ARAÇ KAMERASI GÖRÜNTÜLERİNDE NESNE VE NESNE HAREKETİ TESPİTİ İÇİN BÜTÜNLEŞİK BİR DERİN ÖĞRENME MİMARİSİ

Yüksek Lisans Tezi

Özlem OKUR

Eskişehir 2024

ARAÇ KAMERASI GÖRÜNTÜLERİNDE NESNE VE NESNE HAREKETİ TESPİTİ İÇİN BÜTÜNLEŞİK BİR DERİN ÖĞRENME MİMARİSİ

Özlem OKUR

Yüksek Lisans Tezi

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Donanımı Bilim Dalı

Danışman: Dr. Öğr. Üyesi Mehmet KILIÇARSLAN

Eskişehir

Eskişehir Teknik Üniversitesi

Lisansüstü Eğitim Enstitüsü

Temmuz 2024

JÜRİ VE ENSTİTÜ ONAYI

Özlem OKUR'un ARAÇ KAMERASI GÖRÜNTÜLERİNDE NESNE VE NESNE HAREKETİ TESPİTİ İÇİN BÜTÜNLEŞİK BİR DERİN ÖĞRENME MİMARİSİ başlıklı çalışması 12/07/2024 tarihinde aşağıdaki jüri tarafından değerlendirilerek "Eskişehir Teknik Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliği"nin ilgili maddeleri uyarınca, Bilgisayar Mühendisliği Anabilim dalında Yüksek Lisans Tezi olarak kabul edilmiştir.

Jüri Üyeleri	**Unvan Adı Soyadı**	**İmza**
Üye	: Dr. Öğr. Üyesi Mehmet KILIÇARSLAN	
Üye	: Prof. Dr. Serkan GÜNAL	
Üye	: Doç. Dr. Alper BİLGE	

Prof. Dr. Semra KURAMA

Lisansüstü Eğitim Enstitüsü Müdürü

12/07/2024

DANIŞMAN ONAYI

Danışmanlığını yürüttüğüm Yüksek Lisans öğrencisi Özlem OKUR, ARAÇ KAMERASI GÖRÜNTÜLERİNDE NESNE VE NESNE HAREKETİ TESPİTİ İÇİN BÜTÜNLEŞİK BİR DERİN ÖĞRENME MİMARİSİ başlıklı tez çalışmasını tamamlamıştır. Hazırlamış olduğu tez tarafımca incelenmiş ve öğrencinin tez savunma sınavına alınması bilimsel ve etik açıdan uygun görülmüştür.

Tez Danışmanı

Dr. Öğr. Üyesi Mehmet KILIÇARSLAN

ÖZET

ARAÇ KAMERASI GÖRÜNTÜLERİNDE NESNE VE NESNE HAREKETİ TESPİTİ İÇİN BÜTÜNLEŞİK BİR DERİN ÖĞRENME MİMARİSİ

Özlem OKUR

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Donanımı Bilim Dalı

Eskişehir Teknik Üniversitesi, Lisansüstü Eğitim Enstitüsü, Temmuz 2024

Danışman: Dr. Öğr. Üyesi Mehmet KILIÇARSLAN

Akıllı araçlar ve otonom sürüş teknolojilerindeki gelişmeler güvenli ve konforlu bir sürüş için kritik öneme sahiptir. Kısmi ya da tamamen sürücüden bağımsız gerçekleşen sürüş faaliyetleri olarak tanımlanan otonom sürüş ve akıllı araçlarda bir nesnenin tespitinin ve hareketinin hızlı bir şekilde belirlenmesi için çok sayıda uygulama bulunmaktadır. Bu uygulamalarda kullanılan geleneksel algoritmalar genellikle önce nesneleri video karelerinde tespit eder, ardından bu nesnelerin hareketlerini izleyip analiz eder. Bu çalışma, uzamsal (görüntüdeki konum) ve zamansal (zaman içindeki değişim) bilgileri birleştiren yeni bir derin öğrenme tabanlı nesne ve hareket tespit yöntemi sunmaktadır. Model, nesnelerin konumlarını ve hareketlerini aynı anda analiz ederek farklı nesne sınıflarının benzer hareket desenlerini tespit eder. Nesnelerin bu hareket desenleri, uzamsal ve zamansal alanda izler bırakır. Hareket Profili Görüntüleri olarak adlandırılan bu izler, nesnelerin zaman içindeki hareketlerini görselleştirir ve karmaşık sürüş ortamlarında hareket kalıplarını anlamayı sağlar. Bu tez çalışması, geniş bir sürüş videosu veri seti kullanarak nesneleri ve hareketlerini doğrudan tespit eden yenilikçi bir yaklaşım sunmaktadır. Dinamik sürüş senaryolarının zorluklarıyla başa çıkmak için özel olarak tasarlanan bu yöntem, gerçek zamanlı performans sunar. Kamuya açık veri kümelerinde %78 hassasiyet ve 3,09˚ ortalama hata ile yüksek doğruluk sağlar ve %22'lik bir performans iyileştirmesi göstermektedir.

Anahtar Sözcükler: Nesne algılama, Hareket analizi, Uzamsal zamansal görüntüler, Derin öğrenme.

ABSTRACT

A UNIFIED DEEP LEARNING ARCHITECTURE FOR OBJECT AND MOTION DETECTION IN VEHICLE CAMERAS

Özlem OKUR

Department of Computer Engineering

Programme in Program in Computer Hardware

Eskişehir Technical University, Institute of Graduate Programs, July 2024

Supervisor: Lütfen seçiniz Mehmet KILIÇARSLAN

Developments in intelligent vehicles and autonomous driving technologies are crucial for ensuring safe and comfortable driving. In autonomous driving and intelligent vehicles, where driving activities are defined as partially or fully independent of the driver, various applications exist for the rapid detection and determination of object movement. Traditional algorithms used in these applications typically involve a two-step process: first detecting objects in video frames, and then tracking and analyzing their movements. This study presents a novel deep learning-based method that integrates spatial (position in the image) and temporal (change over time) information into a single architecture. The model simultaneously analyzes the objects' positions and movements to detect similar motion patterns across different object classes. These motion patterns leave traces in both spatial and temporal domains. These traces, known as Motion Profile Images, visualize the objects' movement over time and help in understanding motion patterns in complex driving environments. The thesis introduces an innovative approach that directly detects objects and their movements from a large dataset of driving videos. Specifically designed to address the challenges of dynamic driving scenarios, this method provides real-time performance. It demonstrates its effectiveness in practical applications by delivering interpretable motion detection in both spatial and temporal domains. Additionally, it confirms its reliability with high accuracy, achieving 78% average sensitivity and 3.09˚ average error on public datasets, reflecting a 22% improvement in performance.

Keywords: Object detection, Motion analysis, Spatial temporal images, Deep learning.

TEŞEKKÜR

Tez çalışmam boyunca emeğini ve desteğini esirgemeyen, bilgi ve deneyimlerini aktararak yol gösterici olan danışmanım Sayın Dr. Öğr. Üyesi Mehmet KILIÇARSLAN'a çok teşekkür ederim.

Özlem OKUR

12/07/2024

ETİK İLKE VE KURALLARA UYGUNLUK BEYANNAMESİ

Bu tezin bana ait, özgün bir çalışma olduğunu; çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalarında bilimsel etik ve kurallara uygun davrandığımı; bu çalışma kapsamında elde edilen tüm veri ve bilgiler için kaynak gösterdiğimi ve bu kaynaklara kaynakçada yer verdiğimi; bu çalışmanın Eskişehir Teknik Üniversitesi tarafından kullanılan "bilimsel intihal tespit programı"yla tarandığını ve hiçbir şekilde "intihal içermediğini" beyan ederim. Herhangi bir zamanda, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun saptanması durumunda, ortaya çıkacak tüm ahlaki ve hukuki sonuçları kabul ettiğimi bildiririm.

Özlem OKUR

İÇİNDEKİLER

Sayfa

TABLOLAR DİZİNİ

Sayfa

ŞEKİLLER DİZİNİ

Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

α	: Alpha
β	: Beta
ADAS	: Gelişmiş Sürücü Destek Sistemleri
CNN	: Evrişimli Sinir Ağları
GNSS	: Küresel Navigasyon Uydu Sistemi
IoU	: Kesişim Üzerinde Birlik
Lidar	: Işık Algılama ve Aralık Belirleme
LSTM	: Uzun-Kısa Süreli Hafıza
MAE	: Ortalama Mutlak Hata
mAP	: Ortalama Hassasiyet Puanı
MPI	: Hareket Profili Görüntüsü
R-CNN	: Bölge Tabanlı Evrişimli Ağlar
SAE	: Uluslararası Otomobil Mühendisleri Topluluğu
SSD	: Tek Atışlı Çok Kutu Dedektörü
TTC	: Çarpışma Zamanı
TÜİK	: Türkiye İstatistik Kurumu
YOLO	: You Only Look Once

1. GİRİŞ

Son yıllarda teknolojinin hızla ilerlemesiyle otomotiv sektöründe önemli gelişmeler kaydedilmiştir. Gelişmiş sürücü destek sistemleri (Advanced Driver Assistance Systems - ADAS) ve otonom sürüş gibi inovatif çözümler, sürüş güvenliğini artırmak, trafik akışını optimize etmek ve sürücülere ek konfor sağlamak amacıyla geliştirilmiştir. Ancak, artan araç kullanımıyla birlikte trafik kazaları önemli bir sorun olmaya devam etmektedir. Tablo 1.1'de yer aldığı üzere Türkiye İstatistik Kurumu (TÜİK) verilerine göre, yolcu, yaya, yol ve araç kusurları gibi çeşitli faktörler etkili olsa da trafik kazalarının çoğunun sürücü hatalarından kaynaklandığı tespit edilmiştir (TÜİK, 2023). Otonom araçların yaygınlaşması ve otonom araç teknolojisinin ilerlemesiyle birlikte trafik kazalarının önemli ölçüde azalabileceği öngörülmektedir.

