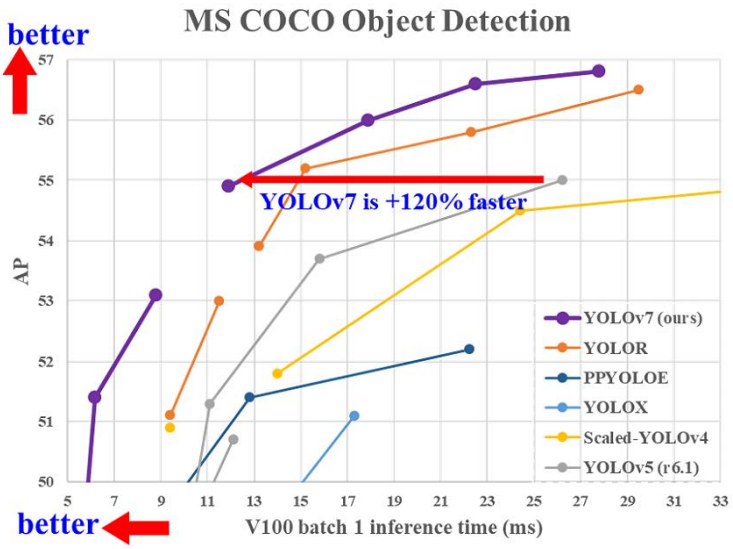
Farklı temel YOLOv7 modelleri arasında YOLOv7, YOLOv7-tiny ve YOLOv7-W6 bulunur: Görüntü işleme modellerinin "minik" soneki, Uç Yapay Zekâ ve derin öğrenme iş yükleri için optimize edildikleri ve mobil bilgi işlem cihazlarında veya dağıtılmış uç sunucular ve cihazlarda makine öğrenimi çalıştırmak için daha hafif oldukları anlamına gelir, (docs.ultralytics,2023)



Şekil 2.12: Uçak Tespiti İçin YOLO V7 Kullanan Bilgisayar Görüş Sistemi

Kaynak: Viso.ai (2022)

2.1.10.6 YOLOv7 nesne algılama performansı



Şekil 2.13: Diğer Gerçek Zamanlı Nesne Dedektörleriyle Karşılaştırma

Kaynak: Viso.ai (2022)

Cascade -Maske R-CNN modelleriyle karşılaştırıldığında YOLOv7, önemli ölçüde artırılmış çıkarım hızında (%509 daha hızlı) %2 daha yüksek doğruluk elde eder, (viso.ai, 2022).

YOLOv7, hız ve doğrulukta birçok nesne algılama algoritmasından daha iyi performans gösterir.

Çizelge 2.2: Temel Nesne Dedektörleri YOLOR Ve Yolov4'ün Yeni Yolov7 İle Karşılaştırılması

Model	#Param.	FLOPs	Size	AP^{val}	AP^{val}_{50}	AP^{val}_{75}	AP^{val}_{S}	AP^{val}_{M}	AP^{val}_{L}
YOLOv4 [3]	64.4M	142.8G	640	49.7%	68.2%	54.3%	32.9%	54.8%	63.7%
YOLOR-u5 (r6.1) [81]	46.5M	109.1G	640	50.2%	68.7%	54.6%	33.2%	55.5%	63.7%
YOLOv4-CSP [79]	52.9M	120.4G	640	50.3%	68.6%	54.9%	34.2%	55.6%	65.1%
YOLOR-CSP [81]	52.9M	120.4G	640	50.8%	69.5%	55.3%	33.7%	56.0%	65.4%
YOLOv7	36.9M	104.7G	640	**51.2%**	**69.7%**	**55.5%**	**35.2%**	**56.0%**	**66.7%**
improvement	-43%	-15%	-	+0.4	+0.2	+0.2	+1.5	=	+1.3
YOLOR-CSP-X [81]	96.9M	226.8G	640	52.7%	**71.3%**	57.4%	36.3%	57.5%	68.3%
YOLOv7-X	71.3M	189.9G	640	**52.9%**	71.1%	**57.5%**	**36.9%**	**57.7%**	**68.6%**
improvement	-36%	-19%	-	+0.2	-0.2	+0.1	+0.6	+0.2	+0.3
YOLOv4-tiny [79]	6.1	6.9	416	24.9%	42.1%	25.7%	8.7%	28.4%	39.2%
YOLOv7-tiny	6.2	5.8	416	**35.2%**	**52.8%**	**37.3%**	**15.7%**	**38.0%**	**53.4%**
improvement	+2%	-19%	-	+10.3	+10.7	+11.6	+7.0	+9.6	+14.2
YOLOv4-tiny-3l [79]	8.7	5.2	320	30.8%	47.3%	32.2%	**10.9%**	31.9%	51.5%
YOLOv7-tiny	6.2	3.5	320	**30.8%**	**47.3%**	**32.2%**	10.0%	**31.9%**	**52.2%**
improvement	-39%	-49%	-	=	=	=	-0.9	=	+0.7
YOLOR-E6 [81]	115.8M	683.2G	1280	55.7%	73.2%	60.7%	40.1%	**60.4%**	69.2%
YOLOv7-E6	97.2M	515.2G	1280	**55.9%**	**73.5%**	**61.1%**	**40.6%**	60.3%	**70.0%**
improvement	-19%	-33%	-	+0.2	+0.3	+0.4	+0.5	-0.1	+0.8
YOLOR-D6 [81]	151.7M	935.6G	1280	56.1%	73.9%	61.2%	**42.4%**	60.5%	69.9%
YOLOv7-D6	154.7M	806.8G	1280	56.3%	73.8%	61.4%	41.3%	60.6%	70.1%
YOLOv7-E6E	151.7M	843.2G	1280	**56.8%**	**74.4%**	**62.1%**	40.8%	**62.1%**	**70.6%**
improvement	=	-11%	-	+0.7	+0.5	+0.9	-1.6	+1.6	+0.7

Kaynak: Viso.ai (2022)

2.1.10.7 YOLOv7 uygulamaları

Bu çalışmanın ilerleyen bölümlerinde YOLOv7'nin farklı endüstrilerdeki gerçek dünya uygulamaları listelenmiştir. Nesne detektörünün, genellikle kamera entegrasyonundan görüntü elde etmeye, işlemeye, çıktı biçimlendirmeye ve sistem entegrasyonuna kadar bir dizi adımı içeren tüm bir görüş hattının yalnızca bir parçası olduğunu anlamak önemlidir, (github.com,2022)

2.1.10.7.1 Güvenlik ve gözetim

Nesne algılama aynı zamanda birçok yüz tanıma sisteminin ayrılmaz bir parçasıdır

Şekil 2.14: Derin Öğrenme İle Yüz Algılama

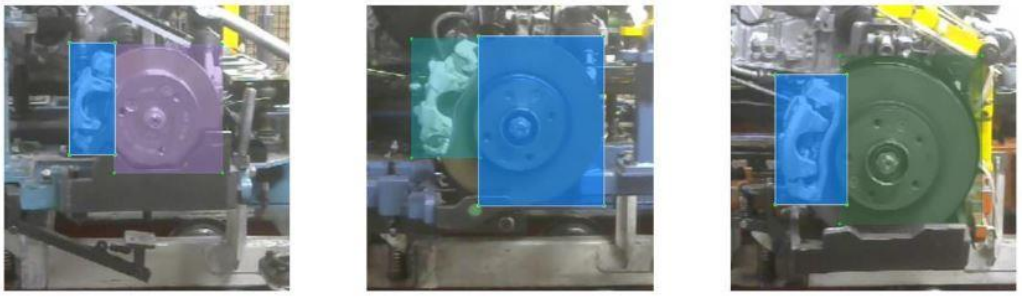
Kaynak: Viso.ai (2022)

2.1.10.7.2 Akıllı şehir ve trafik yönetimi

Bu nedenle, nesne algılamanın akıllı şehirlerde büyük insan kalabalığını analiz etmek ve altyapıyı denetlemek için birçok kullanım durumu vardır.

Şekil 2.15: YOLO Modellerini Kullanarak Perakende Mağazalarında Kişi Tespiti

Kaynak: Viso.ai (2022)



Şekil 2.16: Otomotiv İmalatında Nesne Tespiti İçin Bilgisayar Görüşü

Kaynak: Viso.ai (2022)

2.1.11 Nesne takibi yöntemleri

Nesne takibi yöntemleri temel olarak 3 kategoriye ayrılır. Bunlar sırasıyla nokta tabanlı, çekirdek tabanlı ve siluet tabanlı yöntemlerdir. Nokta takip yönteminde takip edilecek nesne noktalar ile ifade edilir. Bu noktaların bir sonraki imgedeki konumları ve birbirlerine olan uzaklıkları gibi verilerin sonraki gelen video çerçevesinde de birbirine paralel olması beklenir. Bu bilgilerden yola çıkılarak nesne takibi sağlanır. Bu yöntemde temel amaç nesnenin video çerçevesi içinde tespit edilmesi ve bir önceki çerçevede kullanılan nokta benzerliklerin hesaplamasıdır. Bu yöntem deterministik ve istatistiksel yöntemler olarak ta kendi içinde alt sınıflandırmalar içermektedir Talu M.F. (2010), Yilmaz A., Javed O., Shah M. (2006). Nokta takibi yöntemlerinde en yaygın kullanılan yöntemlerden birisi kalman filtresidir. Bu yöntem nesnenin Gaussian dağılıma sahip durum değişkenleri yardımıyla videodaki bir sonraki gelen çerçevede nesne konumunu tahmin etmektedir. Kalman filtresi basit ve hızlı olma açısından gerçek zamanlı nesne takip uygulamalarında kullanıma uygundur Talu M.F. (2010), Parekh H.S., Thakore D.G., Jaliya U.K. (2014). Durum değişkenleri Gaussian dağılımına sahip olmayan sistemlerde kalman filtresi başarısız olabilmektedir. Bu tür problemlerin giderilmesi için parçacık filtresi yöntemi geliştirilmiştir. Parçacık filtresi olasılıksal yönteme dayanmaktadır. Bu yöntemin en büyük avantajı doğrusal olmayan ve çoklu dağılıma sahip sistemlerde çalışabilmesidir Fan L., Wang Z., Cail B., Tao C. (2016). Çekirdek tabanlı yöntemlerde bir geometrik şekil yardımıyla takip edilecek nesne çerçevelenir. Bu çerçeve içerisinde bulunan nesne parçasının anlamlı bilgileri hesaplanarak başlangıçtaki şekil yardımıyla nesne takip edilir. Bu yöntemde nesnenin şeklinden ziyade kullanılan geometrik şeklin içerisinde bulunan nesne bilgilerinin çıkarılması yeterli olabilmektedir. Bu şekil içinde bulunan piksellerin hesaplanan olasılık yoğunluk bilgileri veya histogram özellikleri gibi bilgileri sonraki video çerçevelerinde takip edilebilmektedir. Siluet tabanlı yöntemler genellikle takip edilen nesnenin insan ya da hayvan gibi belli bir geometrik şekille ifade edilemediği durumlarda kullanılır. Bu yöntemin temel amacı nesneyi tanımlayacak kenar bilgisi ya da şekil bilgisi çıkartılarak sonraki imgelerde bu bilgiyi aramaktır. Bu yöntem şekil değişikliğine karşı oldukça hassas olmaktadır. Çekirdek ve siluet tabanlı yöntemler kıyaslandığında, çekirdek tabanlı yöntemlerin daha düşük işlem zamanına ve daha yüksek başarı oranlarına sahip oldukları görülmektedir. Bu sebepten dolayı

çalışmalarda çekirdek tabanı yöntemler geniş bir kullanım alanına sahiptir. Nokta tabanlı yöntemler diğer yöntemlere oranla daha düşük işlem zamanına sahip olmakla birlikte daha düşük başarı oranına sahiptirler.

2.2 İHA Kontrolü

İHA'nın yüksek hassasiyetle düşmanı algılayabilmesi için yakın bir mesafede uçabilmesi gerekmektedir. Bu duruma uygun bir şekilde mekanik donanımlar seçilmiştir. Bu seçimler sonucunda çeşitli programlar kullanılıp İHA'nın itiş gücüne ilişkin testler yapılmıştır.

İHA'ların istenilen hassasiyette yüksek manevra yeteneğine sahip bir şekilde uçabilmesi için yeterli koşul, itiş gücünün dronun hesaplanan ağırlığının yaklaşık 2 katı olmasıdır. Havada olan bir cismin dengede kalabilmesi için dronun ağırlığı, motorların pervane sayesinde oluşturduğu itiş kuvvetine (Fitiş) eşit olmalıdır.

Bu hesaplar göz önünde bulundurularak aranan özelliklere sahip motor ve pervaneler incelenmiştir. Mekanizmaların itişleri ve motorların çektikleri akımlar, uygulanan gerilime ve kanat boyutuna göre değişmektedir. Yapılan hesaplamalar sonucu DJİ 2212/920 kv motor ile DJİ Propeller (10 inç boyutlarında) markalar seçilmiştir.

İHA yüksek manevra kabiliyetine sahip olmasının yanı sıra üzerindeki faydalı yükleri taşıyabilecek bir yapıda seçilmiştir. Bu yüzden desteklediği motor sayısı 6 olan gövde tercih edilmiştir. Ayrıca dronun üzerine konulacak olan, 5300mAh pil (uçuş süresini maksimum sürede tutmak için) gibi elemanları taşıması gerektiğinden gövdenin mümkün olduğunca büyük seçilmesine özen gösterilmiştir.

Çizelge 2.3: F450, F550 ve TF680H4 Gövdelerinin Teknik Özellikleri

Özellikler	**F450**	**F550**	**TF680H4**
Yapıldığı Malzeme	Plastik	Plastik	Karbon Fiber
Ağırlık (gram)	270 gram	478 gram	590 gram
Desteklediği Motor Sayısı	4	6	4
Boyutları (Genişlik, Yükseklik)	450 mm, 55 mm	550 mm, 60 mm	680 mm, 410 mm

Kaynak: (Tiwari, M. Ve Singhai, R. 2017: s.48)

İHA'nın nihai durumda yaklaşık olarak özellikleri şunlar olacaktır:

- Yaklaşık olarak toplam ağırlığı: 1.5 – 2 kilogram.

İHA döner kanatlı olacak şekilde 6 kanatlı üretilecektir. İtki sisteminde 3s 5300 mah. lipo pil ve 920 kv'lik motorlar kullandığından çalışma için yeterli bir itki sağlanabilecek ve ESC'nin çalışabilmesi için yeterli anlık akım karşılanabilecektir. İHA da kullanılacak bütün elemanlar göz önüne aldığında 4 kg ağırlığını geçmeyecek bir İHA tasarımı ortaya çıkarılması hedeflenmektedir.

Bu çalışmada kullanılacak İHA döner kanatlı olacağı için kalkış mesafesi 0 kabul edilecektir. İHA'da 3s 5300 mah lipo pil, 10 inçlik pervaneler ve kv değeri 920 olan motorlar kullanıldığında pervanenin en uç kısmından alınan maksimum hız değeri yaklaşık olarak 744,055 km / saat olmaktadır. Bu hız değeri yüksüz durumdaki hız değeridir. İHA'nın kamera mekanizması sabit olacak ve faydalı yüklerin toplam ağırlığı yaklaşık olarak 250-400 gr arasında olacaktır. İHA'nın ağırlık merkezini dikkate alarak kamera mekanizmasının orta kısımda olması planlanmaktadır. İHA'da kullanılan RC kumanda sinyal karışıklıklarını önlemek amacıyla DSM özellikli olacaktır. Donanımsal olarak uzaktan kontrol sistemi ile uzaktan kontrol sistemi yapılacaktır. Bu nedenle otonom kontrol edilecektir. Hız, GPS, motor durumu, kanat durumları, uçuş pozisyonu, eksen dereceleri hakkında bilgilendirme yapılacak bir kontrol bilgisayarı kullanılacaktır.

2.2.1 İHA bileşenleri

2.2.1.1 Elektrikli bileşenler

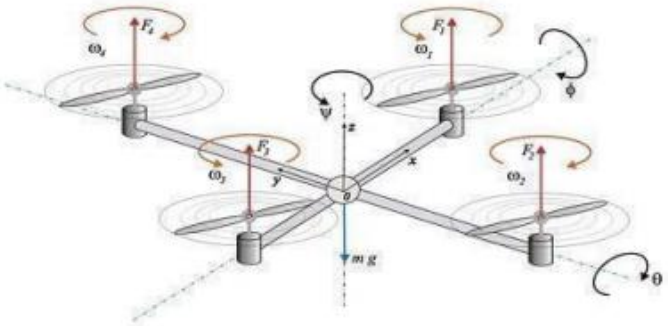
2.2.1.1.1 Fırçasız motorlar

Fırçasız motorların (Şekil 2.17) diğer motorlara göre daha sessiz çalışma, elektriksel gürültü oluşturmama, daha kolay bakım, daha uzun ömür, daha hızlı çalışma ve daha güçlü torklara sahip olma gibi avantajları vardır.

Şekil 2.17: Kullanılan 2212 920KV Fırçasız Dc Motor Modeli

Kaynak: (Bolubeyi, 2022)

Multicoptor'larda Şekil 2.18'de görüldüğü gibi 2 farklı dönüş yönüne sahip pervane kullanılır. Motorların 2 farklı yöne düzgünce dönebilmesi için hem pervane yönleri hem de esc-motor bağlantılarının ters olması gerekmektedir



Şekil 2.18: Pervane Dönüş Yönleri

Kaynak: (Bolubeyi, 2022)

Bu çalışmada 3 adet CW (saat yönünde) ve 3 adet CCW (saat yönünün tersine) pervaneleri döndürmek için yüksek hızlı fırçasız DC (BLDC) motorlar kullanılmaktadır. Şekil 2.18'de bir motor parçası gösterilmektedir. Motor fırçasız olarak isimlendirilir. Yani dış kasa dönerken iç kısım sabit kalır. Pervaneleri döndürmek için fırçalı tip motora göre fırçasız motor daha fazla tork üretir, bu nedenle İHA'lar için kullanılmaya daha uygundurlar (Dronetrest, 2019).

Kv, üretilen back-emf'in motor motor hızı ile ilgilidir. Böylece motor 920 rpm'de bir volt ters emk verir. Kv değeri ile gerilim seviyesi çarpılarak gerilim elde edilir.

2.2.1.1.2 A 2212 920KV fırçasız motor

Çalışmada fırçasız olarak isimlendirilen A2212 model motorlar kullanılacaktır. A2212 920 KV Fırçasız Motor, 4-6-8 kollu multicopter ve quadcopter dronlar ile uyumludur.

A2212 920 KV Fırçasız Motor Teknik Özellikleri:

Boyutlar: 28X24mm

Kv: 920kv

Ağırlık: 56gr

Mil: 8.0mm

Çalışma voltajı: 3S-4S (11-16V)

Maks akım: 15-20A

Uygun ESC: 25-30A

Uygun pervane: 10x4.5, 10x5, 9x6 (3S için), 8x5, 8x4.5 (4S için)

2.2.1.1.3 Batarya

Batarya kapasitesi uçuş süresini belirleyin en önemli parametrelerden biridir. Bu çalışmada İHA'larda kullanılan fırçasız motorların devir hızları 10000 rpmdir. Bu kapasitede bir motorun ve ESC lerin anlık ihtiyaç duyacağı elektrik kuvvetini iletecek piller uygun ölçü ve kapasitede seçilmelidir. (Revolutions per minute – 1 dakika içerisinde gerçekleştirilen dönüş/devir sayısı) Kullanılan pilin fotoğrafı Şekil 2.19 da görülebilir.

Şekil 2.19: Kullanılan 5300 mAh 3S Li-po Pil

Kaynak: (Robotistan ,2022)

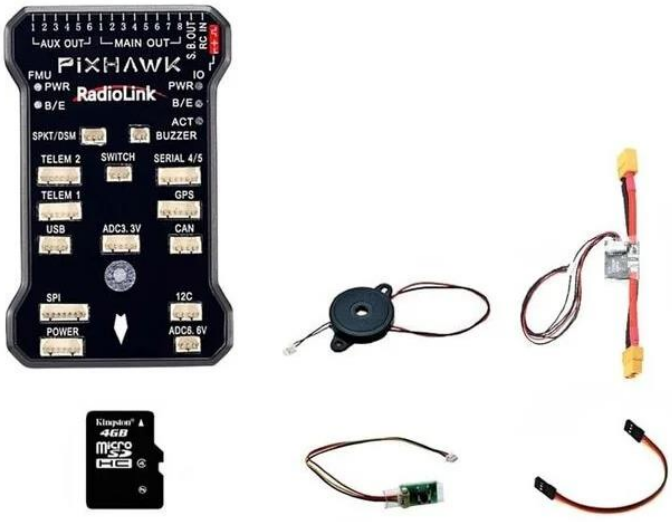
2.2.1.2 Elektronik bileşenler

2.2.1.2.1 Uçuş kontrolörü

Uçuş kontrolörü birim sistemin beynidir ve tüm dengede kalma, kumanda verisi okuma, batarya kontrolü gibi kritik işlemler bu birim sayesinde gerçekleştirilmektedir. Bu kapsamda sistem üzerindeki en kritik parçadır. Temel görevi denetim birimi ve algılayıcılardan gelen bilgiyi alma, işleme ve gerekli şekilde çıkışa aktarmaktır. Bu çalışmada Pixhawk uçuş kontrolörü kullanılacaktır. Temsili Pixhawk devre şeması ve Pixhawk uçuş kontrolörü (Şekil 2.20ve Şekil 2.21) de görülebilir.

Şekil 2.20: Pixhawk Devre Şeması Temsili

Kaynak: (oyuncakhobi.com, 2022)



Şekil 2.21: Pixhawk Uçuş Kontrolörü Üstten Görünüm ve Bağlantı Noktalarının Görünümü

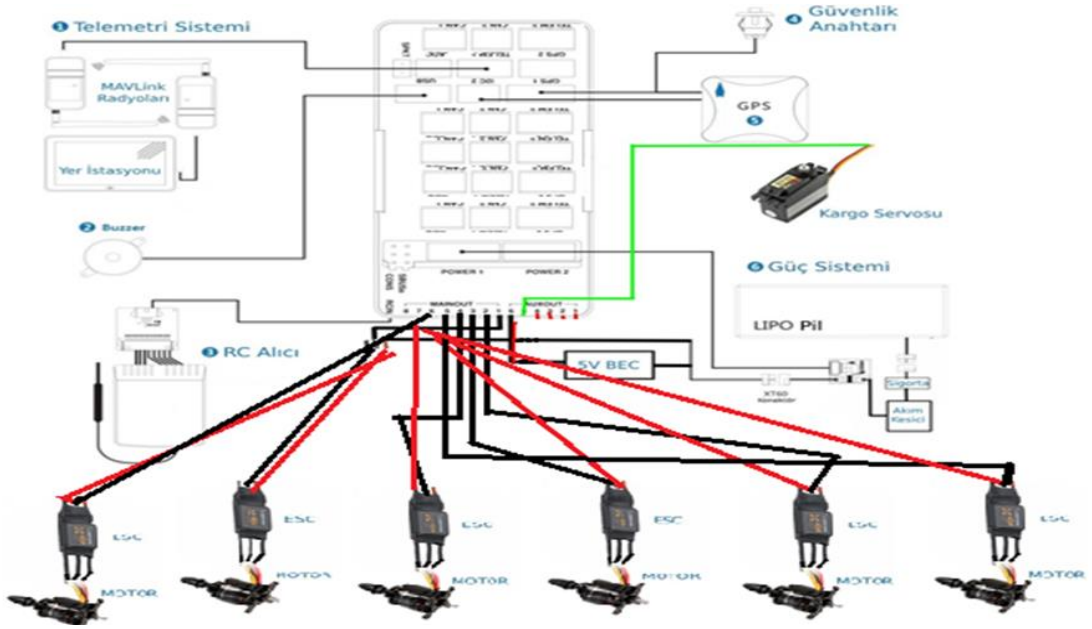
Kaynak: (oyuncakhobi, 2022)

Şekil 2.21: (Devamı) Pixhawk Uçuş Kontrolörü Üstten Görünüm ve Bağlantı Noktalarının Görünümü

Kaynak: (oyuncakhobi, 2022)

2.2.1.2.2 Radiolink pixhawk 32 bit uçuş kontrol kartı + güç modülü

Pixhawk uçuş kontrolörü, Ardupilot mega veya 3DR robotikten açık kaynaklı bir proje olan "APM" ye dayanmaktadır. Bu uçuş kontrol cihazı, kullanıcının isteğine bağlı olarak GPS Modülüyle ve ara noktalarla programlanmış GPS görevlerini yerine getirebilen sabit kanatlı, döner kanatlı veya çok kanatlı taşıtları (hatta tekneleri ve arabaları) tamamen otonom bir araca dönüştürmesini sağlar. Şekil 2.22 Sistem devre şeması gösterilmiştir.



Şekil 2.22: Sistem Devre Şeması

Kaynak: (Robot malzemeleri, 2022)

2.2.1.2.3 GPS ve pusula

Bu çalışmada konum belirlemek ve irtifa tanımlayıcısının ikinci kaynağı için GPS kullanıldı. GPS bağlantısının arducopter üzerinde kendine ait pini vardır. GPS, arducopter içindeki sensörlerin mayetik alanlarından etkilenmemek için genellikle bir çubuk ile daha yüksek bir yere sabitlenir. GPS monte edilirken dikkat edilmesi bir gereken husus ise GPS ile arducopter ve dronun önü aynı yöne bakmalarıdır. GPS ve pusula modülü Şekil 2.23'te gösterilmiştir.

Şekil 2.23: GPS Modülü

Kaynak: (Robot malzemeleri, 2022)

2.2.1.2.4 Uzaktan kumanda + receiver

Projenin geliştirme aşamasında uzaktan kumanda kullanılarak kontrollü uçuş sağlanmıştır. Otonom uçuş denemelerinde güvenlik prosüdürü olarak uçuş esnasında gerekli durumlarda İHA'yı manuel olarak kontrol etmek için devamlı hazır konumda bekletilir. Çalışma sırasında kullanılan uzaktan kumanda Şekil 2.24'te görülebilir.

Şekil 2.24: 2.4 GHz'lik 6 Kanallı Kumanda Verici ve Alıcı

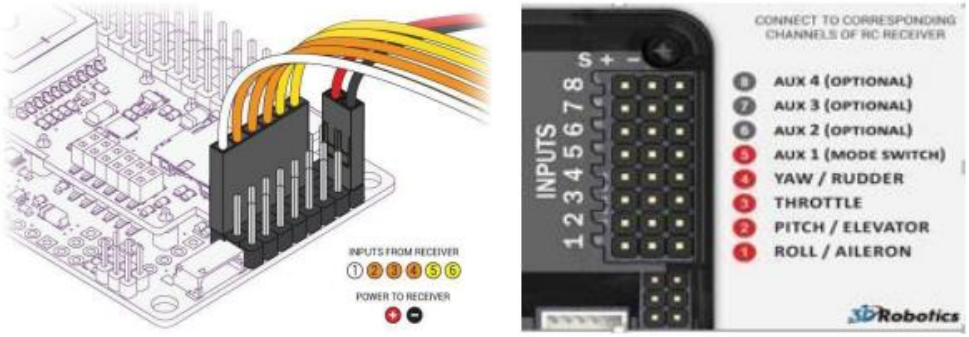
Kaynak: (Engineerstoys, 2023)

2.2.1.2.5 Uzaktan kumanda alıcısı

Şekil 2.25: Uzaktan kumanda alıcısı

Kaynak: (Engineerstoys, 2023)

Bir İHA'yı uçurmak için en az 4 kanala (Thorttle,Elevator,Aileron ve Rudder) sahip bir hava aracı kumandası kullanılmalıdır. Kanallar Şekil 2.26'daki gibi arducopterin input pinlerine takılır.



Şekil 2.26: Kanal Bağlantıları

Kaynak: (Arducopter, 2022)

2.2.1.2.6 Telemetri alıcı-vericisi

Radyo telemetri modülleri, kablosuz iletişim için yer istasyonu ile İHA nın iletişimini sağlar. Telemetre (Şekil 2.27), Bir aygıtı yer istasyonuna (bilgisayar) diğer ucu ise arducoptera takılır.

Şekil 2.27: Telemetri Modülü Alıcı-Vericileri

Kaynak: (Holybro, 2022)

Çizelge 2.4: Telemetre Sinyal Tablosu

Özellik	Değer
RF Veri Hızı	9,6 kbps
RF Çıkış Gücü	+20 dBm (100 mW)
Modülasyon Tipi	FHSS (Frequency Hopping Spread Spectrum)
RF Bant Aralığı	902-928 MHz
RF Alıcı Hassasiyeti	-106 dBm
Haberleşme Mesafesi	9,6 km
Seri Veri Arayüzü	UART
Seri Veri Hızı	1,2-57,6 kbps
Besleme Voltajı	3,3 V
Besleme Akımı (Maks.)	256 mA

Kaynak: (Tiwari, M. Ve Singhai, R. 2017: s.48)

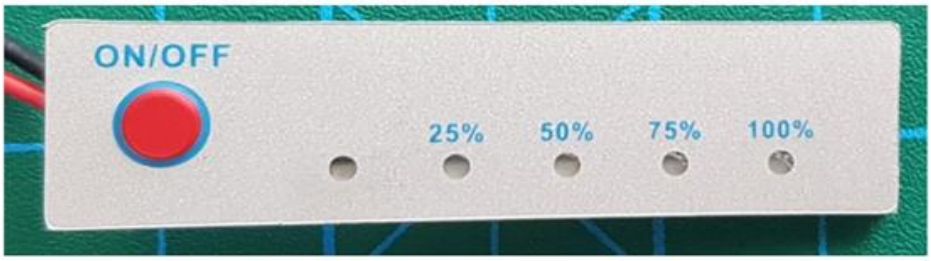
Telemetri radyo modülünün özellikleri: 915 veya 433 MHz

- 100 mW maksimum çıkış gücü (ayarlanabilir)
- -117 dBm alma hassasiyeti Uyarlanabilir TDM aracılığıyla 2 yönlü tam çift yönlü iletişim
- UART arayüzü
- Şeffaf seri bağlantı
- MAVLink protokol çerçevesi
- Frekans Atlamalı Yayılmış Spektrum (FHSS)
- Hata düzeltme, bit hatalarının %25'ine kadarını düzeltir

2.2.1.2.7 Pil göstergesi

Daha güvenli uçuş sağlamak ve çalışma kolaylığı için pil göstergesi kullanılması düşünülmektedir. Akü voltaj seviyesi yer kontrol istasyonundan da gözlemlenebilir ancak daha doğru bir sonuç elde etmek için pil voltaj seviyesi yardımı ile izlenmelidir.