Tablo 1.1. *Trafik kazalarına neden olan kusurlar (TÜİK, 2023)*

Yıl	**Sürücü Kusuru** (%)	**Yaya Kusuru** (%)	**Araç Kusuru** (%)	**Yolcu Kusuru** (%)	**Yol Kusuru** (%)
2015	89,3	8,8	0,55	0,43	0,91
2016	89,59	8,73	0,47	0,41	0,81
2017	89,87	8,48	0,52	0,37	0,7
2018	89,46	8,44	0,62	0,88	0,6
2019	88,00	8,20	0,20	1,30	0,5
2020	88,30	7,00	2,70	1,40	0,5
2021	87,1	8,2	2,6	1,8	0,4
2022	86,8	9,5	2,1	1,2	0,4
2023	88,9	9,00	1,1	0,6	0,3

Otomotiv sektöründe otonom; yapay zekâ ve derin öğrenme algoritmalarıyla desteklenen yazılım ve gelişmiş donanımlarla toplanan anlık verilerin işlenmesi yoluyla çalışan kısmi ya da tamamen sürücüsüz hareket edebilen araçları tanımlamak için kullanılmaktadır. Otonom sürüş seviyeleri, araçların fonksiyonel özelliklerini ve sürücü müdahalesinin gerekip gerekmediğini belirten bir kategorizasyon sistemidir. Uluslararası Otomobil Mühendisleri Topluluğu (SAE) tarafından belirlenen otonom sürüş seviyeleri

otomasyon teknolojisi bulunmayan Seviye 0 ile tamamen sürücüden bağımsız Seviye 5 arasında sınıflandırılmıştır (Abraham ve Rabin, 2019).

- Seviye 0 – Sürücü Destekli: Bütün sürüş fonksiyonları sürücü tarafından yapılır.
- Seviye 1 – Sürücü Destekli: Araçta hız sabitleyici, şerit takip sistemi ve fren asistanı gibi sistemler bulunur, ancak sürücü sürekli aracı kontrol eder ve sisteme müdahale edebilir.
- Seviye 2 – Kısmi Otomasyon: Araç, belirli durumlar için kendi kendine yönetim yapabilir (örneğin, hızlanma, frenleme ve yönlendirme), ancak sürücü sürekli olarak dikkatini korumalı ve gerektiğinde müdahale etmelidir.
- Seviye 3 – Koşullu Otomasyon: Araç, belirli koşullar altında tamamen otonom olarak hareket edebilir, ancak sürücü, sistem uyarısına göre aracı devralabilir.
- Seviye 4 – Koşullu Otonom: Araç, belirli sınırlı alanlarda veya koşullarda tamamen otonom olarak hareket edebilir. Ancak, bazı durumlarda sürücü müdahalesi gerekir. Günümüzde otonom araç endüstrisinin ulaştığı en yüksek seviyeyi temsil etmektedir.
- Seviye 5 – Tam Otonom: Araç her türlü yol ve hava koşullarında tamamen otonom olarak hareket edebilir ve sürücü müdahalesi gerektirmez.

Otonom sürüş teknolojileri, sürücü hatalarını en aza indirerek güvenliği artırmayı ve sürüş deneyimini optimize etmeyi hedefleyen, sürekli gelişen ve iyileşen bir alandır. Bu teknolojiler, araçların yol ve çevre koşullarına dair detaylı bilgileri toplamak için çeşitli donanımlar kullanmaktadır. Bu donanımlar genellikle kameralar, radarlar, lidarlar ve ultrasonik sensörler gibi çeşitli algılayıcılardan oluşur (Yeong vd., 2021). Bu sensörler, aracın etrafındaki nesneleri algılar ve çevresel koşulları değerlendirir. Toplanan bu veriler, aracın konumunu, trafiğin durumunu ve yol ile hava koşullarını anlamasına yardımcı olur.

- Radar: Elektromanyetik dalgalar ile aracın çevresindeki nesnelerin yerini ve hızını belirlemede kullanılır. Uzak mesafedeki araçları algılamada lidardan daha başarılı olsa da uzaklık arttıkça hata oranı yükselmektedir. Hareketli nesnelerin tespiti ve hızının algılanmasında başarılı iken hareketsiz nesnelerin tespitinde zayıftır.
- Lidar (Light Detection and Ranging): lazer ışınları kullanarak çevredeki nesnelerin 3 boyutlu haritasını oluşturur. Yüksek çözünürlüklü olması sayesinde

aracın etrafındaki objelerin mesafesini ve şeklini hassas bir şekilde tespit edebilmesi güçlü özelliklerindendir. Ancak, lidarların zamansal çözünürlüğü kameralarla karşılaştırıldığında daha zayıftır. Bu, hızlı hareket eden nesnelerin tespitinde lidarların kameralar kadar etkili olmamasına yol açabilir. Bu nedenle, otonom araçlarda lidarlar genellikle kameralar ve diğer sensörlerle birlikte kullanılarak eksiksiz ve güvenilir bir algılama sistemi oluşturulur.

- Ultrasonik Sensörler: Kısa mesafede engelleri algılar, çoğunlukla park gibi düşük hızlı manevralarda kullanılır.
- GNSS (Global Navigation Satellite System): GPS gibi sistemler, aracın konumunu belirlemede ve harita oluşturmada kullanılır.
- Kameralar: Aracın etrafındaki nesneleri ve trafik işaretlerini algılamak için kullanılır. Yüksek çözünürlüklü kameralar, aracın ön, yan ve arka bölgelerini kapsayacak şekilde yerleştirilir. Görüntü işleme algoritmaları, kamera verilerini analiz ederek aracın çevresini anlamasına yardımcı olur. Kameralar diğer sensörlere göre daha düşük maliyetlidir ayrıca lidar ve radarlara göre daha geniş bir görüş alanında, yüksek çözünürlükte nesne tespiti ve takibi için kullanılırlar (Gürtaş, 2021).

Kameralar işlevselliklerine göre farklı kategorilere ayrılmıştır. İç kameralar sürücü ve yolcu güvenliği için araç içinde bulunmaktadır. Bu kameralar, sürücünün dikkatini izlemek ve iç mekandaki yolcuların güvenliğini sağlamak amacıyla kullanılmaktadır. Ön görüş kameraları ise genellikle aracın ön camına monte edilip, yolu ve çevreyi görüntülemektedir. Bu kameralar, adaptif hız kontrolü, otomatik frenleme ve şerit takip sistemleri gibi güvenlik özelliklerinin etkin bir şekilde çalışmasını sağlamaktadır. Dış kameralar, aracın ön, arka ve yanlarına yerleştirilerek çevredeki nesneleri algılamak ve diğer araçları izlemek için kullanılır. Bu kameralar, park yardımı, kör nokta tespiti ve çevresel farkındalık gibi özellikleri destekler. Gece görüş kameraları, özellikle düşük ışık koşullarında ve gece sürüşlerinde performans göstermek amacıyla infrared (kızılötesi) teknoloji ile donatılmış olup sürücüye daha iyi bir görüş sağlar (Liu vd., 2021). Her bir kamera türü, aracın farklı yönlerini ve sürüş güvenliğini artırmak için belirli bir görevde optimize edilmiştir. İç kameralar, sürücünün dikkatini izlerken, ön görüş ve dış kameralar yol ve çevre hakkında bilgi toplar. Böylece, bu kameralar bir arada çalışarak araç içi ve dışı güvenlik ve sürüş konforunu artırır.

Bu tez çalışmasının amacı, otonom ve akıllı araç teknolojilerinde nesneleri ve nesnelerin hareketlerini tespit ederek, araçların çevresel verileri daha etkin bir şekilde analiz etmesini sağlamaktır. Bu doğrultuda, nesnelerin hareketlerini, yönlerini ve niyetlerini öznitelik olarak anlamak, akıllı araçların karar alma mekanizmaları için kritik bir görevdir. Bu amaçla, çalışma kapsamında kamera verileri kullanılmıştır. Kameralar, aracın çevresindeki nesnelerin tespit edilmesi ve bu nesnelerin davranışlarının detaylı bir şekilde analiz edilmesi için önemli bir rol oynamaktadır. Böylece, kameralar aracılığıyla elde edilen veriler, aracın doğru ve zamanında kararlar alabilmesi için hayati öneme sahip bilgiler sunmaktadır.

Mevcut yöntemler, genellikle ayrı algılama ve izleme aşamalarına dayanmakta veya kareler arası optik akış kullanarak video karelerinden hareket bilgileri çıkarmaktadır. Ancak, dinamik olarak değişen arka planlar ve gerçek zamanlı sürüş videolarındaki işleme gereksinimleri, statik kameraların kullanımını daha zor hale getirmektedir. Bu yöntemler uçtan uca bir model sunmamakta, bunun yerine ardışık süreçleri bağlayan yapılar olarak tasarlanmaktadır. Çalışmada önerilen İki Akışlı Hareket (Two Strea[M]) YOLOv8 yöntemi ise tek bir kareyi ve uzamsal-zamansal bir görüntüyü birleştirerek nesneleri ve anlık hareketleri doğrudan tespit etmektedir. Bu yaklaşım, akıllı araçlar alanında umut verici bir alternatif sunmaktadır.

Akıllı aracın çevresindeki nesnelerin hareketi karmaşık ve dinamiktir. Bu hareketler, bir video karesinde gerçekte olduğu gibi 3 boyutlu (yatay x, dikey y ekseni ve z eksenindeki değişimler) olarak değil, genelde 2 boyutlu (görüntü uzayında yatay x ve dikey y yönde hareket) göreceli hareket olarak algılanmaktadır. Bu durum, nesnelerin görüntüde yatay hareket olarak algılanan yanal hareketlerin, gerçekte farklı derinlik seviyelerindeki değişimleri temsil ettiği anlamına gelmektedir. Örneğin, bir nesnenin kameraya yaklaşması ya da uzaklaşması, görüntüde yalnızca ölçeğinin değişmesi olarak gözükür. Bir çarpışma durumunda ise, nesne ile aracın yaklaşma hızı hem yatay hem de dikey yönde simetriktir. Ayrıca, akıllı aracın genellikle düz bir yol üzerinde seyrettiği ve sınırlı manevra kabiliyetine sahip olduğu varsayılarak, dikey hareketler göz ardı edilebilir. Bu bağlamda, nesnelerin hareketlerini anlamak ve akıllı aracın kontrolüyle ilgili kararlar almak için yatay hareketler önemlidir ve genellikle önceliklidir.

Tez çalışmasının ilerleyen bölümlerinde, “hareket” terimi, nesnenin farklı zaman dilimlerinde nasıl hareket ettiğini anlamak için kullanılan bir kavram olarak

tanımlanmaktadır. Bu bağlamda, hareket analizleri çeşitli zaman perspektiflerine göre ayrılmaktadır. “Orta vadeli” zaman dilimlerinde hareket, nesnenin belirli bir süre içinde yatay (x eksenindeki) hareketinin nasıl algılandığını belirten 2 boyutlu yatay göreceli hareket olarak tanımlanır. Bu terim, nesnenin belirli bir süre boyunca nasıl hareket ettiğini daha geniş bir perspektifte anlamaya yönelik bir yaklaşımı ifade etmektedir. “Kısa vadeli” olarak tanımlanan iki karelik optik akış terimi ise, nesnenin anlık hareketini gözlemlemek için kullanılır ve sadece iki ardışık karede nesnenin nasıl hareket ettiğini izlemeye olanak tanır. Bu yaklaşım, nesnenin anlık hareketlerini yakalamak için daha dar bir zaman dilimi kullanır. “Orta vadeli” terim, kısa vadeli gözlemlerden biraz daha uzun bir süreyi (genellikle 1 saniyeye kadar) kapsar ve bu süre zarfında nesnenin hareketi hakkında daha ayrıntılı bilgi sağlar. Bu, nesnenin hareket yönünü ve hızını daha kesin bir şekilde belirlemeye yardımcı olur. “Uzun vadeli” olarak adlandırılan süreç ise, nesnenin etkileşimlerini ve davranışlarını analiz etmek için 1 saniyeden daha uzun bir süreyi ifade eder. Bu süre boyunca, nesnenin çevresiyle olan etkileşimleri ve hareketleri daha geniş bir zaman aralığında incelenir, böylece daha kapsamlı bir hareket analizi yapılabilir.

Geleneksel yaklaşımlarda, nesnelerin algılanması ve takibi için genellikle iki aşamalı bir süreç izlenmektedir. İlk olarak, nesneler video görüntülerinde algılanır ve ardından bu nesnelerin görüntüler arasında takibi yapılır. Bu takip sürecinde, nesnenin konumu ve hareketi belirlenir. Alternatif olarak, optik akış adı verilen yöntemler kullanılarak da nesnenin bir kareden diğerine olan hareketi analiz edilir. Bu süreçler genellikle nesne izleme sistemlerinin temelini oluşturmaktadır. Optik akış; bir görüntüdeki nesnelerin piksellerinin zaman içindeki hareketini (hız ve yön) ifade eden bir konsepttir ve bir karede görünen nesnenin bir sonraki karedeki konumunu belirleme sürecini tanımlamaktadır. Yatay ve dikey olarak iki yönde harekete dayanan bir vektördür.

Eski yaklaşımların aksine, son zamanlarda önerilen yöntemlerden biri Hareket Profili Görüntüsü (Motion Profile Image – MPI) kullanımıdır (Kilicarslan vd, 2014). MPI, bir hareket profili boyunca bir nesnenin belirli bir zamana göre konumunu, hızını ve ivmesini görsel olarak temsil eden bir görüntüdür. MPI, sürüş videolarında nesnelerin hareketlerini anlamak için kullanılan bir yöntem olup, ufuk çevresindeki piksellerin dikey birikimini içerir ve yaklaşık olarak 1 saniyelik bir zaman diliminde bu birikimi anlık olarak saklar. Bu birikim, nesnelerin belirli zaman aralıklarında nasıl hareket ettiğini

özetler ve videonun tamamını izlemeye gerek kalmadan tek bir görüntüdeki nesne izlerine dönüştürebilir. Şekil 1.1'de ufuk çizgisine yakın yeşil dikdörtgen bant kullanılarak MPI oluşturulmuştur. Görselde yer alan üstteki görüntü mevcut kareyi, alttaki ise yaklaşık 1 saniye öncesinden alınmış MPI'yi göstermektedir. Bu yöntemle, orta vadeli hareketler uzaysal-zamansal görüntüde gözlemlenebilir ve nesneler tek bir video görüntüsüyle hızlıca tespit edilebilir.

Özellikle araçlar ve yayalar gibi hareketli nesneler ile sabit arka plan gibi duran unsurlar, hareket profili görüntüsü içinde belirli bir iz şeklinde hareket ederler. Nesne görüntüde açıkça görünüyorsa, bu hareket düzgün ve süreklidir. MPI'nin en büyük avantajı, nesne izleme ve hareket tespitini gerçekleştirebilmesidir, bu da optik akış veya diğer önceden hesaplama gerektiren yöntemlere ihtiyaç duymadan yapılabilir. Bu sayede, hareketin hızlı ve etkili bir şekilde tespit edilip analiz edilmesi sağlanabilir.

Şekil 1.1. *Yeşil bantla oluşturulan MPI, üstteki mevcut kareyi ve alttaki 1 saniye öncesini gösterir.*

Bu çalışmada, YOLOv8 tabanlı nesne algılama modelinin bir varyantı olan Two-Strea[M] YOLOv8 geliştirilmiş ve kullanılmıştır. Bu model, geleneksel YOLOv8'den farklı olarak, nesne tespiti ve hareket analizini bir araya getiren bir yaklaşım sunmaktadır. Two-Strea[M] YOLOv8, nesneleri ve hareketlerini tespit etmek için doğrudan video karelerini ve MPI'i girdi olarak almaktadır. Model, nesne algılama işlemi için tek bir karedeki görsel özellikleri kullanarak nesneleri sınıflandırır ve konumlarını belirlerken,

aynı zamanda MPI'deki izlerden nesnelerin zamansal hareketlerini analiz eder. Bu sayede hem nesnenin anlık konumu hem de zaman içindeki hareket izleri bir arada değerlendirilebilmektedir.

Two-Strea[M] YOLOv8'in bu yaklaşımı, nesne tespiti ve hareket analizini entegre bir şekilde gerçekleştirerek, sürüş videolarındaki nesne tespitini daha doğru ve kapsamlı hale getirir. Bu yöntem, özellikle dinamik sürüş senaryolarında ve karmaşık trafik ortamlarında nesnelerin hareketlerinin daha iyi anlaşılmasına ve akıllı araç sistemlerinin güvenliğini ve verimliliğini arttırmaya olanak tanımaktadır.

Akıllı araçlar, çevrelerindeki potansiyel riskleri hızlı bir şekilde değerlendirmelidir. Bu riskler, görüntü uzayında ~0 akışı ve merkezi akış olarak tanımlanabilir (Kilicarslan vd, 2019). Sürüş videolarında, ~0 akışı, bir çarpışma olayının meydana gelmesi için gerekli bir durumdur. ~0 akışı, bir nesnenin hareket etmediği veya çok yavaş hareket ettiği anlamına gelir. Bu tür bir akış, bir çarpışma olasılığına işaret eder çünkü hareketin durması veya çok yavaşlaması, iki nesnenin birbirine çok yakın olduğu anlamına gelir.

Örneğin, merkeze doğru yanlardan yaklaşan araçlar akıllı aracının yoluna girdiklerinde, hızlarını azaltırlar veya dururlar. Bu durum, sıfır akışı yaratır. Akıllı aracının bakış açısından, bu tür hareketler, yaklaşan aracın yoluna girme ve çarpışma riskini artırır. Dolayısıyla, sıfır akış bölgelerinin tespiti, bu tür potansiyel çarpışma senaryolarının erken tanımlanması için kritiktir. Park halindeki bir araç genellikle sabit olduğundan, aniden hareket etmediği sürece doğrudan bir tehdit oluşturmaz. Aynı şekilde, kaldırımda yürüyen veya karşıdan karşıya geçen yayalar farklı düzeylerde tehlike oluşturabilirler.

Karşıdan karşıya geçen araçlar veya yayalar gibi dinamik nesnelerin hareketi, MPI'deki sabit veya hareketli arka plan nesnelerinin hareketinden farklıdır. Dolayısıyla, MPI içindeki arka plan izlerinden sadece aktif dinamik nesne hareketlerini net bir şekilde ayırt etmek mümkündür. Kaldırımda yürüyen bir yaya, arka plan yörüngesine yakın olduğu için MPI'de doğru bir şekilde tespit edilemeyebilir. Benzer şekilde, dönüş için kenarda bekleyen bir araç, dönüş anında tespit edilmesi zor olabilir. Ayrıca, akıllı araç hareket halinde olduğunda ve bu yörüngeler arka planla birleştiğinde, bir nesnenin yörüngesinden sınıfını belirlemek zorlaşabilir. MPI içinde, yürüyen yayalar genellikle

zincir benzeri desenler oluşturur ve yol yüzeyine göre farklı özelliklere sahip oldukları için yakın ve orta derinlik aralıklarında ayırt edilebilirler (Kilicarslan vd, 2023).

Bu tez çalışması, Temel vd. (2024) tarafından sunulan güncel yaklaşımlar da dahil olmak üzere diğer çalışmalardan farklı olarak, nesnelerin ve hareketlerin uzamsal ve zamansal boyutta tespit edildiği bir çalışmayı içermektedir:

(1) İki akışlı bir model kullanarak nesnelerin konumunu, sınıfını ve hareketini tanımlama yeteneği geliştirilmiştir. Bu model, bir video karesi ile bir MPI kullanarak hareket analizini gerçekleştirmektedir ve sadece MPI kullanarak yapılan mevcut çalışmaların ötesine geçmektedir.

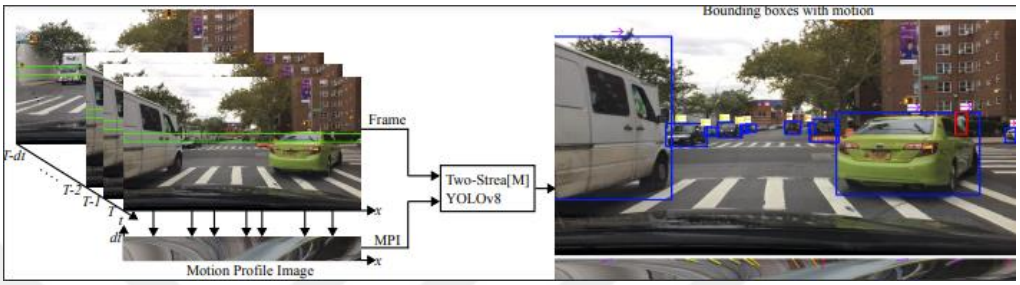
(2) Two-Strea[M] YOLOv8 adı verilen YOLOv8 tabanlı bir mimari, uçtan uca geliştirilmiştir. Bu model, akıllı araçlar için daha fazla bilgi üretme ve karar verme yeteneğini arttırmakta ve özellikle, gerçek zamanlı uygulamalarda kullanılabilen pratik bir çözüm sunmaktadır.

(3) MPI ile birleştirildiğinde, model son derece yorumlanabilir sonuçlar üretmektedir. Bu, video dizilerini izleme ihtiyacını ortadan kaldırmakta ve sadece iki görüntüyü analiz ederek detaylı bilgi sağlamaktadır.

(4) Model, kamuya açık büyük ve çeşitli bir veri kümesi olan MOT-BDD100K üzerinde eğitilmiş ve çeşitli sürüş ortamlarında başarılı sonuçlar göstermiştir.