Batarya voltaj seviyesi göstergesi Şekil 2.28'da görülebilir.



Şekil 2.28: Batarya Voltaj Seviyesi Göstergesi

Kaynak: (Robotistan, 2022)

2.2.1.2.8 ESC (Elektronik hız kontrolörü)

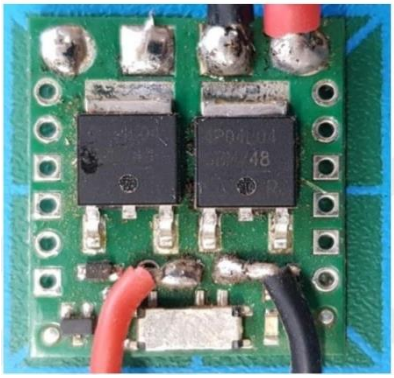
Fırçalı ve fırçasız motorların hızını ayarlayan, hız kontrol ünitesidir. Diğer bir ifade ile pilden aldıkları elektrik enerjisini, alıcının gaz kanalından aldığı sinyal ile sürerek motorlara ileten ve motor devrini kontrol eden hız kontrolcüleridir (Şekil 2.29).

Şekil 2.29: Kullanılan 30A'lik ESC Modeli

Kaynak: (Robotistan, 2022)

2.2.1.2.9 Güç anahtarı

Bu çalışmada besleme voltajını kontrol etmek için bir güç anahtarı kullanıldı. Bu modül pile takılarak Açma/Kapama anahtarını sisteme verir ve bu sistemin kısa sürede enerjilenmesini engeller.



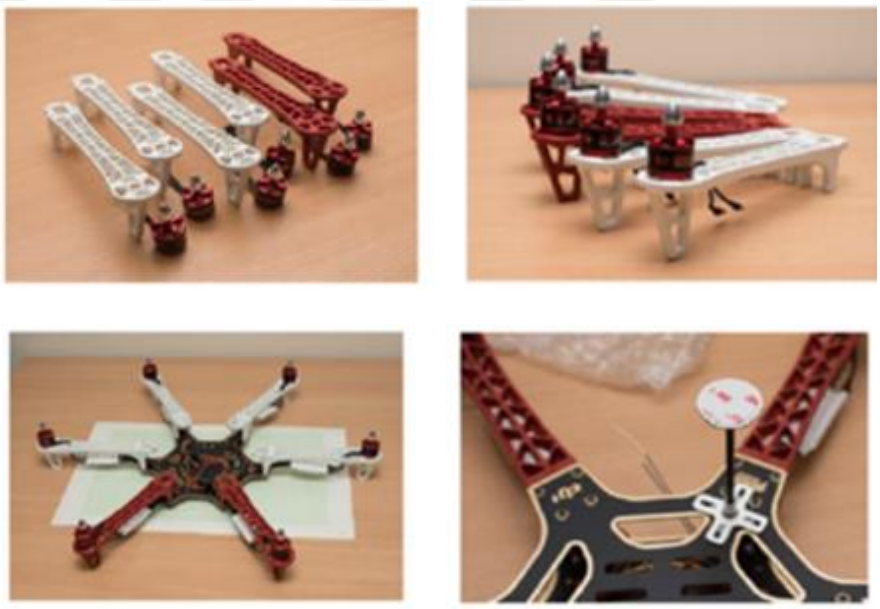
Şekil 2.30: Güç Düğmesi

Kaynak: (Robotistan, 2022)

2.2.1.3 Mekanik bileşenler

2.2.1.3.1 Çerçeve ve pervaneler

İnsansız hava Aracının çerçevesi 3D yazıcıdan üretilebileceği gibi hazır olarak piyasadan da temin edilebilir. Bu çalışma esnasında üretilen İnsansız Hava Aracının çerçevesi piyasadan hazır alınmıştır. F 550 sınıfında 6 kollu Hexacopter sınıfı bir İHA'dır. Üzerine altı adet motor ve altı adet ESC yerleştirilmeye uygundur. Özellikle altı kollu bir gövde seçilmiştir. Bunun iki ana nedeni vardır. Birincisi altı kollu İHA'lar tek bir motor arızasında bile güvenli uçuş ve iniş kabiliyetine sahiptir. İkinci ana nedeni ise Altı kollu İHA'lar faydalı yük olarak tanımlanan kamera, taşıma haznesi, gerektiğinde mühimmat gibi 2 ila 3 kilo arasında bir ağırlığı taşıma kapasitesine sahiptirler. Rüzgar dirençleri oldukça yüksektir. Olumsuz hava koşullarında nispeten stabil uçuş imkanı sağlayabilirler.



Şekil 2.31: Montaj Aşamaları

Kaynak: Kişisel Arşiv

Şekil 2.32: 1045 İHA Pervanesi Seti Cw/ccw – Siyah

Kaynak: (Robotzade, 2022)

1045 İHA Pervanesi Seti CW/CCW - Siyah 1045 (10X4.5) ABS mavi pervaneler, multicopterler için özel olarak tasarlanmış yüksek kaliteli pervanelerdir. Hafif ve yüksek mukavemetli pervane, uçarken girdaptan kaçınmak için pervanenin sonunda 15° açılı bir tasarıma sahiptir.

2.2.4 Test ve simülasyon

2.2.4.1 Test sonucu ile yapılan tasarımın uyumluluğu

Yapılan testler sonucunda İHA da kullanılan ekipmanların göreve tam uyumlu olduğu tespit edilmiştir. 6 motorlu yapısı ile İHA nın yeterince güçlü ve stabil olduğu, çıplak gözle görünme rasyosunun manuel modda kullanıma uygun olduğu, pillerin minumum uçuş süresini sağlayabildiği, telemetri ve yer istasyonu ile görüntü aktarım organlarının tam uyumlu çalıştığı tespit edilmiştir.

2.2.5 Güvenlik

2.2.5.1 Güvenlik ihtiyaçlarının karşılanması için önlemler ve çözüm yöntemler

Güvenlik kontrollerinde aşağıdaki sıaralanan unsurların asgari düzeyde sağlamasına dikkat edildi: Aracın, teknik çizimleriyle uyumlu , yapısal/görsel bütünlük yönünden güvenli, tüm bileşenlerin güvenli bir şekilde İHA'ya monte edilmiş, tüm bağlantıların sıkı ve emniyet teli, sıvı yapıştırıcı ve/veya somunla yapıldığı,bağlantı malzemelerinin uçuş sırasında bağlantıların kopmasını önleyecek şekilde seçildiği, pervanenin yapısal ve bağlantı bütünlüğü yeterli kalınlıkta tel ve konnektör kullanıldığı, radyo menzil kontrolü, motor açma ve kapama, İHA'nın tüm kontrol mekanizmalarının yeterli hassasiyete sahip olduğu, yük sisteminin genel bütünlüğü olduğu, tüm araç radyolarının sinyal kaybında otomatik olarak fail-safe moduna geçebilir özellikte olduğu, sigortanın İHA'nın dış yüzeyinde, kolay ulaşılabilir bir yerde monte edildiği, kontrol edilmiştir.

2.4 Maliyet Analizi

Tüm sistemin toplam maliyeti 13384 liradır. Ayrıntılı maliyet analizi Çizelge 2.5' te yazılmıştır.

Çizelge 2.5: İHA Sisteminin Ayrıntılı Maliyet Analizi

	Malzeme Adi	**Malzemenin Kullanım Amacı**	**Adedi**	**Maliyeti**
1	WFT 07 2.4 Ghz 6 Kanallı Kumanda Vericisi+Alıcı	İHA'nın genel kontrolünü sağlayan ana kumanda	1 adet	2006 TL
2	SimonK 30 A ESC	Motor Elektrik hız kontrol ünitesi	6 adet	1698 TL
3	Radiolink Pixhawk 32 Bit Uçuş Kontrol Kartı+Güç Modülü	İHA'nın Kontrol edilebilmesi için	1 adet	4300 TL
4	A2212 boyutlarında 920 kv'lik fırçasız	Uçuş enerjisini sağlayan ana unsur	6 adet	1980 TL
5	5300 mah 11.1v 25c 3s lipo pil	İHA ya enerji sağlamak	2 adet	3400 TL

3. GEREÇ VE YÖNTEM

Tez konusu çalışmanın başlangıçtan sonuçlanmasına kadar olan aşamaları 6 ana başlıkta ilerledi.

1.Adım: Askeri araç fotoğraflarının toplanması.

2.Adım: Roboflow Üzerinden Çalışma Seti Eğitildi. İkincil kullanıcılar için ticarileştirilebilecek "Ağırlık Dosyalarının" oluşturulması.

3.Adım: Google Colab üzerinden YOLOv7 ve YOLOv8 kullanılarak, videolardan hedef yakalama çalışmalarının oluşturulması.

4.Adım: Pycharm editörü üzerinden kullanılabilir program oluşturulmaya çalışılması.

5.Adım: HSV(Renk Tonu Doygunluğu Değeri) ile hedef yakalama çalışmasının yapılması. HSV(Renk Tonu Doygunluğu Değeri) ile YOLO'nun karşılaştırılması.

6.Adım: İHA üzerinden montaj çalışmalarının çalışılması.

3.1 Askeri Araç Fotoğraflarının Toplanması

Çalışma konusu Setin eğitilebilmesi için, en önemli ihtiyaç olan askeri araçların gökyüzünden görüntülerini elde edebilmek için, internetteki açık kaynaklardan, Google Map ve Google Earth, Kaggle gibi internet sitelerinden toparlanan fotolar ile bir veri seti oluşturdu.

Tez çalışmasında örnek fotoğraflar ile eğitilen algoritmalar kullanıldı. Görüntü işleme aşamasında yaşanılan en büyük zorlukların başında, söz konusu tezin amacı olan İHA'ların muhtemel yer hedeflerini belirlerken kullanacağı algoritmaların eğitilebilmesi için gerekli hazır veri setlerine ulaşılması olmuştur. Hazır veri setlerine ulaşılamayınca ve/veya yetersiz kalması sebebiyle veri setini elde edilen fotoğraflardan LabelİmageAnalyst ve RoboFlow gibi uygulamalar üzerinden editleyerek algoritmanın eğitmi yolu tercih edilmiştir.

Şekil 3.1: Uydudan Askeri Araç Görüntüleri

Kaynak: (Google Earth, 2022)

İHA'ların yerden açılacak ateşlerden korunabilecek yüksekliklerden uçarken askeri araçların tespit edilebileceği örnek fotolar ancak Google Earth gibi uygulamalardan elde edilebilmiştir (Şekil 3.1, Şekil 3.2)

Şekil 3.2: Uydudan Askeri Araç Görüntüleri

Kaynak: (Google Earth, 2022)

3.2 Roboflow Üzerinden Set Eğitimi

Çalışma dosyalarında biriktirilen görüntülerden LabelİmageAve RoboFlow uygulamaları ile veri setleri hazırlamak için fotoğraflar işlenmiştir (Şekil 3.3 -Şekil 3.4).

İşlenen görüntüler ile Train-Test-Valid için uygun sayıda ve oranda ayrılan resimler ile eğitim için kullanılacak veri seti oluşturuldu (Şekil 3.5).

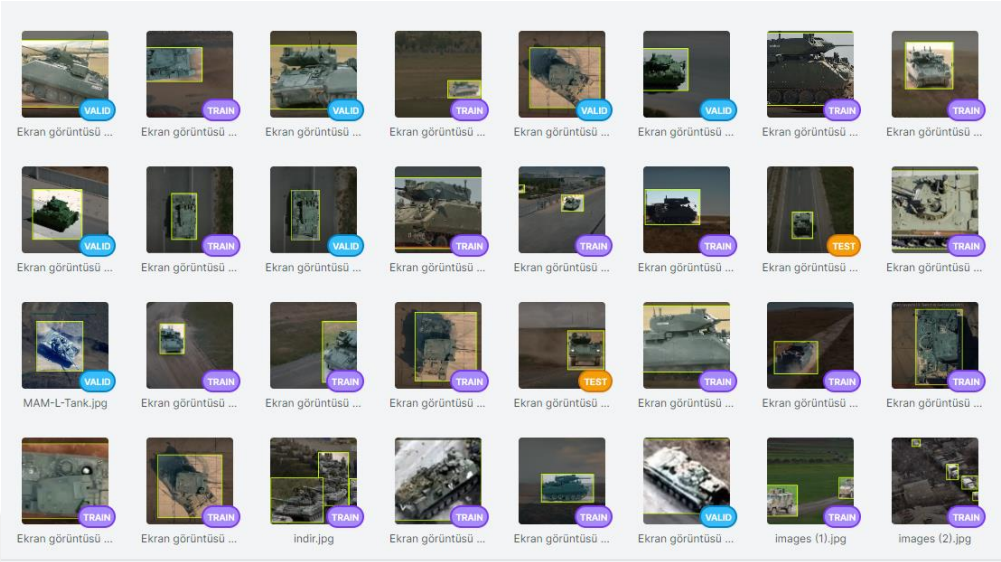
Şekil 3.3: İşlenen Görüntü Örneği

Kaynak: (Kişisel Arşiv, 2022)

Şekil 3.4: İşlenen Görüntü Örneği

Kaynak: (Kişisel Arşiv, 2022)

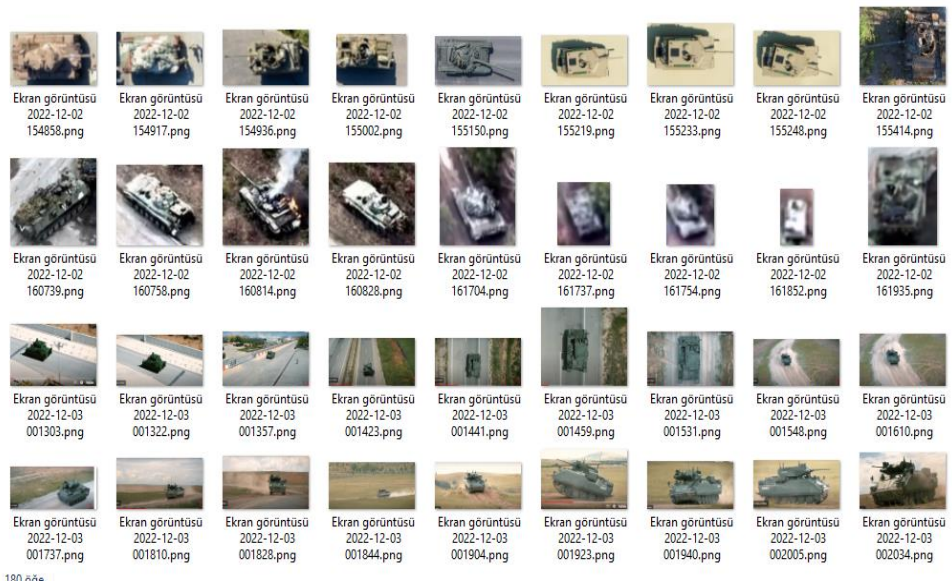
Kullanılan kodlar, veri setleri ve kullanılan fotolar ekler bölümünde sunulacaktır.



Şekil 3.5: İşlenen Görüntüler ile Train-Test-Valid için Oluşturulan Dosya

Kaynak: (Kişisel Arşiv, 2022)

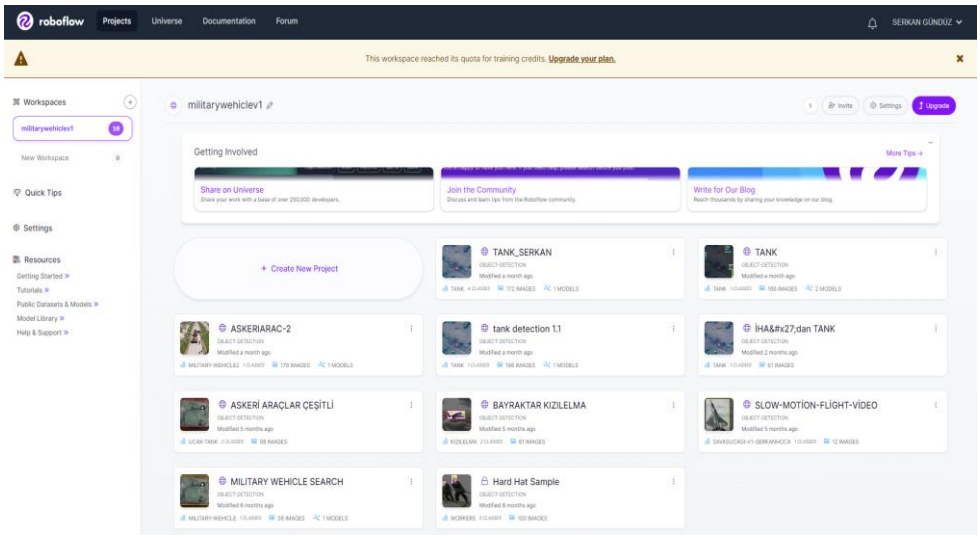
Gerçekleştirilen çalışmanın iş akış diyagramı Şekil 3.6'da verilmiştir. Çalışmada oluşturulan veri setinden yararlanılarak modelin eğitim ve test işlemi gerçekleştirilmiştir.



Şekil 3.6: Çalışmada Kullanılan Askeri Araç Fotoları

Kaynak: (Kişisel Arşiv, 2022)

Label Images benzeri bir program olan Roboflow kısmi olarak ücretsiz online uygulama sayfasıdır. Burada kullanıcı kendi klasörünün içerisindeki fotoğrafları işleyerek test ve eğitim verileri olarak ayırıp eğitilmiş bir veri seti oluşturabilir.



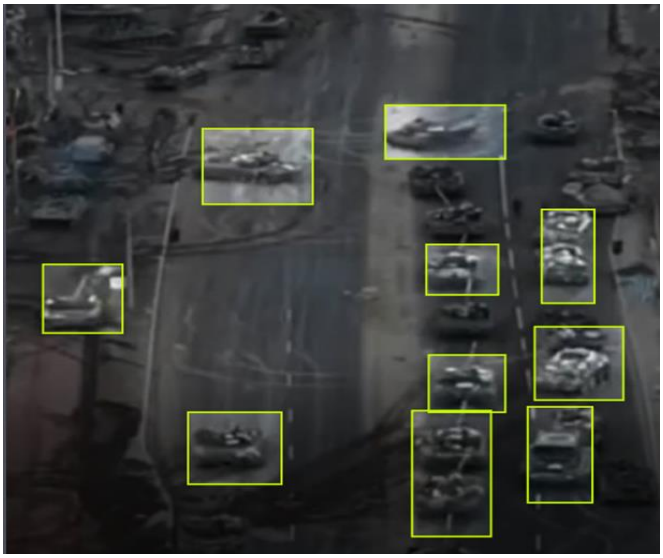
Şekil 3.7: Kişisel Roboflow Sayfası

Kaynak: (Kişisel Arşiv, 2022)

Python kodları kullanılması gerekmeden sadece sürükle bırak işaretle yöntemi ile yüklü kütüphaneleri üzerinden eğitim setini oluşturur ve kullanıcıya bir anahtar kod vererek nesne yakalama işlemi sırasında kullanıma olanak sağlar. Hedef tespit edildiğinde GPS bilgisi ile hedefin konumu belirlenmiştir. Böylelikle hedeflerin hızlı bir şekilde, tehlike oluşturmadan imhası mümkün olacaktır.

3.3 Hedef Yakalama Çalışmaları

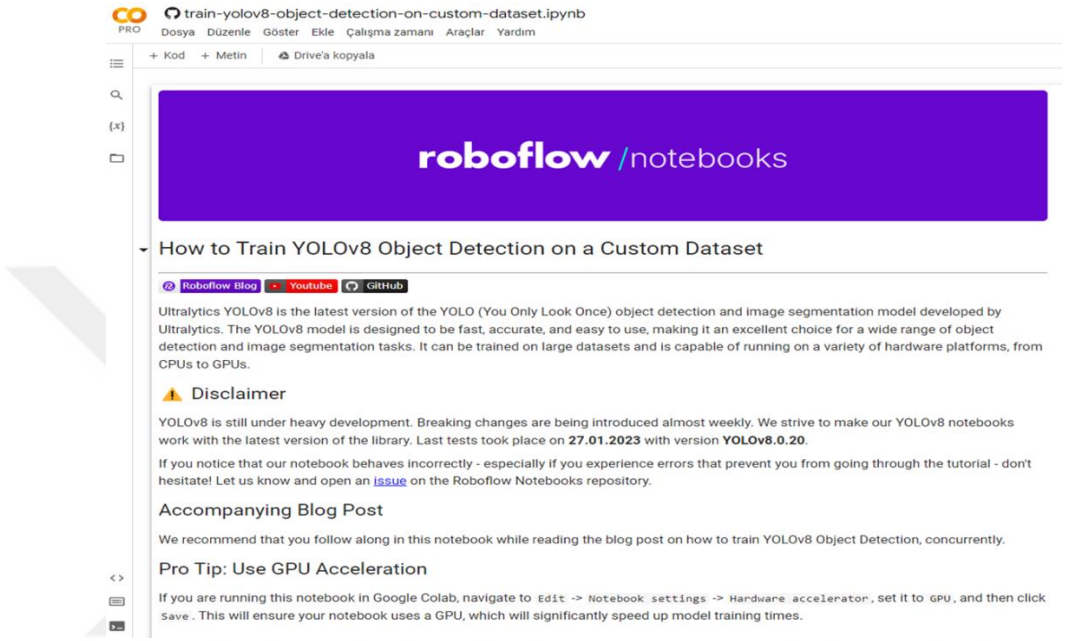
Bu çalışma sırasında Google Colab üzerinden YOLOv7 ve YOLOv8 kullanılarak, videolardan hedef yakalama çalışmaları yapıldı.



Şekil 3.8: İşlenen Görüntüler İle Yakalanan Askeri Araçlar

Kaynak: (Kişisel Arşiv, 2022)

Bu çalışmaların sonucunda elde edilen örnek sonuçlarda Şekil 3.8 görülebileceği gibi askeri araçların seçilebildiği ve çerçeve içerisine alınarak belirgin hale geldiği tespit edilmektedir. Görüntü üzerinden kolayca anlaşılabileceği gibi sahadaki hedef görüntülerin tamamı tespit edilememektedir. Bu da ağırlık dosyasının zenginleştirilmesi gerektiğini, ışığın yetersiz olduğunu, benzer şekillerin aynı sahnede ayırt edilmesi zor bir konumda olduğunu göstermektedir.



Şekil 3.9: Roboflow Ana Sayfası Kişisel Bilgisayar Ekran Görüntüsü

Kaynak: (Kişisel Arşiv, 2022)

Pycharm veya farklı editörler üzerinden tüm kodların yazılması yerine Google Colab tercih edilmesinin sebepleri olarak: Kişisel bilgisayarlardaki ekran kartının yetersizliği sebebiyle videolar üzerinden görüntü akışı son derece yavaş olmaktadır. Adeta ağır çekim şeklinde videolar ilerlemektedir. Google Colab ise kullanıcısına kendi bilgisayarları, sürücüleri üzerinden geçici olarak sanal ekran kartları sağlayarak görüntünün akış hızında düşüş yaşanmadan sonuç alma imkanı sağlamaktadır.

YOLOv7 ve YOLOv8 sadece görüntü işleme için daha özel olarak Object detection Nesne Tespiti yapmak için oluşturulmuş kütüphaneler bütünüdür. Kendisine ait 80 e yakın nesneyi tanıyabilen ağırlık dosyalarına sahiptir. Ancak Askeri Araçlar ile ilgili hazır ağırlık dosyaları henüz çalışılmamıştır. Roboflow kullanarak oluşturduğumuz kendi eğitim setimizden Yolo ve Google Colab kullanarak tez çalışmasında yararlanılacak ağırlık dosyaları oluşturdu. Yolo çalışmaya nesne tespiti özelinde sahip olduğu kütüphaneler yardımı ile görüntü

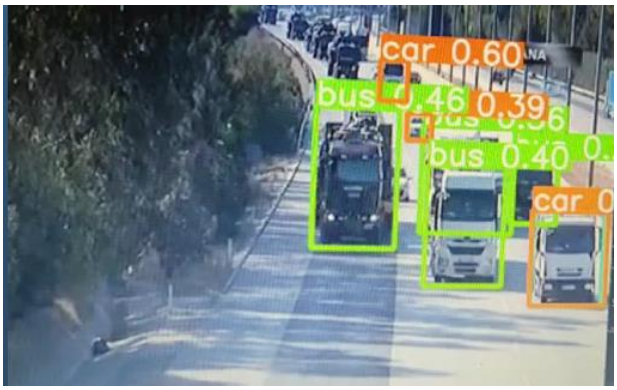
yakalama aşamasında üst düzey programlama bilgisine sahip olunmasa bile askeri araçların yakalanabileceği ağırlık dosyalarını oluşturmada kolaylık sağlamıştır.

Çalışma sürecinde gözlemlenen sorunların başında yolonun hazır ağırlık dosyalarının yetersizliği görülmüştür. Uygulamada askeri araçları tanımlamaya yönelik çalışmalar henüz mevcut değildir. Tez süresince yapılan çalışmaların sonucu olan ağırlık dosyaları kullanıldığında ise hedef nesnelerin bir kısmının yakalanamadığı görülmüştür. Görüntüler çok yukarıdan çekildiğinde askeri araçlar ile ev çatılarının karıştırıldığı ve bazı binalarında seçildiği tespit edilmiştir.

3.4 Pycharm Üzerinde Kullanılabilir Program

Pycharm Üzerinden kullanılabilir program oluşturulmaya çalışıldı.

```
from ultralytics import YOLO

from PIL import Image

model = YOLO('yolov8n.pt')  #YOLO'nun hazır ağırlık dosyası kullanılıyor

# infer on a local image

#print(model.predict("pazar.jpeg", confidence=40, overlap=30).json())

# visualize your prediction

#model.predict("tankfoto.jpg", confidence=40,
overlap=30).save("prediction.jpg")

#model.predict("ZMA.mp4", confidence=40,
overlap=30).save("prediction.mp4")

sonuc = model.predict(source="hamkonvoy.mp4", show=True)

# infer on an image hosted elsewhere

# print(model.predict("URL_OF_YOUR_IMAGE", hosted=True,
confidence=40, overlap=30).json())
```



Şekil 3.10: YOLO Versiyonları ile Tespit Örneği

Kaynak: (Kişisel Arşiv, 2022)

3.5 HSV(Renk Tonu Doygunluğu Değeri) ile Hedef Yakalama Çalışması

HSV(Renk Tonu Doygunluğu Değeri), HSV (Hue, Saturation, Value) veya HSB (Hue, Saturation, Brightness) renk uzayı, renkleri sırasıyla renk özü, doygunluk ve parlaklık olarak tanımlar. Renk özü, rengin baskın dalga uzunluğunu belirler, örneğin sarı, mavi, yeşil, vb. Açısal bir değerdir 0° – 360°, bazı uygulamalarda ise 0-100 arası olağanlaştırılır. Çalışmada HSV(Renk Tonu Doygunluğu Değeri) kullanılarak nesne tespiti çalışması yapılmıştır.