Yöntem, sınırlı veri kullanarak sürüş videolarındaki nesneleri ve hareketi etkili bir şekilde tespit etmektedir. Bu başarı, görünüm özellikleri ile MPI içindeki zamansal özelliklerin başarılı bir şekilde birleştirilmesinden kaynaklanmaktadır. Ayrıca, model, nesne algılama modelleriyle karşılaştırılabilir Ortalama Ortalama Hassasiyet (Mean Average Precision - mAP) puanları ve hareket tahmini için çok düşük bir Ortalama Mutlak Hata (Mean Absolute Error - MAE) sağlamaktadır. Model, nesne tespitine hızlı bir şekilde yanıt vererek akıllı araçların çevrelerini hızla anlamalarına yardımcı olmaktadır. Bu parametreler, modelin nesne algılama ve hareket tahmini alanlarında yüksek performans sergilediğini ve diğer benzer modellerle rekabet edebilir düzeyde olduğunu göstermektedir. Şekil 1.2’de, video görüntülerinin işlenme sürecinden MPI oluşturmaya ve nesne-hareket algılamaya kadar olan süreci gösteren bir diyagram bulunmaktadır. Buna göre; ufuk çizgisine yakın yeşil dikdörtgen bant MPI oluşturmak için kullanılır. Siyah oklar MPI'deki nesnelerin konumlarını gösterir. Model, nesneleri ve

hareketleri aynı anda tespit etmek için kareyi ve MPI'yı birleşik bir mimaride kullanır. Sağdaki görüntü, nesne ve hareket tespit yöntemini net ve kapsamlı bir şekilde görselleştirir. Araçlar için mavi, yayalar için kırmızı renklendirilmiş sınırlayıcı kutular, sınıfları belirtir. Sınırlayıcı kutuların içindeki oklar hareket yönünü gösterir; okların rengi, hareket değerlerini derece cinsinden temsil eder.



Şekil 1.2. *MPI oluşturma süreci*

Tez çalışmasının devamı olarak sırasıyla ikinci bölümde önceki araştırmalar ve literatürdeki benzer çalışmalar incelenmiştir. Üçüncü bölümde sürüş videolarından elde edilen veriler üzerinde yapılan hareket analizleri ve sınırlayıcı kutulardan (bounding box) hareket hesaplamaları sunulmaktadır. Ayrıca, MPI görüntüsü içinde bulunan çevrenin karmaşıklığı tartışılmaktadır. Dördüncü bölümde uzamsal-zamansal alanda nesne ve hareket tespiti için önerilen derin öğrenme modeli olan İki Akışlı Two Strea[M] YOLOv8 modeli detaylı bir şekilde açıklanmaktadır. Beşinci bölümde önerilen yöntemin deneysel sonuçları özetlenmekte ve analiz edilmektedir. Altıncı bölümde elde edilen sonuçlar ve bulgular çerçevesinde geniş bir tartışma sunulmaktadır. Yedinci bölümde tez çalışmasının özeti verilerek elde edilen sonuçlar ve çalışmanın katkıları vurgulanmaktadır.

2. İLGİLİ ÇALIŞMALAR

Nesne tespiti, günümüz görüntü işleme ve bilgisayarla görme alanındaki en dinamik araştırma konularından biridir. Grafik işleme birimlerinin ve derin öğrenme algoritmalarının hızlı gelişimi, nesne algılama yöntemlerinin doğruluğunu ve etkinliğini önemli ölçüde artırmıştır. Bu bağlamda, You Only Look Once (YOLO) (Redmon vd., 2018) ve varyantları, Single Shot Multibox Detector (SSD) (Liu vd., 2015), Region-Based Convolutional Networks (R-CNN) (Girshick vd., 2016), Fast R-CNN (Girshick, 2015), Faster R-CNN (Ren, 2017) ve Mask R-CNN (He, 2017) gibi derin öğrenme tabanlı yöntemler, nesne algılama ve tanıma alanında geniş bir uygulama yelpazesi sunmaktadır. Bu modeller, yüksek verimlilikleri, doğrulukları ve hızlı işleme yetenekleri sayesinde, otonom sürüş, gözetim ve robotik gibi gerçek zamanlı uygulamalarda yaygın olarak tercih edilmektedir.

Görüntü tabanlı tespite ek olarak, zamansal bilgiler de nesne tespitinde önemli bir rol oynamaktadır. Zhang vd. (2019), Flow-Net (Dosovitskiy vd., 2015) kullanarak harici olarak hesaplanan optik akışı mevcut video görüntüsüne entegre eden ve bu akışları Long-Short Term Memory (LSTM) ile birleştiren çok akışlı bir yaklaşımı benimsemektedir. Spatio-Temporal Fusion (STF) tespit modeli (Anwar vd., 2024), video içinde nesne algılamayı iyileştirmek için derin bir mimaride dikkat modülleri kullanarak iki ardışık kareyi entegre etmektedir. Jazayeri vd. (2011), sürüş videolarında araç tespiti için hareket tabanlı bir izleme modülü kullanmaktadır. Corsel vd. (2023) ise zamansal bağlamdan yararlanmak için üç ardışık kareyi 3 kanallı bir görüntüde birleştirip YOLOv5 modeline beslemektedir. Heo vd. (2017), Appearance Net ve Motion Net mimarileriyle hareketli nesneleri tespit etmek için arka plan görüntüsüne sahip iki ardışık çerçeveyi kullanmaktadır. Ullah vd. (2018), yayaların bir çerçeve içinde uzamsal bölümlenmesi için ardışık üç çerçeveyi birleştirmektedir. Bu yöntemler, nesne tespitinin doğruluğunu değişen derecelerde artırmakta olup, herhangi bir ek nitelik veya bilgi sağlamaksızın yalnızca nesne tespitine odaklanmaktadır.

Nesne algılama modelleri, zamansal bilgiyi kullanarak nesne tespitini iyileştirmenin yanı sıra, çerçeveler aracılığıyla nesnelerin yönelimi gibi özellikleri de öğrenmek için eğitilmektedir. Bu sayede, nesne algılama modelleri nesnelerin hem varlığını hem de belirli özniteliklerini başarıyla tespit edebilirler. Cai vd. (2022) yayaların izlenen sınırlayıcı kutuları üzerinde bir model tasarlayarak yaya hareket izlerinin tahminini araştırmaktadır. Dafrallah vd. (2021), sürüş videolarındaki tek bir kareden

yayaları yönlerine göre sınıflandırmaktadır. Li vd. (2023)'de tek bir çerçeve modeli ile transformatör tabanlı bir çözüm önerilmiştir. Kim vd. (2022)'de araç yönelim tespiti için iki aşamalı bir yöntem önerilmiştir. Dominguez-Sanchez vd. (2017), statik bir kamera ile yaya hareketinin yönünü tespit etmek için çerçeveleri bir CNN modelindeki optik akışla birleştirir. Deokar vd. (2022), sadece çerçeve bilgilerini kullanarak yaya hareketini tespit etmek için özel bir CNN mimarisi kullanır. Monika vd. (2023) nesneleri tespit etmek ve hareket yönünü tanımak için sınırlayıcı kutuları izlemek için YOLOv5 kullanmaktadır. Yönelim veya hareket yönü, uzamsal özelliklerden sınıflandırılabilir; ancak otonom araçların karar verme süreçlerinde daha iyi performans elde etmek ve araç ile nesne arasındaki etkileşimi anlamak için zamansal analize de ihtiyaç vardır. Örneğin, sola bakan bir yayaya karşı, ayakta duran bir yayada veya karşıdan karşıya geçen bir yayada farklı hareket dinamikleri görülebilir. Zamansal veri analizi, sadece hareket profili görüntülerini kullanan çalışmalar tarafından da gerçekleştirilmektedir. Hareket profili görüntüleri ilk olarak Kilicarslan vd. (2014)'de ortam görselleştirmesi için tanıtılmıştır. Kilicarslan vd. (2019) çalışmasında, çarpışma zamanı (Time To Collision- TTC), nesne tespiti olmadan izlerdeki değişimin göreceli boyutunu ölçmek için yatay ve dikey hareket profilleri ile tahmin edilmiştir. Kilicarslan vd. (2023) tarafından yürüyen yayaları tespit etmek için birden fazla MPI kullanımını önerilmiştir. Cheng vd. (2022), araç kontrolü için hareket profillerini ardışık bir derin öğrenme modelinde işlemek için bir yöntem önermiştir. Lin vd. (2023) ise hareket profillerini araç boyutları ile birleştirerek YOLOv5 nesne dedektörü ile araç davranışlarını analiz etmeyi başardılar. Hareket profili görüntülerindeki semantik segmentasyondan, araçlar, kesme ve takip etme gibi ayrık sınıflara ve durma ve dönme gibi akıllı araç durumlarına sınıflandırılmıştır. Ancak, Lin vd. (2023) hala iki aşamalı bir yaklaşım kullanmaktadır: önce araçları çerçevelerde tespit edip, daha sonra bu tespitleri hareket profillerinde işlemektedir. Temel vd. (2024) ise yalnızca araçlar için hareket profili görüntülerinden hareket yönü bilgisini çıkarmaktadır. YOLOv3 tabanlı model, otoyol videolarında eğitilmiş ve test edilmiştir; bu videolar, arka plan karmaşıklığı açısından kentsel videolardan önemli ölçüde farklılık göstermektedir. Kilicarslan vd. (2022) ilk kez bir çerçeve ve hareket profili görüntüsünü araç tespiti için birleştirmeyi deneyen çalışmadır. Örnekleme bandının üst ve alt bölgeleri ile MPI ve çerçevesini birleştirerek uzamsal-zamansal bir görüntü oluşturur. Bu oluşturulan görüntü, daha sonra YOLOv3 nesne tespit modeline beslenerek araçların ve hareket yönlerinin üç

farklı yönde tespit edilmesini sağlamaktadır. Bu çalışma Temel vd. (2024) çalışmasını genişletir ve iki açıdan farklılık gösterir:

1) kentsel ve otoyol videolarında daha büyük bir MOT-BDD100K veri kümesinde eğitilmiş ve test edilmiştir ve 2) hareketle birlikte genel nesne tespiti için uçtan uca Two-Strea[M] modeli geliştirilmiştir.

3. SÜRÜŞ VİDEOLARINDA NESNE HAREKETİ

Önerilen yöntem, mevcut çerçeveye orta vadeli hareket verilerini entegre ederek nesne tespiti ve hareket analizini iyileştirmeyi amaçlamaktadır. Bu hareket verilerini elde etmenin bir yolu, MPI kullanmaktır. Bu bölümde, sürüş videolarından MPI oluşturma sürecinin detaylı bir açıklaması sunulmakta ve bu yöntemin, tüm sürüş videosunu işleme yöntemlerine kıyasla sağladığı avantajlar vurgulanmaktadır. Bu bölümde ayrıca, MPI'nin hareket bilgilerini nasıl topladığı ve bu bilgilerin nesne tespitinde nasıl kullanıldığı üzerinde durulacaktır. MPI'nin entegrasyonu, aracın çevresindeki nesneleri ve bu nesnelerin hareketlerini doğru bir şekilde anlamasını sağlayarak, akıllı araçların güvenli ve verimli bir şekilde çalışmasına katkıda bulunur. Bu yöntem, gelecekteki araştırmalar ve uygulamalar için de yeni perspektifler sunmakta olup, otonom araç teknolojilerinin gelişimine önemli katkılar sağlamaktadır.

3.1. Hareket Profili

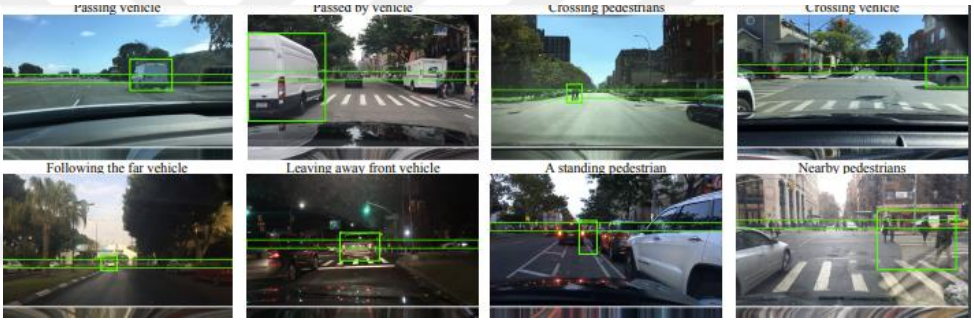
Bir aracın ön kamerası tarafından kaydedilen görüntüden elde edilen hareket profili, ufka yakın bir sürüş videosunun özetini oluşturur. Ufuk çizgisine yakın bölgelerde, yolun ve diğer araçların hareketi en net şekilde gözlemlenebilir. Bu nedenle, MPI bu bölgeden alınarak daha doğru, anlamlı ve sürüş sırasında en çok ilgilenilen hareket verileri elde edilir. Bu veriler, aracın hareket yönü ve hızını belirlemek için daha uygun olup, hareketin zaman içindeki değişimini (spatio-temporal changes) daha doğru bir şekilde analiz etmeyi sağlar. Ufuk çizgisinin yukarısındaki veya aşağıdaki bölgeler, hareketin daha az belirgin olduğu veya daha fazla gürültü (noise) içeren alanlardır. Daha yukarısı, genellikle gökyüzü veya uzak nesneler gibi sürüşle doğrudan ilgili olmayan bilgileri içerir. Daha aşağısı ise genellikle aracın önündeki kısa mesafeyi ve araç ön kısmını kapsar, bu da hareket algısı için yeterli bilgi sağlamaz.

MPI oluşturulurken şu adımlar takip edilir: Hesaplanan ufuk çizgisinin üzerine dh yüksekliğinde yatay bir kuşak (bant) yerleştirilir. Ardından, bu yatay kuşak içindeki piksellerin her bir karesinde dikey ortalaması alınır. Ardışık her karede, bu 1 boyutlu dikey ortalamalar dizisi, önceden tanımlanmış sayıda (dt) kare üzerinde birleştirilir. Bu işlem sonucunda, belirlenen video dizisi için 2 boyutlu bir MPI görüntüsü elde edilir, MPI (x, t).

$$MPI(x,t) = \prod_{k=t-dt}^{t} \sum_{j=h-dh}^{h} \frac{I(x,j,k)}{dh} \tag{3.1}$$

Burada; *I(x,j,k)* video karesinin piksel değerini, *h* ufuk çizgisinin yüksekliğini, *dh* kullanılan bant genişliğini, *dt* MPI'de kullanılan geçmiş karelerin sayısını, ∏ dikey birleştirme operatörünü ifade eder.

Çeşitli video dizileri için hareket profili görüntüleri Şekil 3.1'deki gibidir. MPI, ufuk çizgisine yakın bir bölgede dikey olarak örnekleme yaparak elde edilir. Bu örnekleme sayesinde tüm derinliklerdeki nesnelerin MPI üzerinde izleri oluşur. Arka plan, genişleme odağından dışa doğru genişlerken, statik nesnelerin izleri arka plan izlerine yaklaşır. Yavaş hareket eden nesneler dikey izler olarak gözlemlenirken, dönen bir araç dönüş yönüne doğru izler oluşturur. Arka plan izleri ise durmuş bir aracın üzerinde dikey olarak belirir. Bu yöntem, sürüş videosundaki çeşitli olayları ve nesnelerin hareketini özetlemek için kullanılır.



Şekil 3.1. *Çerçevenin alt kısmında karşılık gelen hareket profili birleştirme*

Uzun vadeli (>> 1 sn.) hareket profili görüntüsü, nesnelerin uzun vadeli geçmişinin gözlemlenmesini sağlar, ancak bu geçmiş doğrudan mevcut araç kontrol kararlarıyla ilişkili değildir. Bu tür görüntüler, gerçek zamanlı işlemeden ziyade uzun vadeli trafik analizi veya düzenli örüntü analizi için daha uygun olmaktadır. Ancak akıllı araçlar için en kritik zaman dilimi, kontrolle ilgili kararlar için son birkaç eylemdir, genellikle yaklaşık ~ 1 saniye içinde gerçekleşir (Knoblauch vd., 1996). Örneğin, ortalama bir yaya yürüme adımı ~ 1 saniye, bir araç şerit değiştirme olayı ise ~ 3 saniye sürmektedir (Ataelmanan vd., 2021). Kazalar genellikle aniden, bir saniyeden daha kısa sürede gerçekleşir. Bu nedenle, bu çalışmada, nesnenin hareketini analiz etmek için orta vadeli zaman aralıkları (~ 1 saniye) dikkate alınmıştır.

3.2. Sürüş Videolarında Hareketi Hesaplama

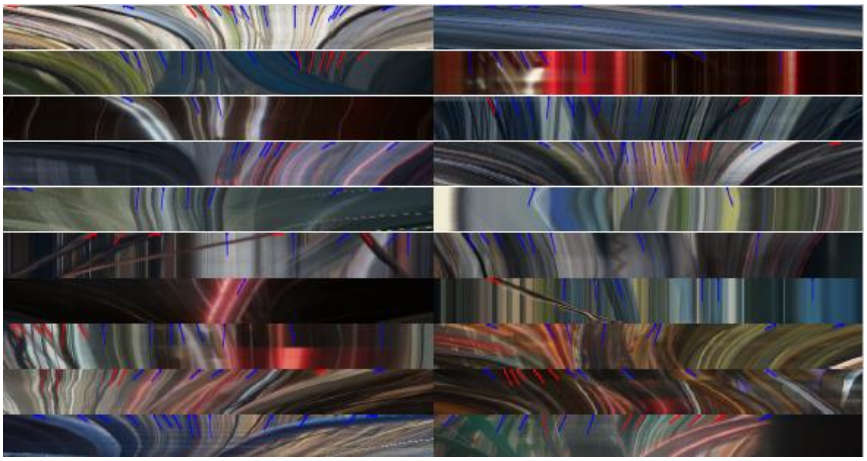
Sürüş videolarında hareketi hesaplamak, nesnelerin hızlarını ve yönlerini doğru bir şekilde belirlemek için kritik öneme sahiptir; bu süreç, çeşitli sensörlerden elde edilen verilerin analiz edilmesini gerektirir. Örnek olarak 3 boyutlu lidar sensörlerinden elde edilen veriler çevredeki nesnelerin üç boyutlu yapısını, konumunu çok hassas bir şekilde belirleyip, doğrudan hız hesaplanabilir ancak zamansal çözünürlükleri kameralara göre düşüktür. Lidar sensörleri genellikle 10 Hz (saniyede 10 kare) hızında veri toplarken, kameralar 30 Hz (saniyede 30 kare) veya daha yüksek hızlarda çalışabilir. Bu nedenle, nesnelerin tespiti ve takibi için hareketin sürekli ve detaylı bir şekilde izlenmesi gereken uygulamalarda yeterince uygun olmayabilir. Kameralar, daha yüksek zamansal çözünürlükleri sayesinde bu tür uygulamalarda daha avantajlı olmaktadır.

Önerilen yöntemin, nesneleri ve nesnelerin hareketini tespit edebilmesi için kullanılacak veri setinde sınırlayıcı kutulara ek olarak her nesneye ait hareket bilgisini içermelidir. Mevcut bilgilere göre, kamuya açık büyük bir sürüş videosu veri setinde nesnelerin hareketlerini doğru bir şekilde gösteren veriler bulunmamaktadır. Bu nedenle, pratik bir çözüm olarak, nesnelerin çerçeve bazlı sınırlayıcı kutularını kullanarak gerekli hareket verileri üretilmiştir. Veri setinde izlenen her nesne için, nesnelerin merkezlenmiş hareketini hesaplamada 3.2. formül kullanılmıştır.

$$M = \underset{0<t0\leq N}{argmax} \frac{xc(t) - xc(t - t0)}{t - t0} \tag{3.2}$$

Burada; $xc(t)$ t zamanındaki sınırlayıcı kutunun merkezini, $xc(t)$ ve $t - t0$ zamanındaki sınırlayıcı kutunun merkezini, N MPI'nin uzunluğundan kısa olan önceki çerçevelerin maksimum sayısını ifade eder, hareket hesaplaması $t{-}N$ inci çerçevede başlatılır.