HSV(Renk Tonu Doygunluğu Değeri) kullanarak askeri araç tespitinde kullanılan kodlar şu şekildedir:

"Kodların orijinal çıktıları ekte sunulmuştur."

import cv2 # cv2 isimli kütüphanemiz OpenCV kütüphanesidir.

import numpy as np # Numpy kütüphanesini belirli yerlerde rastgele fotoğraflar oluşturmak için ekliyorum.

from collections import deque

import time # Time kütüphanesini zamanı gelince video işlemlerinde kullanacağımız için ekliyorum.

import matplotlib.pyplot as plt

 # nesne merkezini depolayacak veri tipi

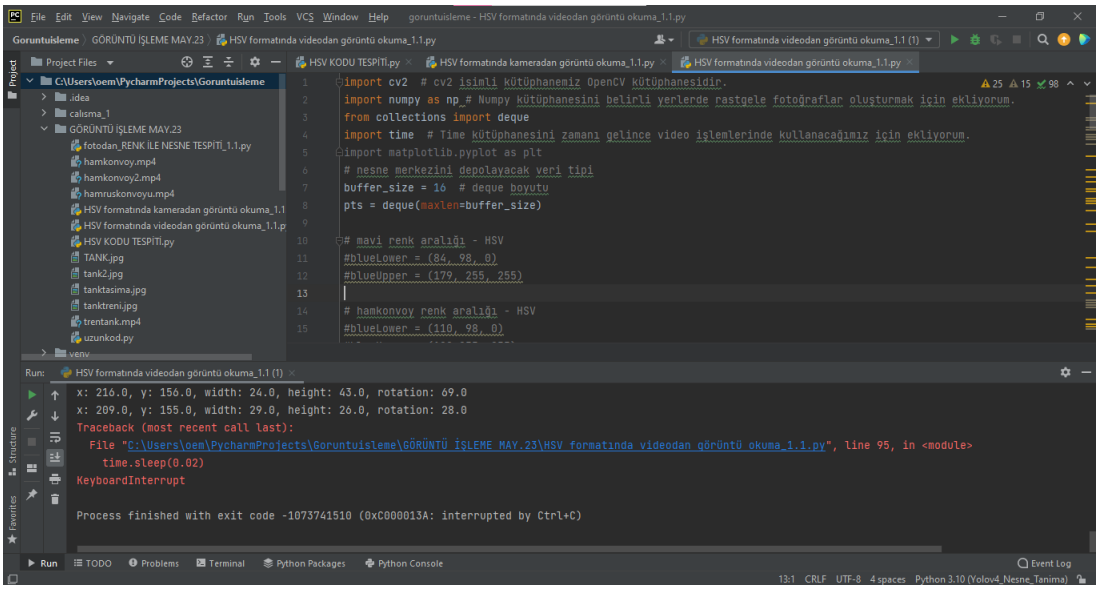
buffer_size = 16 # deque boyutu

pts = deque(maxlen=buffer_size)

```
# mavi renk aralığı - HSV
#blueLower = (84, 98, 0)
#blueUpper = (179, 255, 255)
# hamkonvoy renk aralığı - HSV
#blueLower = (110, 98, 0)
#blueUpper = (120,255, 255)
# tanktreni renk aralığı - HSV
#blueLower = (11, 98, 0)
#blueUpper = (14,255, 255)
#hamkonvoy2 renk aralığı - HSV
blueLower = (3, 80, 10)
blueUpper = (25,255, 255)
# "VideoCapture()" fonksiyonu içine yazılan adresteki videoyu içeri aktarır ve bir değişkene eşitler
cap = cv2.VideoCapture("hamkonvoy2.mp4")
while True:
    success, imgOriginal = cap.read()
    if success:
        # detayi azaltip noise azaltma
        blurred = cv2.GaussianBlur(imgOriginal, (11, 11), 0)
        # hsv
        hsv = cv2.cvtColor(blurred, cv2.COLOR_BGR2HSV)
# "imshow()" fonksiyonu ile pencereyi görselleştiriyoruz
        cv2.imshow("HSV Image", hsv)
        # mavi icin maske
        mask = cv2.inRange(hsv, blueLower, blueUpper)
```

```
            cv2.imshow("mask Image", mask)
            # maskenin etrafinda kalan gurultuleri sil
            mask = cv2.erode(mask, None, iterations=1)
            mask = cv2.dilate(mask, None, iterations=1)
            cv2.imshow("mask+erode-dilate", mask)
            # contour
            (contours, _) = cv2.findContours(mask.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
            center = None
            if len(contours) > 0:
                # en buyuk konturu al
                c = max(contours, key=cv2.contourArea)
                # dikdortgene cevir
                rect = cv2.minAreaRect(c)
                ((x, y), (width, height), rotation) = rect
                s = "x: {}, y: {}, width: {}, height: {}, rotation:
{}".format(np.round(x),
                                                        np.round(y),
                                                        np.round(width),
                                                        np.round(height),
                                                        np.round(rotation))
                print(s)
                box = cv2.boxPoints(rect)
                box = np.uint64(box)
                # moment - goruntunun merkezini bulmamiza yarayan yapi
                M = cv2.moments(c)
                center = (int(M["m10"] / M["m00"]),
```

```
                int(M["m01"] / M["m00"]))
            # contour'u cizdir
            cv2.drawContours(imgOriginal, [box], 0, (0, 255, 255), 2)
            # merkeze nokta ciz
            cv2.circle(imgOriginal, center, 5, (255, 0, 255), -1)
            # bilgileri ekrana yazdir
            cv2.putText(imgOriginal, s, (20, 20),
cv2.FONT_HERSHEY_COMPLEX_SMALL, 0.4, (0, 0, 0), 1)
            # deque (nokta takip)  yeşil çizgiler çiziyor
          #pts.appendleft(center)
            #for i in range(1, len(pts)):
                #if pts[i - 1] is None or pts[i] is None: continue
                #cv2.line(imgOriginal, pts[i - 1], pts[i], (0, 255, 0), 3)
        # Time modülündeki "sleep()" fonksiyonunu kullanarak pencerelerin
        # daha yavaş görselleştirilmesini sağlıyoruz.
        # Bunu yapmaz isek videomuz çok hızlanır.
            time.sleep(0.02)
            cv2.imshow("SERKANIN ASKERI ARAC TESPITI", imgOriginal)
            if cv2.waitKey(1) & 0xFF == ord("q"):
                cv2.destroyAllWindows()
                break
```



Şekil 3.11: HSV(Renk Tonu Doygunluğu Değeri) ile Tespit Kodları

Kaynak: (Kişisel Arşiv, 2022)

Şekil 3.12: HSV (Renk Tonu Doygunluğu Değeri) ile Tespit Örneği

Kaynak: (Kişisel Arşiv, 2022)

Şekil 3.13: HSV (Renk Tonu Doygunluğu Değeri) ile Tespit Örneği

Kaynak: (Kişisel Arşiv, 2022)

3.6 İHA Montajı

İHA üzerinden montaj çalışmaları yapıldı. Çalışmanın başından sonuna kadar özgün bir çalışmanın ürünü olacak şekilde sıfırdan bir İHA üretilmeye çalışıldı. Bazı parçaları 3D yazıcıda basılan İHA'nın ağırlıklı olarak kullanılan parçaları satın alma yolu ile temin edilmeye çalışıldı.

Şekil 3.14: İHA Montaj Aşamaları

Kaynak: (Kişisel Arşiv, 2022)

4. BULGULAR VE KARŞILAŞTIRMA SONUÇLARI

İHA'ların kullanımı her geçen gün artmakta ve İHA'lar popülerlik kazanmaktadır. Günlük hayatta birçok sektörde, yeni bir trendin başladığını söylemek yanlış olmaz. Son zamanlarda yapay zekâ olarak adlandırılan teknolojiler ile bazı çalışmalar yapılmaktadır. Bu teknik sayesinde bağımsız olarak iş yapmak otonom olarak daha kolay, daha doğru ve daha güvenli olacaktır.

Bu çalışma ışığında üretilen ürünün; savunma sanayi, takip, arama kurtarma, tarımsal amaçlı, hobi vb. nesne algılama ve izleme ile ilgili birçok çalışma alanı vardır, ancak bu sistem tarama alanını ve hedef tanımını değiştirme kolaylığı ile hedefi daha spesifik hale getirme yeteneği sağlar. Bu özelliklere ek olarak tüm işleri insan yardımı olmadan otonom olarak yapmak son kullanıcıya kullanım kolaylığı sağlar. Bazı durumlarda, görüntü işleme, istenen görüntünün algılanmasını hedefin gözden kaçırılmasını önleyecektir.

Nesne takibi yöntemleri temel olarak nokta tabanlı, çekirdek tabanlı ve siluet tabanlı yöntemler olarak 3 kategoriye ayrılabilir. Nokta takip yönteminde temel amaç nesnenin video çerçevesi içinde tespit edilmesi ve bir önceki çerçevede kullanılan nokta benzerliklerin hesaplamasıdır. Bu yöntemin en büyük avantajı doğrusal olmayan ve çoklu dağılıma sahip sistemlerde çalışabilmesidir Fan L.ve diğ. (2016). Çekirdek tabanlı yöntemlerde nesnenin şeklinden ziyade kullanılan geometrik şeklin içerisinde bulunan nesne bilgilerinin çıkarılması yeterli olabilmektedir. Bu yöntemin temel amacı nesneyi tanımlayacak kenar bilgisi ya da şekil bilgisi çıkartılarak sonraki imgelerde bu bilgiyi aramaktır. Çekirdek ve siluet tabanlı yöntemler kıyaslandığında, çekirdek tabanlı yöntemlerin daha düşük işlem zamanına ve daha yüksek başarı oranlarına sahip oldukları görülmektedir. Bu sebepten dolayı çalışmalarda çekirdek tabanlı yöntemler geniş bir kullanım alanına sahiptir. Nokta tabanlı yöntemler diğer yöntemlere oranla daha düşük işlem zamanına sahip olmakla birlikte daha düşük başarı oranına sahiptirler.

Literatürde nesne takibi için yaygın olarak kullanılan yöntemler Çizelge 4.1'de gösterilmiştir Tiwari, M. ve Singhai, R. (2017)

Çizelge 4.1: Literatürde Nesne Takibi İçin Yaygın Kullanılan Yöntemler

Nesne Takip Yöntemi		Kullanımı	İşlem Zamanı	Başarı Oranı
Nokta tabanlı	Kalman filtresi	Kalman filtre algoritmaları	Düşük-Orta	Orta
	Parçacık filtresi	Öz yinelemeli bayes filtresi	Orta-Yüksek	Orta
	Çoklu hypothesis takip	MHT algoritması	Düşük	Düşük-Orta
Çekirdek tabanlı	Temel şablon uydurma	Video içinde uyum aram algoritmaları	Düşük-orta	Düşük
	Destek vektör makinesi (dvm)	Görüntü içindeki piksellerin nesne ve arka plan olarak sınıflandırması	Orta	Orta
	Uydurma tabanlı sınıflandırma	Şekil içindeki piksel yoğunluklara bakılır	Orta	Orta-Yüksek
Siluet tabanlı	Kenar kesiştirme	Gradient descent algoritmaları	Orta	Orta-Yüksek
	Şekil uydurma	Hough dönüşümü	Yüksek	Yüksek

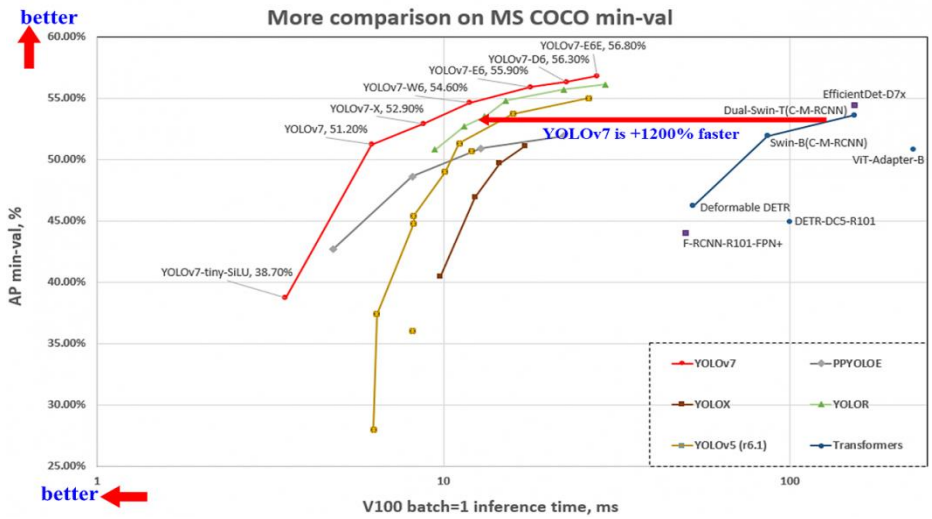
Kaynak: (Hanbay, K. ve Üzen, H. 2017: s. 44)

Nesne takibinde kullanılan fakat yaygın olmayan diğer yollara, bölge tabanlı izleme (Deori ve Meitei, 2014), kontur tabanlı izleme (Jacob and Anitha, 2012) ve sınır tabanlı izleme (Ojha ve Sakhare, 2015) örnek verilebilir.

Nesne takibinde uygulamanın içeriğine göre kullanılabilecek veri setleri değişmektedir. ImageNet (Deng vd., 2009), WordNet tabanlı yaklaşık 50 milyon etiketlenmiş temiz görüntü içeren veri kümesidir. Hiyerarşinin her bir düğümü binlerce resimle tasvir edilmektedir. ImageNet ölçek ve çeşitlilik açısından mevcut görüntü veri setlerinden çok daha büyüktür. Nesne tanıma, otomatik nesne kümeleme, görüntü sınıflandırma gibi uygulamalarda başarılı sonuçlar vermektedir. Caltech101/256 (Griffin vd., 2007), MSRC (Shotton vd., 2006), PASCAL (Everingham vd., 2008), TinyImage (Torralba vd., 2008) küçük görüntülere ait veri setleri içermektedir. Buradaki görüntüler WordNet veri tabanında da saklanmaktadır. Wordnet'ten gelen anlamsal bilgilerle etiketleme gürültüsünün etkilerini en aza indirmek için en yakın komşu yöntemiyle birlikte kullanılır. Nesne sınıflandırması uygulamalarında özellikle insan gibi yaygın olarak kullanılan nesnelerde başarılı sonuçlar alınmaktadır. ESP (Ahn ve Dabbish, 2004), çevrimiçi oyun aracılığı ile kullanıcılar tarafından etiketlenen resimlerden oluşmaktadır. İnsanlar oyun sırasında onlar için anlamlı etiketler ile görüntülerin içeriklerini belirlerler. Görüntü etiketleme sorununu ele alarak eğlenceli bir şekilde webde bulunan tüm görüntüleri etiketleyebileceklerini tahmin etmektedirler. Çok az bir miktarda etiket ve görüntü kümesi haricinde halka açık değildir. Lotus Hill ve LabelMe (Russell vd., 2008; Yao

vd., 2007), yaklaşık 200 kategoriye sahiptir. Nesnelere ait etiket, konum gibi detaylı özellikleri içermektedir. ImageNet'e kıyasla daha az kategori ve veri setine sahiptir. LabelMe çizim arayüzü sayesinde etiketlenen görüntü özelliklerinin web üzerinde anında paylaşılmasına olanak tanır. Bu şekilde birçok platformdan etiketli görüntü kümeleri toplanmış olur. Lotus Hill, diğer veri etiketleme araçlarından daha farklı bir yöntem izleyerek 3 seviyede bilgiyi gruplamayı amaçlamıştır. Bu araç sayesinde genel amaçlar için kullanılabilecek 636.748'den fazla ek açıklamalı görüntü ve video karesinden oluşan bir veri tabanı oluşturmuşlardır. MS-COCO (Lin vd., 2014), PASCAL-VOC (Everingham vd., 2010), KITTI (Geiger vd., 2012) silah, tabanca gibi küçük nesne algılama uygulamalarında kullanılan popüler veri kümelerindendir. MS-COCO nesne algılamadaki sahne sorununun ele alınmasıyla yeniden oluşturulan bir veri kümesidir. 91 kategori, yaklaşık 328 bin görüntü ve 2,5 milyon etiketli örnekle nesne, kategori algılama, kümeleme sınıflandırma uygulamalarında kullanılmaktadır. KITTI, otonom bir sürüş platformu kullanarak oluşturulmuş, görüş görevi içeren, nesne algılama uygulamalarında kullanılan veri kümesidir. Veri kümesi, 3B sınırlayıcı kutularla açıklanmış 7481 eğitim görüntüsü içermektedir Tan ve diğ. (2021).

Nesne tanıma ve takip yöntemlerine farklı bir bakış açısı getiren YOLO versiyonları bünyelerinde yer alan ve sürekli geliştirilen hazır kütüphaneleri üzerinden kullanıcıya hız ve pratiklik sağlamaktadır. Yolov7 ve akabinde geliştirilen Yolov8 ile görüntü tanıma çalışmalarında yüksek düzeyde bir başarı elde edilmesi amaçlanmıştır.