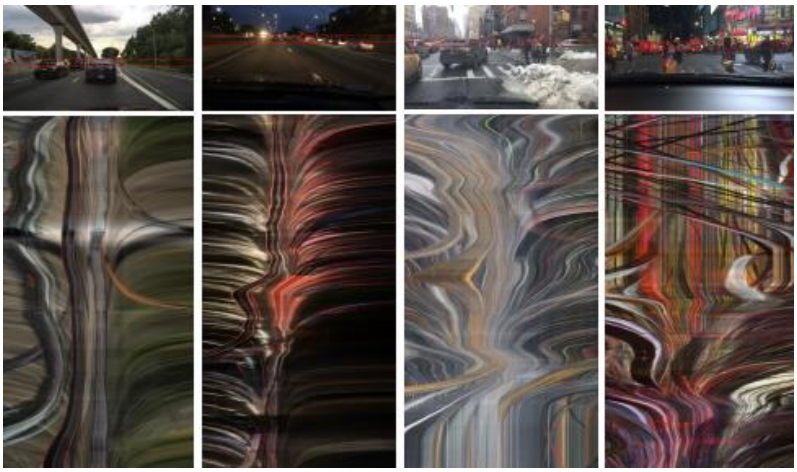
Sınırlayıcı kutu başlangıç çerçevesinde bulunamazsa, süreç bir sonraki çerçevelerde bulunana kadar tekrarlanmaktadır. Şekil 3.2, hareket profili görüntüleri ve üzerindeki nesne hareketlerini göstermektedir. Renkler nesne sınıflarını temsil eder: kırmızı yaya sınıfını, mavi ise araç sınıfını gösterir. Çizgilerin yönelimi, hareket değerlerini belirtir. Şekil, MPI ile hesaplanan hareket vektörlerini sunar ve bu hareket nesnenin ortalama hareketini gösterir. Bu yaklaşım, nesnenin sınırlayıcı kutusunu tek bir hareket değeriyle ilişkilendirir. Ancak hareket, her piksel için enterpolasyon yapılarak nesnenin sol ve sağ sınırlarından da hesaplanabilir, böylece nesne üzerindeki her piksel için hareket vektörleri elde edilebilir.



Şekil 3.2. *Hareket profili görüntüleri ve üzerlerine yapıştırılmış nesne hareketleri*

3.3. Hareket Profili Görüntüsünün Karmaşıklığı

Hareket profili görüntüsü sürüş sırasında nesnelerin nasıl hareket ettiğini görsel olarak temsil eden ve bu hareketlerin analiz edilmesine yardımcı olan bir görüntüleme aracıdır. Kentsel ve otoyol ortamları, farklı karmaşıklık seviyelerine sahiptir. Kentsel ortam, binalar, altyapı ve kalabalıklar nedeniyle genellikle otoyollardan daha karmaşıktır. Şekil 3.3, gündüz ve gece sürüş koşullarında kentsel ve otoyol ortamlarındaki değişen karmaşıklıkları göstermektedir. Arka plan karmaşıklığı, soldan sağa doğru otoyoldan kentsel ortama geçişle artmaktadır. Görseller sırasıyla gündüz otoyol, gece otoyol, gündüz kentsel ve gece kentsel olarak yer almaktadır.



Şekil 3.3. *Farklı sürüş senaryoları için hareket profili görüntü örnekleri*

Hareket profili görüntülerindeki yapıların karmaşıklığı, gündüz otoyolundan gece kentsel ortamlara kadar değişmektedir. Bu çeşitlilik, özellikle arka plan, araçlar ve yayalar gibi farklı hedef nesneler arasındaki ayrımın zorluğunu artırmaktadır. Bu durum, sürüş senaryolarında MPI'nin kullanımının yalnızca hareket profillerine güvenmekten öteye geçen kapsamlı bir yaklaşım gerektirdiğini ortaya koymaktadır. Bu zorluklar, MPI'nin orta veya uzun vadeli olma durumundan bağımsız olarak karşımıza çıkmaktadır.

Sonuç olarak, tek bir kare ile yapılan anlık nesne tespiti bazı durumlarda yeterli olabilir ancak MPI gibi zamansal verilerin entegrasyonu, özellikle karmaşık sürüş ortamlarında nesnelerin dinamik davranışlarını kapsamlı bir şekilde anlamak için önemli bir araçtır. Bu çalışma, MPI'nin, karmaşık sürüş ortamlarında nesne hareket dinamiklerini anlama konusundaki kritik rolünü vurgulamaktadır.

4. ÖNERİLEN YÖNTEM OLARAK NESNE VE HAREKET TESPİTİ İÇİN İKİ AKIŞLI TWO STREA[M] MODELİ

YOLO (You Only Look Once) evrişimsel sinir ağları kullanarak nesne tespiti yapan bir algoritmadır. Geleneksel nesne tespiti yöntemlerinin aksine, YOLO, tüm görüntüyü tek bir ağ geçişinde ele alarak nesne tespitini büyük ölçüde hızlandırır. Bu tek aşamalı yaklaşım, görüntü üzerinde birden fazla sınırlayıcı kutu (bounding box) ve sınıf etiketini aynı anda tahmin eder. YOLO'nun bu hızlı ve verimli tasarımı, gerçek zamanlı uygulamalarda kullanımını mümkün kılmaktadır.

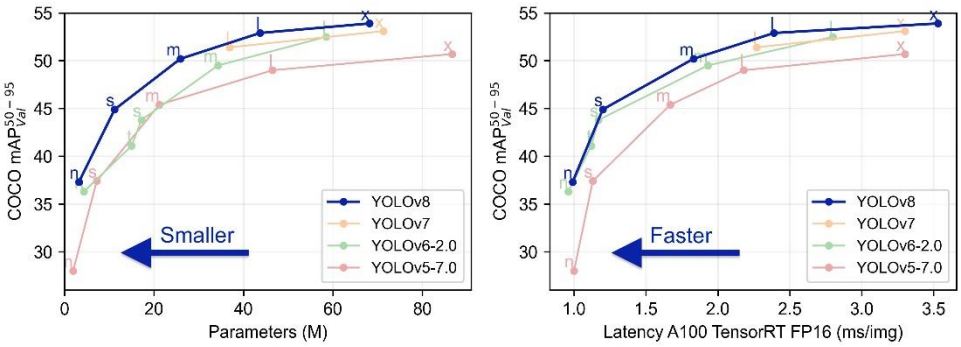
YOLO algoritmasının ana süreçleri:

- Giriş görüntüsünü N×N boyutlarında ızgaralara böler (örneğin, 5×5, 9×9).
- Her ızgara, içerdiği alanda nesne olup olmadığını kontrol eder ve nesnenin merkezinin bu ızgarada olup olmadığını belirler.
- Nesne merkezini tespit eden ızgara, nesnenin sınıfını, boyutlarını tahmin eder ve çevresine bir sınırlayıcı kutu çizer.
- Aynı nesne için birden fazla ızgara sınırlayıcı kutu çizebilir, bu da görüntüde gereksiz kutuların oluşmasına neden olabilir.
- Non-Maximum Suppression (NMS) algoritması çakışan sınırlayıcı kutular arasında en yüksek güven skoruna sahip olanı seçer ve diğerlerini kaldırır, böylece daha net ve doğru bir tespit sağlar.

YOLO algoritması, çeşitli sürümlerle evrim geçirmiştir. Bu sürümler, her defasında yeni özellikler ve iyileştirmeler sunarak modelin performansını artırmayı amaçlamıştır. YOLOv8 (Jocher vd., 2023), bu evrimin sekizinci sürümüdür ve önceki sürümlerine göre birkaç önemli avantaj sunmaktadır. Şekil 4.1, YOLO algoritmalarının farklı sürümlerinin (YOLOv5-7.0, YOLOv6-2.0, YOLOv7 ve YOLOv8) performansını iki farklı metriğe göre karşılaştırılmasını sunmaktadır. Buna göre YOLOv8, daha az parametre ve daha düşük gecikme süresi ile daha yüksek doğruluk sağlayarak diğer YOLO sürümlerine kıyasla hem daha verimli hem de daha hızlı çalışmaktadır.

- Gelişmiş Performans ve Doğruluk: YOLOv8 yüksek doğruluk oranları sunar ve karmaşık görüntülerde nesne tespitini daha iyi bir performansla gerçekleştirir.
- Hız ve Verimlilik: Yüksek işlem hızları ve düşük gecikme sürelerine sahiptir.

- Ankorsuz (Anchor-Free) Yaklaşım: Önceki sürümlerden farklı olarak, ankorsuz (anchor-free) bir nesne tespit yaklaşımına sahiptir. Geleneksel YOLO modelleri, nesneleri tespit etmek için önceden tanımlanmış ankraj kutuları (bounding boxes) kullanırken, YOLOv8 bu gereksinimi ortadan kaldırır. Bunun yerine, model doğrudan sınırlayıcı kutuları ve nesne skorlarını tahmin eder. Bu yaklaşım, daha esnek ve ölçeklenebilir bir tespit süreci sağlar.

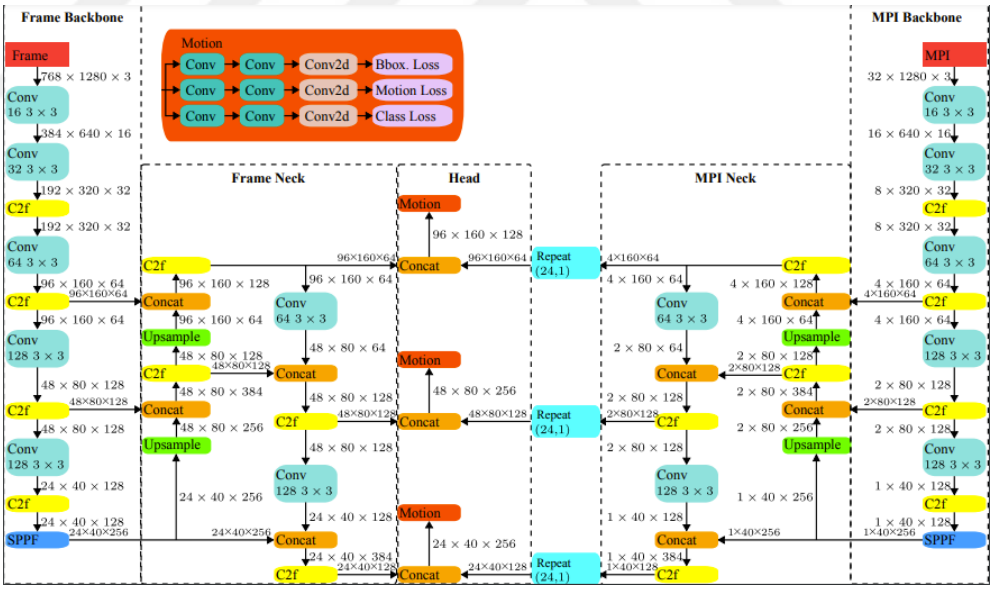


Şekil 4.1. *YOLO algoritmalarının farklı sürümlerinin performansı*

Bu çalışmada, YOLOv8 modelinin yeniden yapılandırılmış bir versiyonu olan Two-Strea[M] YOLOv8 geliştirilmiştir. Bu yeni model, giriş katmanına hareket akışı ekleyerek ve nesneleri tespit etmek için sınırlayıcı kutuların hareketle algılanmasını sağlayan bir tespit katmanıyla değiştirilerek geliştirilmiştir. YOLOv8, farklı boyutlarda sunulan bir modeldir: nano (n), küçük (s), orta (m), büyük (l) ve ekstra büyük (x). Two-Strea[M] YOLOv8 modeli, bu boyutlar arasından nano (n) boyutuna dayanmaktadır. Böylece daha az hesaplama gücü ve bellek gereksinimi, modelin daha hafif ve hızlı çalışmasının sağlanması, belirli uygulamalar için yeterli hassasiyet ve doğruluk oranı ile avantaj sağlamaktadır. Aşağıda, TwoStrea[M] için YOLOv8 modelinde yapılan değişiklikler listelenmiştir.

Girdi boyutları: Orijinal YOLOv8 modeli, eğitim sırasında yalnızca kare görüntüleri girdi olarak kabul eder. Video görüntüsü için, dikdörtgen görüntüler sorun oluşturmaz çünkü görüntülerin yüksekliği ve genişliği benzer ölçeklerdedir. Bu nedenle, görüntüyü yeniden boyutlandırdıktan sonra nesnelerin özellikleri bozulmaz. Ancak, MPI akışı için durum farklıdır. MPI görüntüsünün yüksekliği, genişliğinden önemli ölçüde daha küçüktür ve yalnızca ufuk çevresindeki hareket verilerini içerir. Kare görüntüler

kullanıldığında, MPI akışında gereksiz hesaplamalara neden olur. Kare çerçeve ve MPI akışlarını girdi olarak kullanmak, görüntünün merkezi zorunlu olarak ufuk olmayabileceğinden, uzamsal ve zamansal ilişkileri bozabilir. MPI görüntüsünü yeniden boyutlandırmak, girdi katmanında hareket verilerinin kaybına neden olabilir çünkü en-boy oranının değiştirilmesi, hareket değerlerini etkileyebilir. Bu nedenle, YOLOv8 modeli eğitim aşamasında dikdörtgen biçimli görüntüleri kabul edecek şekilde ayarlanmıştır. Böylece her iki akışta da farklı çözünürlükler kullanılabilmektedir: çerçeve akışı $h_F \; x \; w \; x \; 3$ ve hareket akışı $h_M \; x \; w \; x \; 3$, burada h_F, h_M ve w 32 ile bölünebilir ve $h_M \ll h_F$ 'dir. Şekil 4.2'de Two-Strea[M] YOLOv8 mimarisi, iki özdeş akıştan oluşur. MPI boyun (neck) katmanı, çerçeve (Frame) boyun ile uyumlu hale getirmek için çözünürlüğü eşleştirecek şekilde yeniden düzenlenir. Çerçeve ve MPI akışlarının çözünürlükleri sırasıyla 768 × 1280 × 3 ve 32 × 1280 × 3'tür. Çerçeve (Frame) Backbone katmanlarının ağırlıkları, transfer öğrenme için ImageNet veri kümesinde önceden eğitilmiş olup, sadece bu katmanlar dondurulmuş (freeze) olarak kullanılmıştır.



Şekil 4.2. *Two-Strea[M] YOLOv8 mimarisi*

Zamansal Özelliklerin Tekrarlanarak Uzamsal Özelliklerle Birleştirilmesi: Çoklu akışlı derin öğrenme yöntemlerinde, zamansal ve uzamsal özelliklerin birleştirilmesi genellikle iki temel strateji ile gerçekleştirilir: erken birleştirme ve geç birleştirme. Her iki yaklaşımın avantajları ve dezavantajları bulunur. Örneğin Erken Birleştirme

stratejisinde giriş akışları modelin başlangıcında birleştirir. Ancak bazı dezavantajları bulunmaktadır:

1) Büyük veri setlerinde, önceden eğitilmiş ağırlıkların kullanılamaması modelin sıfırdan eğitilmesini gerektirir. Bu durum, eğitim sürecini uzatır ve maliyetli hale getirebilir.
2) MPI çözünürlüğünün, video çerçevelerinin çözünürlüğü ile uyumlu hale getirilmesi gerekir. Uyumsuzluk durumunda, evrişim katmanlarında yapay ağırlıklar oluşabilir. Bu yapay ağırlıklar modelin performansını düşürebilir ve sonuçların gerçek veriyle uyumunu bozabilir.
3) Erken birleştirme stratejisinde, MPI üzerindeki aynı özellikler tekrar tekrar öğrenilir. Bu durum, modelin verimliliğini düşürebilir ve öğrenme sürecinin gereksiz yere uzamasına neden olabilir.

Erken birleştirme yönteminde, her algılama katmanı için MPI'nin çıktısı tekrarlanarak çerçeve görüntüsünün çıktısıyla birleştirilir. Tekrarlama miktarı, çerçeve ve MPI görüntü yüksekliklerinin oranına bağlıdır, *(k,1)* burada $k = \frac{h_F}{h_M}$ bir tam sayıdır. Bir çerçevedeki nesne, birkaç çerçeve boyun özellik haritasını kapsadığı sürece, video görüntüsü ve MPI akışlarının özellik düzeyinde aynı nesneye ait olmasını sağlar. Bu nedenle, video görüntüleri ve MPI akışlarında YOLOv8 mimarisi kullanılmıştır. Bu yaklaşım, önceden eğitilmiş ağırlıkların kullanımına olanak tanır ve modelin verimliliğini artırır.

Erken birleştirme stratejisinin sınırlamaları göz önüne alındığında, geç birleştirme stratejisi daha etkili bir seçenek olarak ortaya çıkar. Geç birleştirme, MPI ve çerçeve çıktılarının birleştirilmesi sırasında tekrarlama işlemini gerçekleştirir. Bu sayede model, uzamsal ve zamansal özellikleri daha verimli bir şekilde öğrenir. Bu yöntem, nesnelerin hareket ve konum bilgilerini doğru bir şekilde eşleştirmeyi ve izlemeyi mümkün kılar. Ayrıca, aynı mimarinin kullanılması modelin performansını artırırken, önceden eğitilmiş ağırlıkların kullanımına olanak tanır; bu da eğitim sürecini hızlandırır ve doğruluğu artırır.

Hareket Duyarlı Birleşimlerin Kesişimi (Intersection over Union - IoU): Bu çalışma, yönlendirilmiş sınırlayıcı kutular (Oriented Bounding Boxes - OBB) kullanarak nesne tespiti yapan modellerle benzerlik göstermektedir. YOLOv8 modeli, nesneleri doğru bir şekilde tespit etmek için Görev Uygunlaştırma Atayıcı (Task Alignment Assigner - TAL) adlı bir yöntemi kullanır (Feng vd., 2021). Bu yöntem, sınırlayıcı kutuları (anchor boxes) dikkatlice atayarak modelin doğruluğunu artırır.

Nesne tespitinde kullanılan temel metriklerden biri IoU'dur. IoU, tahmin edilen sınırlayıcı kutu ile gerçek sınırlayıcı kutu arasındaki örtüşme oranını ölçer. Bu metrik, genellikle genelleştirilmiş IoU veya tam IoU gibi varyantları içerir. Bu metrikler, modelin tespit başarısını değerlendirmek için kullanılır.

Eğitim sırasında, modelin doğru sınırlayıcı kutuları öğrenmesi hedeflenir. Bu öğrenme sürecinde, sınırlayıcı kutuların hareket yönelimi de dikkate alınır. Hareket yönelimi, nesnenin hareket ettiği yönü belirten bir açı olup derece cinsinden ifade edilir. Bu açılar -90˚ ile 90˚ arasında değişir. Hareket yönelimi şu dönüşümle elde edilir:

$$M^{\circ} = arctan(M) \tag{4.1}$$

Burada; M, hareket yönelimi ile ilgili temel değer, M˚, derece cinsinden hareket yönelimi temsil eder.

Liu vd. (2017) tarafından önerilen Hareket Duyarlı IoU (mIoU) metrik, geleneksel IoU metriğinin bir uzantısıdır. mIoU, doğru sınırlayıcı kutu ile tahmin edilen sınırlayıcı kutular arasındaki farkı ölçen bir kosinüs uzaklığı kullanır. Formül şu şekildedir:

$$mIOU = IOU \times cos\ (M^{g} - M^{p}) \tag{4.2}$$

Burada; IoU, geleneksel Intersection over Union metrik değeridir, M^{g} , doğru sınırlayıcı kutunun hareket değeridir, M^{p} , tahmin edilen sınırlayıcı kutunun hareket değeridir. Derece cinsinden hareket değerleri -90° ile 90° arasında olduğundan, bu aralık döngüsellik sorunlarını ortadan kaldırır.

Hareket değerleri, eksen hizalı sınırlayıcı kutulardan (frame-level) ve MPI görüntülerinden öğrenilir. mIoU, yalnızca eğitim aşamasında kullanılırken, geleneksel IoU tahmin sırasında NMS algoritması için kullanılır. NMS, modelin tahmin ettiği sınırlayıcı kutular arasında en doğru ve güvenilir olanını seçmek için gereklidir. Bu süreç, modelin doğruluğunu artırırken, gereksiz veya fazla sayıda tahmin edilen kutuların önüne geçer.

Hareket Katmanı ve Kayıp Fonksiyonu: YOLOv8 tespit başlığı, nesne tanımlama işleminde iki farklı kola sahiptir. Birinci kol, nesne sınırlayıcı kutuların (x merkezi, y merkezi, genişlik, yükseklik) değerlerini tahmin etmek için kullanılırken, ikinci kol ise nesnenin hangi sınıfa ait olduğunu belirler.

Toplam kayıp, modelin performansını değerlendirmek için dört farklı kayıp teriminin bir kombinasyonunu içerir. Bu terimler aşağıdaki gibidir:

1) Box Loss: Nesne sınırlayıcı kutularının tahmin edilen değerleri ile gerçek değerleri arasındaki farkı ölçer.
2) Class Loss: Nesnenin tahmin edilen sınıfı ile gerçek sınıfı arasındaki farkı ölçer.
3) Distribution Focal Loss: Nesnenin sınıf dağılımı üzerindeki hatayı değerlendirir.
4) Motion Loss: Hareket değerlerinin tahmin edilen ve gerçek değerleri arasındaki hata.

Toplam kayıp şu şekilde tanımlanır:

$$M^{\circ} = \sigma 1 \times boxLoss \; + \; \sigma 2 \times classLoss + \sigma 3 \times DFLoss + \sigma 4 \times motionLoss \qquad (4.3)$$

Burada; σ1, σ2, σ3, σ4 hiper parametreler olup, her kayıp teriminin toplam kayıptaki ağırlığını belirler.