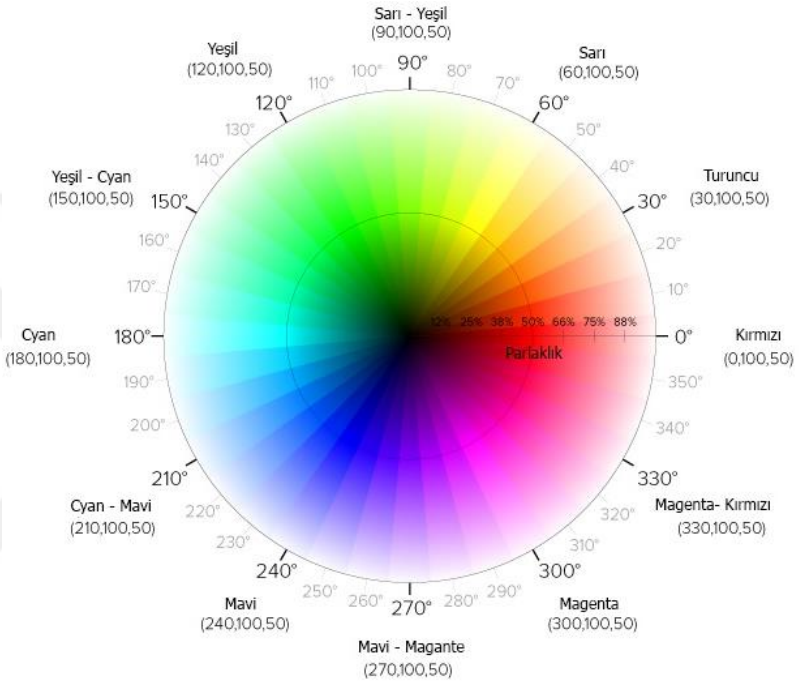


Şekil 4.1: Performans karşılaştırması YOLOv7 - YOLOR - YOLOX - YOLOv5

Kaynak: (Cornell University, 2019)

Bunun yerine renk kodları ile işlem denendiğinde daha yüksek oranda bir başarı elde edilmiştir. Burada yaşanan temel sorunda askeri araçların sabit bir renk kodu olmaması video üzerinden taramalarda da video kalitesine bağlı olarak renk kodlarının değiştiği görülmüştür.

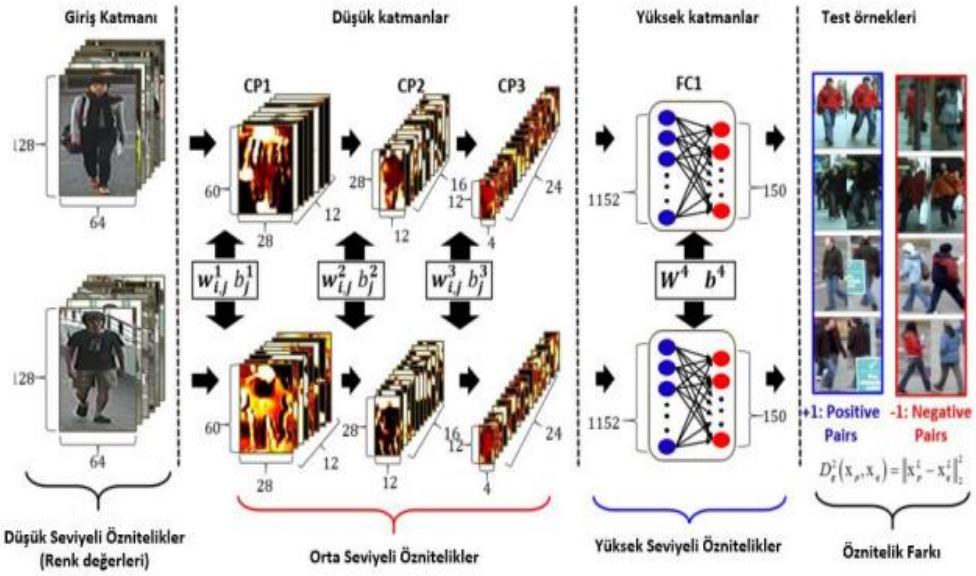
Bu çalışma, sektörde önde gelen sivil ve resmi firmalar ile İHA konusunda deneyimli kurumların kendilerine ait veri havuzlarını paylaşması ile daha çok sayıda fotoğraf kullanılarak eğitilebilen ağırlık dosyalarının etkisi ile geliştirilebilir.



Şekil 4.2: HSV(Renk Tonu Doygunluğu Değeri) Uzaydaki Renk Kodları Aralığı

Kaynak: (Cornell University, 2019)

Nesne içeren görüntü parçalarının RGB renk kanalları ile elde edilen düşük seviye öznitelikleri, geliştirilen yapay sinir ağı katmanlarından geçirilerek sınıfları ayıracak güçlü öznitelikler elde edilmiştir (Şekil 4.3). Bu öznitelikler kullanılarak nesnelerin tespit ve takibi yapılmıştır. Geliştirilen yöntem birçok zorlu veri kümeleri ile test edilmiştir. Testler sonucunda çalışmadaki diğer yöntemlere kıyasla belirgin bir performans artışı gösterilmiştir.



Şekil 4.3: Discriminative Deep Appearance Modelinin Genel Ağ Yapısı

Kaynak : (Bae ve diğ., 2017)

Derin öğrenme çalışmalarında kullanılan birden fazla kütüphane ve yazılım bulunmaktadır. Bunlara ait bilgiler Çizelge 4.2'de gösterilmiştir. Bu kütüphanelerden GPU desteği veren Theano, TensorFlow, Caffe ve Torch kullanılarak bazı veri kümeleri ve modeller üzerinden çalışma zamanı performansı karşılaştırıldığında (Yuret, 2016), Theano'nun en hızlı çalıştığı ardından Caffe, Torch'un ve en yavaş çalışanın ise TensorFlow kütüphanesi olduğu görülmüştür. Github yıldız sayılarına bakıldığında her açıdan, TensorFlow liderdir. Keras, Caffe, PyTorch ilk beş sırada yer almaktadır Tan ve diğ.(2021).

Çizelge 4.2: Derin Öğrenmede Kullanılan Kütüphaneler

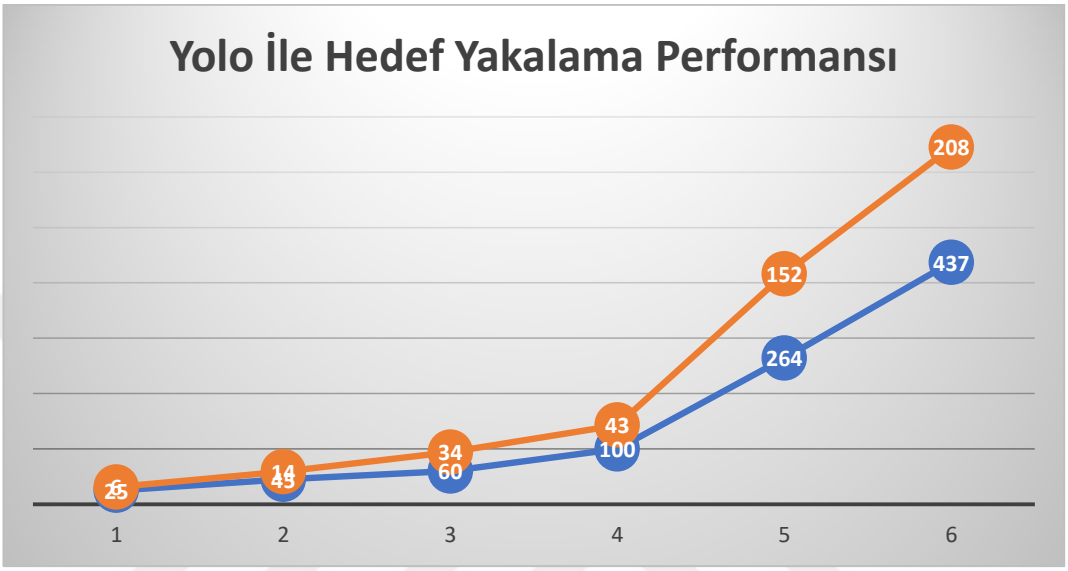
Kütüphane	Dil	Yıl	Açıklama	GitHub Yıldız Sayısı	GPU Çalışma Zamanı Performansı
PyTorch (Collobert vd., 2017)	Python	2017	GPU desteği, hız ve esneklik	45k	2.88
Tensorflow (TensorFlow, 2020)	Python,C++	2020	Otomatik resim yazısı oluşturma yazılımı, tek bir API ile birden fazla GPU ve CPU'ya dağıtma olanağı	152k	5.57
Keras (Keras: The Python Deep Learning API, 2020)	Python	2020	Aktivasyon fonksiyonları ve optimize ediciler	50.4k	-
Digits (NVIDIA DIGITS, 2017)	C++	2017	Çoklu GPU üzerinde verimli, tamamen etkileşimli	4k	-
Theano (Theano Development Team, 2016)	Python	2016	CPU ve GPU üzerinde verimli bir şekilde çalışır, matematiksel hesaplamalar kolay	9.3k	1.40
Caffe (Jia vd., 2014)	Python, C++	2014	Görüntü sınıflandırma ve görüntü bölümleme, GPU desteği	31.2k	2.45
Deep learning 4j (DeepLearning4j, 2020)	Java, C, C++, Python	2020	Görselleştirme aracı	11.9k	-

Kaynak: (Hanbay, K. ve Üzen, H. 2017: s. 44)

4.1 YOLO ve HSV açılımı

Yolo versiyonları gelişme gösterdikçe yeni sürümleri ortaya çıktıkça ve bu tezin konusu özelinde uygun ve yeterli sayıda hava fotoğrafları ile eğitilen ağırlık dosyaları kullanımıyla görüntü yakalama işlemleri çok daha verimli hale gelecektir.

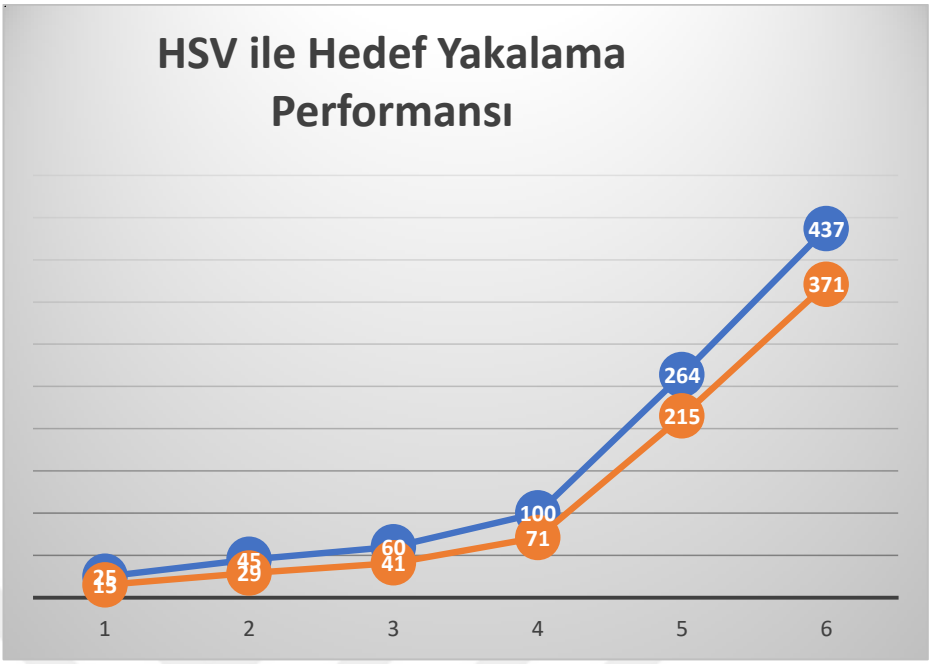
Çizelge 4.3: Yolo ile Hedef Yakalama Performansı



Hedef Sayısı	25	45	60	100	264	437
Yakalanan Hedef Sayısı	6	14	34	43	152	208
Yüzdeler	24%	31%	56%	43%	57%	47%

Çizelge 4.3 de YOLO ile hedef yakalamada gösterilen performans görülebilir. Yolo yapay zeka uygulaması kullanılarak, hedef sayısı ile yakalanan hedef sayısının karşılaştırılması sonucunda 25 hedef bulunan videoda 6 hedef yakalanabilmiş (%24 başarı), 45 hedef bulunan videoda 14 hedef yakalanabilmiş (%31 başarı), 60 hedef bulunan videoda 34 hedef yakalanabilmiş (%56 başarı), 100 hedef bulunan videoda 43 hedef yakalanabilmiş (%43 başarı), 264 hedef bulunan videoda 152 hedef yakalanabilmiş (%57 başarı), 437 hedef bulunan videoda ise 208 hedef yakalanabilmiştir (%47 başarı). Bu değerlerden yola çıkarak hedef sayısı arttıkça Yolo yapay zeka uygulaması ile görüntü tespitinin anlamlı bir artış göstermediği gözlemlenmiştir.

Çizelge 4.4: HSV(Renk Tonu Doygunluğu Değeri) İle Hedef Yakalama Performansı



Hedef Sayısı	25	45	60	100	264	437
Yakalanan Hedef Sayısı	15	29	41	71	215	371
Yüzdeler	60%	64%	68%	71%	81%	84%

HSV (Renk Tonu Doygunluğu Değeri) kullanılarak, hedef sayısı ile yakalanan hedef sayısının karşılaştırılması sonucunda 25 hedef bulunan videoda 15 hedef yakalanabilmiş (%60 başarı), 45 hedef bulunan videoda 29 hedef yakalanabilmiş (%64 başarı), 60 hedef bulunan videoda 41 hedef yakalanabilmiş (%68 başarı), 100 hedef bulunan videoda 71 hedef yakalanabilmiş (%71 başarı), 264 hedef bulunan videoda 215 hedef yakalanabilmiş (%55 başarı), 437 hedef bulunan videoda ise 371 hedef yakalanabilmiştir (%84 başarı) . Bu değerlerden yola çıkarak hedef sayısı arttıkça HSV (Renk Tonu Doygunluğu Değeri) ile görüntü tespitinin doğru orantılı arttığı gözlemlenmiştir. Çizelge 4.4 de HSV (Renk Tonu Doygunluğu Değeri) ile hedef yakalama performansı görülebilir.

5. SONUÇ

Bu çalışmada derin öğrenme yöntemleri kullanılarak nesne tespiti için yapılan uygulamalar incelenmiştir. Bunun yanısıra görüntü işleme donanımlarına sahip bir insansız hava aracı ve özgün bir ağırlık dosyası üretilmiştir. Hava aracı kamerası ve uçuş ekipmanları ile görüntü yakalamaya elverişli bir duruma getirilmiştir. Hazırlanan ağırlık dosyaları ile son kullanıcının rahatlıkla programları içerisine adapte edebileceği verimli bir materyal elde edilmiştir. Çalışmada kullanılan mimariler, ağlar, kütüphaneler, algoritmalar, veri setleri ve modeller hakkında bilgiler sunulmuştur. Oluşturulacak olan model, nesneye ait fotoğraflar, görüntüler kullanılarak eğitilebilir. İncelenen çalışmalarda nesne tespiti aşamasında en çok YOLO, Faster R-CNN ve SSD algoritmaları kullanılmıştır. Modeli eğitirken verinin fazla olması başarıyı arttıracaktır fakat işlem zamanını da arttırdığı için yavaşlamaya neden olacaktır. Özellikle R-CNN ve Fast R-CNN'nin çalışma yöntemine bakıldığında ilk aşamada nesneleri sınıflandırıp ardından konvolüsyonel sinir ağı uyguladığı için yavaş çalışmaktadırlar. Bu algoritmalardaki hız sorununu çözmek için ise Faster R-CNN kullanılmaya başlanmıştır. SSD ve Faster R-CNN karşılaştırıldığında ise SSD, tek bir sinir ağı kullanarak nesne tespiti yaptığı için Faster R-CNN'den daha hızlı çalışmaktadır. Faster R-CNN'nin başarı oranı SSD algoritmasına göre daha yüksektir. Hem yüksek başarı oranı hem de hızın önemli olduğu uygulamalarda ise son yıllarda popüler olan YOLO kullanılmaktadır. Nesne sınıflandırma aşamasında tespit edilecek nesnenin özniteliklerine göre sınıflandırma yöntemi de değişiklik göstermektedir. Renk ve doku tabanlı yöntemlerin daha yüksek başarı oranı sağladığı görülmüştür. Nesne takibi sırasında ise Kalman Filtresi yöntemi daha sık kullanılmıştır. İzlenecek nesneye göre uygulanacak sinir ağı modeli de değişiklik gösterecektir. Çoklu nesne izleme yapılacaksa Deep-Sort, Deep, Multicut algoritmaları, uygunsuz görüntülerin tespiti için de ResNet-50 kullanılması önerilmiştir.

Son yıllarda geliştirilen derin öğrenme yöntemlerinin eğitim süreleri uzun olmasına rağmen test aşamasında elde edilen başarı oranları derin öğrenme

yöntemlerine olan güveni arttırmıştır. Fakat basit problemlerde işlem zamanı ve hesapsal karmaşıklığı az olan algoritmaların kullanılması çok daha uygun olacaktır. Varılan bir diğer sonuç ise HSV renk kodları ile nesne tespit yöntemlerinin bulunmasının nesne takip algoritmalarına farklı bir perspektif kazandırdığıdır.

YOLO versiyonları ve HSV(Renk Tonu Doygunluğu Değeri) uygulamaları ile görüntü yakalama çalışmalarında, çok net olarak görülen HSV(Renk Tonu Doygunluğu Değeri) ile görüntü yakalamanın daha verimli olduğu yönündedir.

Yolo ile hedef yakalama çalışmasında hedef sayısı artıkça tespit sayısı anlamlı şekilde artmamaktadır. HSV Renk Tonu Doygunluğu Değeri (HSV) ile Yapay Zeka uygulaması karşılaştırıltırıldığında renk kodlarını kullanarak elde edilen sonuçların yapay zeka uygulamasına göre daha başarılı olduğu görülmüştür. Ancak burada sistem performansını olumsuz etkileyen faktörlerde gözlemlenmiştir. Bu gözlemler; askeri araç görüntülerinin ideal netlikte ve açıklıkta temel standartlara uygun olmaması, cephe alanında yabancı nesnelerin sıklıkla var olması ve ışık etkisinin zayıf olduğu durumlar olarak belirtilebilir.

Bu çalışmanın başarı seviyesinin yükseltilebilmesi ağırlık dosyalarının çok daha fazla ideal görüntü ile yüksek kapasiteli bilgisayarlar üzerinden yapılacak gelecekteki çalışmalarla mümkündür. Ağırlık dosyalarının daha zengin bir eğitim seti ile eğitilmesi ile başarı oranı daha da yukarılara çekilebilecektir. Karakter tanıma becerisinin arttırılması askeri araç tanıma başarısını doğrudan etkileyecek ve sistemin pratik olarak kullanılabilirliğini arttıracaktır.

KAYNAKLAR

Bae, S. H., & Yoon, K. J. (2017). Confidence-based data association and discriminative deep appearance learning for robust online multi-object tracking. IEEE Trans Pattern Anal Mach Intell 40(3):595–610

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009, June). "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, pp. 248-255, doi: 10.1109/CVPR.2009.5206848.

Fahlstrom, P. G. ve Gleason, T. J. (2012). *Introduction to UAV systems* (4th ed.). United Kingdom: Wiley.

Fan, L., Wang, Z., Cail, B., Tao, C., Zhang, Z., Wang, Y., ... & Zhang, F. (2016, August). A survey on multiple object tracking algorithm. In 2016 IEEE International Conference on Information and Automation (ICIA) (pp. 1855-1862). IEEE. https://doi.org/10.1109/ICInfA.2016.7832121

Geiger, A., Lenz, P., & Urtasun, R. (2012, June). Are we ready for autonomous driving? the kitti vision benchmark suite. In 2012 IEEE conference on computer vision and pattern recognition (pp. 3354-3361). IEEE.

Kaehler, A.&Bradski, G., (2008). Learning OpenCV: Computer vision with the OpenCV library. " O'Reilly Media, Inc."

Keipour, A., Mousaei, M., & Scherer, S. (2021). Alfa: A dataset for uav fault and anomaly detection. *The International Journal of Robotics Research*, 40 (2-3), 515-520.

Kumar, C., Vaddi S. ve Sarkar, A. (2018). A case study on optimal deep learning model for UAVs. *ICLR 2019, Conference International Conference on Learning Representations,* 1-9.

Lin, S., Wang, J., Peng, R., & Yang, W. (2019). Development of an Autonomous Unmanned Aerial Manipulator Based on a Real-Time Oriented-Object Detection Method. Sensors, 19(10), 2396.

Nonami, K., Kendoul, F., Suzuki, S., Wang, W., & Nakazawa, D. (2010). Autonomous flying robots: unmanned aerial vehicles and micro aerial vehicles. Springer Science & Business Media.

Özel, M. A., Baysal, S. S., Şahin, M. (2021). Derin öğrenme algoritması (YOLO) ile dinamik test süresince süspansiyon parçalarında çatlak tespiti. Avrupa Bilim ve Teknoloji Dergisi, (26), 1-5.

Parekh H.S., Thakore D.G., Jaliya U.K. (2014), A survey on object detection and tracking methods, International Journal of Innovative Research in Computer and Communication Engineering, 2.2: 2970-2979

Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

Russell, B. C., Torralba, A., Murphy, K. P., & Freeman, W. T. (2008). LabelMe: a database and web-based tool for image annotation. International journal of computer vision, 77, 157-173.

Sahin, O. ve Ozer, S. (2018, Haziran). IHA Görüntülerinde Nesne Tanıma İçin Geliştirilmiş YOLO Mimarisi. Bilkent Üniversitesi Eğitim Fakültesi Dergisi, 31, 1–15.

Sahin, O. ve Ozer, S. (2021). "Yolodrone: Improved yolo architecture for object detection in drone images," in 2021 44th International Conference on Telecommunications and Signal Processing (TSP), 361–365.

Shotton, J., Winn, J., Rother, C., & Criminisi, A. (2006). Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9 (pp. 1-15). Springer Berlin Heidelberg.

Talu, M. F. (2010, Nisan). Nesne takip yöntemlerinin sınıflandırılması. İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi, 9(18), 45-34.

Tan, F. G., Yüksel, A. S., Aydemir, E., & Ersoy, M. (2021, Kasım). Derin öğrenme teknikleri ile nesne tespiti ve takibi üzerine bir inceleme. Avrupa Bilim ve Teknoloji Dergisi, (25), 159-171.

Tiwari, M., & Singhai, R. (2017). A review of detection and tracking of object from image and video sequences. Int. J. Comput. Intell. Res, 13(5), 745-765.

Torralba, A., Fergus, R., & Freeman, W. T. (2008). 80 million tiny images: A large data set for nonparametric object and scene recognition. IEEE transactions on pattern analysis and machine intelligence, 30(11), 1958-1970.

Valavanis, K. P. (2007). Advances in unmanned aerial vehicles. The Netherlands: Springer.

Valavanis, K. P., Vachtsevanos, G. J. (Eds.). (2015). Handbook of unmanned aerial vehicles (Vol. 1). Dordrecht: Springer Netherlands.

Yao, B., Yang, X., Zhu, S. C. (2007). Introduction to a large-scale general purpose ground truth database: methodology, annotation tool and benchmarks. In Energy Minimization Methods in Computer Vision and Pattern Recognition: 6th International Conference, EMMCVPR 2007, Ezhou, China, August 27-29, 2007. Proceedings 6 (pp. 169-183). Springer Berlin Heidelberg.

Yilmaz, A., Javed, O., Shah, M. (2006). Object tracking: A survey. Acm computing surveys (CSUR), 38(4), 13-es.

İnternet

Cornell University (2019, 24 Aralık). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. 18.03.2023 tarihinde https://arxiv.org/abs/2207.02696 adresinden edinilmiştir.

burakzdd (2022, 5 Mart). Yolo Nedir? Nasıl çalışır? Versiyonlarının özellikleri? 07.06.2023 tarihinde https://burakzdd.medium.com/yolo-621dc8e903aa adresinden edinilmiştir.

bolubeyi (2022, 30 Nisan). Quadrotor nedir. 07.06.2023 tarihinde http:// bolubeyi.net/ quadrotorquadcopter-nedir.html adresinden edinilmiştir.

Arducopter (2022, 3 Şubat). Kanal bağlantıları. 18.03.2023 tarihinde http://www.arducopter.co.uk adresinden edinilmiştir.

defensnews (2011, 18 Şubat). Askeri Araç Fotoları. 22.06.2023 tarihinde https://www.defensenews.com/ adresinden edinilmiştir.

Dronetrest (2023, 20 Nisan). Brushless motors - how they work and what the numbers mean. 07.06.2023 tarihinde https://www.dronetrest.com/t/brushless-motors- adresinden edinilmiştir.

Engineerstoys (2023, 2 Ocak). Kanallı Kumanda Verici ve Alıcı Çeşitleri. 18.03.2023 tarihinde https://www.engineerstoy.com/ adresinden edinilmiştir.

Everingham, M. (2008, 19 Mart). The PASCAL visual object classes challenge 2008 (VOC2008) results. 13.05.2023 tarihinde http://www. pascal-network. org/challenges/VOC/voc2008/year= workshop/index. html. adresinden edinilmiştir.

GeeksforGeeks (2021, 30 Nisan). Erosion and dilation of images using OpenCV in python. 07.06.2023 tarihinde https://www.geeksforgeeks.org/erosion-dilation- images-using-opencv-python/ adresinden edinilmiştir.

Guraysonugur (2021, 14 Şubat). Görüntü işleme notları. 18.03.2023 tarihinde https://guraysonugur.aku.edu.tr/2021/12/21/2021-goruntu-isleme-notlari-morfolojik-islemler-filtreleme-konvolusyon/ adresinden edinilmiştir.

github.com (2022, 14 Şubat). Ultralytics. 07.06.2023 tarihinde <https://github.com/ultralytics/ultralytics< adresinden edinilmiştir.

Howstuffworks (2019, 20 Mart). İHA fotoğrafları. 22.06.2023 tarihinde https://www.howstuffworks.com/ adresinden edinilmiştir.

Holybro (2022, 25 Nisan). Telemetri modülü alıcı-vericileri. 07.06.2023 tarihinde https://holybro.com/ adresinden edinilmiştir.

OpenCV (2022, 6 Mart). Smoothing images. 07.06.2023 tarihinde fhttps://docs.opencv.org/3.4/dc/dd3/tutorial_gausian_median_blur_bilateral_filter. html. adresinden edinilmiştir.

OpenCV (2023, 2 Ocak). Open cv nedir nerelerde kullanılır. 13.05.2023 tarihinde https://opencv.org/about/ adresinden edinilmiştir.

oyuncakhobi (2022, 19 Şubat). Pixhawk Bağlantıları Nasıl Yapılır? 18.03.2023 tarihinde https://blog.oyuncakhobi.com/pixhawk-hex-the-cube-baglantilari-nasil-yapilir/ adresinden edinilmiştir.

Pisupply (2019, 5 Ocak). İHA teknolojileri. 18.03.2023 tarihinde https://www.pisupply.com/ adresinden edinilmiştir.

Raspberrypi (2019, 4Şubat). Rasperypi mikroişlemci. 07.06.2023 tarihinde https://www.raspberrypi.com/ adresinden edinilmiştir.

Robotistan (2022, 11 Mayıs). Lipo pil çeşitleri. 22.06.2023 tarihinde https://www.robotistan.com/li-po-pil adresinden edinilmiştir.

Robot malzemeleri (2022, 8 Ocak). Radiolink pixhawk 32 bit uçuş kontrol kartı + güç modülü. 18.03.2023 tarihinde https://www.robotzade.com/ adresinden edinilmiştir.

Robotzade (2022, 14 Şubat). İHA Pervane Modelleri. 07.06.2023 tarihinde https://www.robotzade.com/ adresinden edinilmiştir.

Ultralytics (2022, 2 Ocak). Ultralytics YOLOv8 Docs. 18.03.2023 tarihinde https://docs.ultralytics.com/ adresinden edinilmiştir.

Ultralytics (2021, 5 Mart). Yolo 8 versiyonları. 13.05.2023 tarihinde https://docs.ultralytics.com/ adresinden edinilmiştir.

Viso.ai. (2022, 12 Mayıs). Yolo7 Nesne Algılama Algoritması Kılavuzu. 07.06.2023 tarihinde https://viso.ai/deep-learning/yolov7-guide/ adresinden edinilmiştir.

Yuret, D. (2016, 3 Kasım). Julia ve Knet ile Derin Öğrenmeye Giriş, 22.06.2023 tarihinde http://www. denizyuret. com/2016/09/julia-ve-knet-ile-derin-ogrenmeye- giris adresinden edinilmiştir.

EKLER

Ek-1: HSV Renk Kodları Kullanarak Askeri Araç Tespitinde Kullanılan Kodlar

```
import cv2  # cv2 isimli kütüphanemiz OpenCV kütüphanesidir.
import numpy as np # Numpy kütüphanesini belirli yerlerde rastgele fotoğraflar oluşturmak için ekliyorum.
from collections import deque
import time    # Time kütüphanesini zamanı gelince video işlemlerinde kullanacağımız için ekliyorum.
import matplotlib.pyplot as plt
# nesne merkezini depolayacak veri tipi
buffer_size = 16  # deque boyutu
pts = deque(maxlen=buffer_size)
# mavi renk aralığı - HSV
#blueLower = (84, 98, 0)
#blueUpper = (179, 255, 255)
# hamkonvoy renk aralığı - HSV
#blueLower = (110, 98, 0)
#blueUpper = (120,255, 255)
# tanktreni renk aralığı - HSV
#blueLower = (11, 98, 0)
#blueUpper = (14,255, 255)
#hamkonvoy2 renk aralığı - HSV
blueLower = (3, 80, 10)
blueUpper = (25,255, 255)
# "VideoCapture()" fonksiyonu içine yazılan adresteki videoyu içeri aktarır ve bir değişkene eşitler
cap = cv2.VideoCapture("hamkonvoy2.mp4")
while True:
    success, imgOriginal = cap.read()
    if success:
        # detayi azaltip noise azaltma
        blurred = cv2.GaussianBlur(imgOriginal, (11, 11), 0)
        # hsv
        hsv = cv2.cvtColor(blurred, cv2.COLOR_BGR2HSV)
```

```
        # "imshow()" fonksiyonu ile pencereyi görselleştiriyoruz
                cv2.imshow("HSV Image", hsv)
                # mavi icin maske
                mask = cv2.inRange(hsv, blueLower, blueUpper)
                cv2.imshow("mask Image", mask)
                # maskenin etrafinda kalan gurultuleri sil
                mask = cv2.erode(mask, None, iterations=1)
                mask = cv2.dilate(mask, None, iterations=1)
                cv2.imshow("mask+erode-dilate", mask)
                # contour
                (contours,    _)    =    cv2.findContours(mask.copy(),
cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
                center = None
                if len(contours) > 0:
                    # en buyuk konturu al
                    c = max(contours, key=cv2.contourArea)
                    # dikdortgene cevir
                    rect = cv2.minAreaRect(c)
                    ((x, y), (width, height), rotation) = rect
                    s  =  "x:  {},  y:  {},  width:  {},  height:  {},
rotation: {}".format(np.round(x),
                                            np.round(y),
                                            np.round(width),
        np.round(height),
                                             np.round(rotation))
                    print(s)
                    box = cv2.boxPoints(rect)
                    box = np.uint64(box)
                    # moment - goruntunun merkezini bulmamiza yarayan
yapi
                    M = cv2.moments(c)
                    center = (int(M["m10"] / M["m00"]),
                              int(M["m01"] / M["m00"]))
                    # contour'u cizdir
                    cv2.drawContours(imgOriginal,  [box],  0,  (0,  255,
255), 2)
                    # merkeze nokta ciz
                    cv2.circle(imgOriginal,  center,  5,  (255,  0,  255),
-1)
                    # bilgileri ekrana yazdir
```

```
                cv2.putText(imgOriginal,      s,      (20,      20),
cv2.FONT_HERSHEY_COMPLEX_SMALL, 0.4, (0, 0, 0), 1)
            # deque (nokta takip)  yeşil çizgiler çiziyor
          #pts.appendleft(center)
            #for i in range(1, len(pts)):
                #if pts[i - 1] is None or pts[i] is None: continue
                #cv2.line(imgOriginal,  pts[i  -  1],  pts[i],  (0,
255, 0), 3)
      # Time modülündeki "sleep()" fonksiyonunu kullanarak
pencerelerin
      # daha yavaş görselleştirilmesini sağlıyoruz.
      # Bunu yapmaz isek videomuz çok hızlanır.
            time.sleep(0.02)
            cv2.imshow("SERKANIN     ASKERI     ARAC     TESPITI",
imgOriginal)
            if cv2.waitKey(1) & 0xFF == ord("q"):
                cv2.destroyAllWindows()
                break
```

Ek-2: Webcam Okumada Kullanılan Kodlar

```
from ultralytics import YOLO
import cv2
import cvzone
import math

cap = cv2.VideoCapture(0)   # For Webcam
cap.set(3, 1280)  #ekranın büyüklüğünü ayarlıyor
cap.set(4, 720)   #ekranın büyüklüğünü ayarlıyor

model = YOLO('../Yolo-Weights/yolov8n.pt')

classNames = ["person", "bicycle", "car", "motorbike", "aeroplane",
          "bus", "train", "truck", "boat",
              "traffic light", "fire hydrant", "stop sign", "parking
          meter", "bench", "bird", "cat",
              "dog",  "horse",  "sheep",  "cow",  "elephant",  "bear",
          "zebra", "giraffe", "backpack", "umbrella",
              "handbag",   "tie",   "suitcase",   "frisbee",   "skis",
          "snowboard", "sports ball", "kite", "baseball bat",
              "baseball  glove",  "skateboard",  "surfboard",  "tennis
          racket", "bottle", "wine glass", "cup",
              "fork",  "knife",  "spoon",  "bowl",  "banana",  "apple",
          "sandwich", "orange", "broccoli",
              "carrot",   "hot   dog",   "pizza",   "donut",   "cake",
          "chair", "sofa", "pottedplant", "bed",
              "diningtable",    "toilet",    "tvmonitor",    "laptop",
          "mouse", "remote", "keyboard", "cell phone",
              "microwave",      "oven",      "toaster",      "sink",
          "refrigerator", "book", "clock", "vase", "scissors",
              "teddy bear", "hair drier", "toothbrush"
              ]

while True:
    success, img = cap.read()
    results = model(img, stream=True)
    for r in results:
        boxes = r.boxes
        for box in boxes:
```

```
            # Bounding Box
            x1, y1, x2, y2 = box.xyxy[0]
            x1, y1, x2, y2 = int(x1), int(y1), int(x2), int(y2)
            # cv2.rectangle(img,(x1,y1),(x2,y2),(255,0,255),3)
            w, h = x2 - x1, y2 - y1
            cvzone.cornerRect(img, (x1, y1, w, h))
            # Confidence
            conf = math.ceil((box.conf[0] * 100)) / 100
            # Class Name
            cls = int(box.cls[0])

            cvzone.putTextRect(img,   f'{classNames[cls]}   {conf}',
          (max(0, x1), max(35, y1)),scale=1, thickness=1)

    cv2.imshow("Image", img)
    if cv2.waitKey(1) & 0xFF == ord('q'):   # q ya basınca ekran
         kapanacak
        break

video.release()

# Destroy all the windows açık kalan birden çok pencereyi kapatır
cv2.destroyAllWindows()
```

Ek-3: Video Okumada Kullanılan Kodlar

```
#  YOLOv7_8   videodan görüntü tanıma
from ultralytics import YOLO
import cv2
import cvzone
import math
cap = cv2.VideoCapture("hamkonvoy.mp4")   # For Video   KENDİ VİDEO
          YOLUNA GÖRE DÜZENLE
model = YOLO('../Yolo-Weights/yolov8n.pt')
classNames = ["person", "bicycle", "car", "motorbike", "aeroplane",
          "bus", "train", "truck", "boat",
               "traffic light", "fire hydrant", "stop sign", "parking
          meter", "bench", "bird", "cat",
               "dog", "horse", "sheep", "cow", "elephant", "bear",
          "zebra", "giraffe", "backpack", "umbrella",
               "handbag", "tie", "suitcase", "frisbee", "skis",
          "snowboard", "sports ball", "kite", "baseball bat",
               "baseball glove", "skateboard", "surfboard", "tennis
          racket", "bottle", "wine glass", "cup",
               "fork", "knife", "spoon", "bowl", "banana", "apple",
          "sandwich", "orange", "broccoli",
               "carrot", "hot dog", "pizza", "donut", "cake",
          "chair", "sofa", "pottedplant", "bed",
               "diningtable", "toilet", "tvmonitor", "laptop",
          "mouse", "remote", "keyboard", "cell phone",
               "microwave", "oven", "toaster", "sink",
          "refrigerator", "book", "clock", "vase", "scissors",
               "teddy bear", "hair drier", "toothbrush"
               ]
while True:
    success, img = cap.read()
    results = model(img, stream=True)
    for r in results:
        boxes = r.boxes
        for box in boxes:
            # Bounding Box
            x1, y1, x2, y2 = box.xyxy[0]
            x1, y1, x2, y2 = int(x1), int(y1), int(x2), int(y2)
            # cv2.rectangle(img,(x1,y1),(x2,y2),(255,0,255),3)
            w, h = x2 - x1, y2 - y1
```

```
            cvzone.cornerRect(img, (x1, y1, w, h))
            # Confidence
            conf = math.ceil((box.conf[0] * 100)) / 100
            # Class Name
            cls = int(box.cls[0])
            cvzone.putTextRect(img, f'{classNames[cls]} {conf}',
        (max(0, x1), max(35, y1)),scale=1, thickness=1)
    cv2.imshow("Image", img)
    cv2.waitKey(1)
```

Ek-4: İHA Yakalamada Kullanılan Kodlar

```
from ultralytics import YOLO
import cv2
import cvzone
import math
import time
classNames = ["person", "bicycle", "car", "motorbike",
"aeroplane", "bus", "train", "truck", "boat",
              "traffic light", "fire hydrant", "stop sign",
"parking meter", "bench", "bird", "cat",
              "dog", "horse", "sheep", "cow", "elephant",
"bear", "zebra", "giraffe", "backpack", "umbrella",
              "handbag", "tie", "suitcase", "frisbee", "skis",
"snowboard", "sports ball", "kite", "baseball bat",
              "baseball glove", "skateboard", "surfboard",
"tennis racket", "bottle", "wine glass", "cup",
              "fork", "knife", "spoon", "bowl", "banana",
"apple", "sandwich", "orange", "broccoli",
              "carrot", "hot dog", "pizza", "donut", "cake",
"chair", "sofa", "pottedplant", "bed",
              "diningtable", "toilet", "tvmonitor", "laptop",
"mouse", "remote", "keyboard", "cell phone",
              "microwave", "oven", "toaster", "sink",
"refrigerator", "book", "clock", "vase", "scissors",
              "teddy bear", "hair drier", "toothbrush"
              ]
prev_frame_time = 0
new_frame_time = 0
# cap = cv2.VideoCapture(1)  # For Webcam
# cap.set(3, 640)
# cap.set(4, 480)
cap = cv2.VideoCapture("hareketli_ucak3.mp4")
# video kayıt için fourcc ve VideoWriter tanımlama
cv2_fourcc = cv2.VideoWriter_fourcc(*'mp4v')
success, img = cap.read()
print(img.shape)
yukseklik = img.shape[0]
genislik = img.shape[1]
cv2.imwrite("ornek_resim.jpg", img)
size = list(img.shape)
```

```
      del size[2]
      size.reverse()
      video  =  cv2.VideoWriter("kaydedilen_video.mp4",  cv2_fourcc,
24, size) #output video name, fourcc, fps, size
      model = YOLO("yolov8n.pt")
      while True:
          new_frame_time = time.time()
          success, img = cap.read()
          img         =         cv2.resize(img,         (1280,720),
interpolation=cv2.INTER_AREA)
          results = model(img, stream=True)
          for r in results:
              boxes = r.boxes
              for box in boxes:
                  # Bounding Box
                  x1, y1, x2, y2 = box.xyxy[0]
                  x1,  y1,  x2,  y2  =  int(x1),  int(y1),  int(x2),
int(y2)
                  # cv2.rectangle(img,(x1,y1),(x2,y2),(255,0,255),3)
                  w, h = x2 - x1, y2 - y1
                  cvzone.cornerRect(img, (x1, y1, w, h))
                  # Confidence
                  conf = math.ceil((box.conf[0] * 100)) / 100
                  # Class Name
                  cls = int(box.cls[0])
                  cvzone.putTextRect(img,         f'{classNames[cls]}
{conf}', (max(0, x1), max(35, y1)), scale=1, thickness=1)
                  cx, cy = x1 + w // 2, y1 + h // 2
                  cv2.circle(img,  (cx,  cy),  5,  (255,  0,  255),
cv2.FILLED)
                  cx2, cy2 = 1280 // 2, 720 // 2
                  cv2.circle(img,  (cx2,  cy2),  5,  (255,  0,  255),
cv2.FILLED)
                  cv2.rectangle(img, (120, 120), (1280-120,720-120),
(255, 0, 0), 2)
                  cv2.line(img, (cx2,cy2), (cx,cy), (255, 0, 0), 1)
          # video kayıt
          video.write(img)
          fps = 1 / (new_frame_time - prev_frame_time)
          prev_frame_time = new_frame_time
          print("fps: ", fps)
```

```
            cv2.imshow("Image", img)
            cv2.waitKey(1)
video.release()
```

Ek-5: Roboflow Veri Seti İle Foto Yakalamada kullanılan Kodlar

```
from ultralytics import YOLO
from PIL import Image
from roboflow import Roboflow
# BURADA HAZIR SET KULLANMIYORUZ
#ROBOFLOW SİTESİNDE KENDİ EĞİTTİĞİMİZ SETİN API SINI ALIYORUZ
#HAZIR SET KULLANSAYDIK AŞAĞIDAKİ 3 SATIR YERİNE
# model = YOLO('yolov8n.pt') KULLANIRDIK
rf = Roboflow(api_key="gzdnFwHGpAFr97C5N0Nq")
project = rf.workspace("militarywehiclev1-
yr0vv").project("tank-lepry")
model = project.version(1).model
# infer on a local image ÖLÇÜLERİ NELER YAKALADIĞINI VERİYOR
görüntü vermez
print(model.predict("tankfoto.jpg", confidence=40,
overlap=30).json())
# visualize your prediction GÖRÜNTÜYÜ YAKALAR prediction.jpg
diye bir dosyaya atar
model.predict("tankfoto.jpg", confidence=40,
overlap=30).save("prediction.jpg")
#model.predict("ZMA.mp4", confidence=40,
overlap=30).save("prediction.mp4")
#sonuc = model.predict(source="hamkonvoy.mp4", show=True)
# infer on an image hosted elsewhere
# print(model.predict("URL_OF_YOUR_IMAGE", hosted=True,
confidence=40, overlap=30).json())
```

Ek-6: Videodan Görüntü Yakalamada Kullanılan Kodlar

```
from ultralytics import YOLO
from PIL import Image
model = YOLO('yolov8n.pt')
# infer on a local image
#print(model.predict("pazar.jpeg", confidence=40,
overlap=30).json())
# visualize your prediction
#model.predict("tankfoto.jpg", confidence=40,
overlap=30).save("prediction.jpg")
#model.predict("ZMA.mp4", confidence=40,
overlap=30).save("prediction.mp4")
sonuc = model.predict(source="hamkonvoy.mp4", show=True)
# infer on an image hosted elsewhere
# print(model.predict("URL_OF_YOUR_IMAGE", hosted=True,
confidence=40, overlap=30).json())
```

Ek-7: Fotoğraf Okumada Kullanılan Kodlar

```
from ultralytics import YOLO
import cv2
model = YOLO('yolov8n.pt')
result = model("pazar.jpeg", show=True)
cv2.waitKey(0)
```

ÖZGEÇMİŞ

Serkan GÜNDÜZ

EĞİTİM DURUMU:

- **Lisans:** Akdeniz Üniversitesi, Matematik Bölümü
- **Yüksek Lisans:** (Devam) İstanbul Gedik Ünivesitesi, Yapay Zekâ Mühendisliği Tezli Yüksek Lisans Programı

MESLEKİ DENEYİM:

- Halen Kocaeli Gebze ilçesinde yer alan TÜBİTAK Yerleşkesi içerisinde faaliyet gösteren TÜBİTAK Fen Lisesinde Matematik Öğretmeni ve Müdür Yardımcısı olarak görev yapmaktadır.