Hareket kaybı, tahmin edilen hareket değerleri ile gerçek hareket değerleri arasındaki ortalama kare hatayı ölçer. Hareket kaybının formülü şu şekildedir:

$$motionLoss \; = \frac{1}{N}\sum_{i=1}^{N}(M_i - M_i{}')^2 \times \; (2 - \; cos(M_i)) \qquad (4.4)$$

Burada; M_i gerçek hareket değeri veya ölçülen hareket parametresi, $M_i{}'$ modelin tahmin ettiği hareket değerini, $2 - cos\,(M_i)$ ağırlık fonksiyonunu temsil eder.

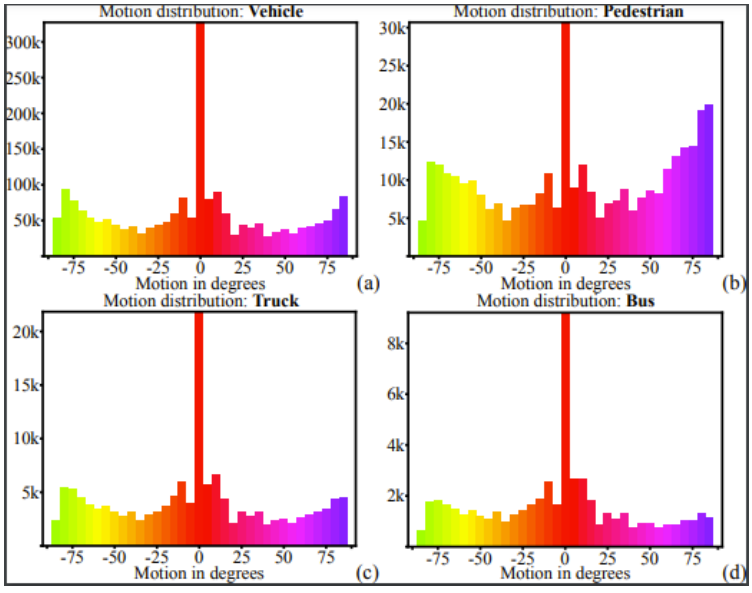
Bu formül, modelin tahmin ettiği hareket değerleri ile gerçek hareket değerleri arasındaki kaybı hesaplamak için kullanılır. Kayıp, her bir örnek için hareket değerlerinin farklarının karesini alarak hesaplanır ve bu değerler, hareketin belirli bir ölçeklendirme faktörüne bölünerek normalleştirilir. Sonuç olarak, bu değerlerin ortalaması, genel model performansını ölçmek için kullanılır. Bu tür bir kayıp fonksiyonu genellikle hareket tahmini, hareket analizi veya benzeri uygulamalarda model performansını değerlendirmek amacıyla kullanılır.

5. DENEYSEL SONUÇLAR VE ANALİZ

5.1. Veri Seti ve Veri Hazırlığı

Bu tez çalışmasında nesne tespit ve nesne hareket tespit algoritmalarının performansını değerlendirmek amacıyla eğitim ve değerlendirme için MOT-BDD100K veri seti kullanılmıştır (Yu vd., 2020). Orijinal veri seti 1400 eğitim, 400 test ve 200 doğrulama dizisi içermektedir. Ancak, veri setinin test dizileri için gerçek veri seti (ground truth data) sağlanmamış olması, bu dizilerin doğru bir şekilde değerlendirilememesine neden olmuştur. Bu problemi çözmek için orijinal veri seti yeniden düzenlenmiştir. Başlangıçta 1400 olan eğitim dizisi, 1200 eğitim ve 200 test dizisi olarak bölünmüştür. Orijinal 200 doğrulama dizisi, test dizisi olarak kullanılmak üzere ayrılmış ve eğitim setinden çıkarılan 200 dizi doğrulama için kullanılmıştır. Bu düzenleme, doğru etiketlenmemiş test dizileri problemini çözmek ve modelin performansını daha güvenilir bir şekilde değerlendirmek amacıyla yapılmıştır.

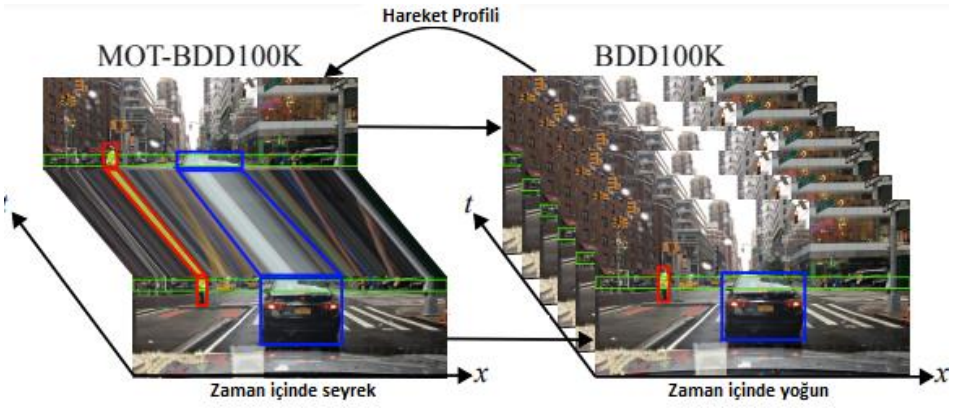
Nesnelerin hareketi, 3.2. formül kullanılarak hesaplanmakta ve 4.1. formül kullanılarak dereceye çevrilmektedir. Hareket hesaplamalarında kullanılan parametrelerin belirlenmesinde, akıllı araçlar ve nesnelerin boyutları göz önünde bulundurulmuştur. Deneylerde iki ana sınıfa odaklanılmıştır: araçlar ve yayalar. Bu nedenle, nesneler sadece bu sınıflar üzerinden değerlendirilmiştir. Sınıfa özgü hareket dağılımları Şekil 5.1'de görselleştirilmiştir. Tüm sınıflar pozitif ve negatif akışlarda simetri gösterse de oranları önemli derecede farklılık göstermektedir.



Şekil 5.1. *MOT-BDD100K veri setindeki sınıf düzeyindeki hareket dağılımları*

MOT-BDD100K veri seti yalnızca görüntüler içermekte olup, bu görüntüler BDD100K veri setindeki orijinal videolardan elde edilmiştir. MOT-BDD100K'nın temel farkı, görüntülerin 2Hz gibi düşük bir frekansta örneklenmiş olması ve izleme sınırlayıcı kutular (tracking-bounding boxes) içermesidir. Ancak bu düşük kare hızı, yüksek kaliteli hareket profili görüntüleri oluşturmak için bazı sınırlamalar getirmektedir. BDD100K veri seti ise, 30 ila 60 fps arasında değişen yüksek kare hızındaki videolar içermektedir. Bu videolar kullanılarak hareket profili görüntülerinin kalitesi önemli ölçüde artırılabilir. Ancak BDD100K videoları, kareler arasında nesne sınırlayıcı kutu ilişkileri sağlamamaktadır.

Bu sorunu çözmek için, MOT-BDD100K veri setinden görüntüler ve bu görüntülerin karşılık gelen hareket profili görüntüleri BDD100K veri setinden kullanılmıştır. Önerilen yöntemde her iki veri setinin nasıl kullanıldığını gösteren genel işlem akışı Şekil 5.2'de görselleştirilmiştir. MOT-BDD100K'daki ardışık görüntü çiftleri, BDD100K videolarında eşleştirilmiştir. Daha sonra, bu videolardan hareket profili görüntüleri oluşturulmuş ve görüntülerle ilişkilendirilmiştir. MOT-BDD100K'dan alınan görüntüler ve BDD100K'dan alınan hareket profili görüntüleri, Two-Strea[M] modeline girdi olarak verilmiştir. Bu yöntem, yüksek çözünürlüklü hareket profili görüntüleri için zamansal olarak yoğun veri oluşturulmasını sağlamaktadır. Sonuç olarak, MOT-BDD100K'dan alınan görüntüler ile BDD100K'daki karşılık gelen hareket profili görüntüleri kullanılarak hareket profili görüntülerinin kalitesi artırılmıştır.



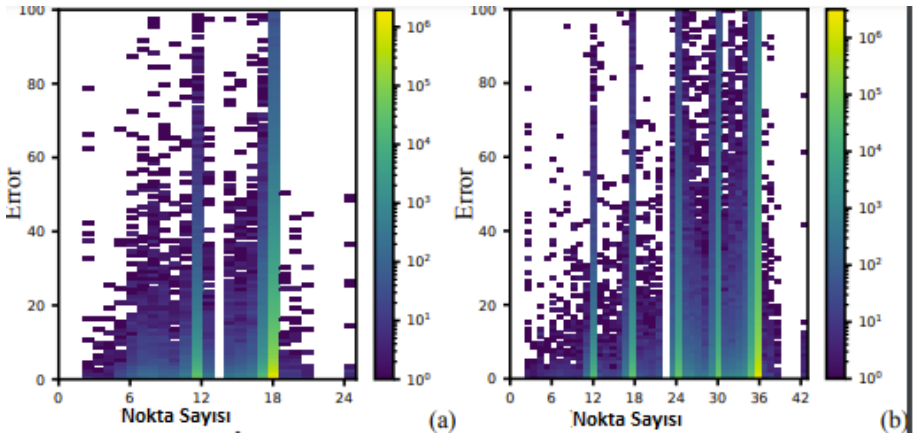
Şekil 5.2. *MOT-BDD100K'daki ardışık görüntü çiftleri, BDD100K videolarında eşleştirilmesi*

Araçların ve yayaların kısa bir süre boyunca (1 saniyeden az) doğrusal ve sabit hareket ettiği varsayılmaktadır. Bu varsayımın ve izleme uzunluğunun (N) etkisini anlamak için doğrusal bir çizgiye uydurma yöntemi kullanılmış ve elde edilen değerler ile gerçek veriler karşılaştırılarak ortalama mutlak hata hesaplanmıştır. Tüm eğitim örneklerinin ortalama mutlak hatası, sırasıyla N=18 ve N=36 için 3.61˚ ve 6.92˚ pikseldir. Şekil 5.3, hareket hatası (motion error) ve hareketi hesaplamak için kullanılan önceki noktaların sayısı (N) ile ilgili histogramı göstermektedir.

MOT BDD100K görüntüleri, BDD100K videolarından aynı zaman adımı boyutunu kullanarak (ortalama olarak her 6. karede bir) çıkarılmamıştır çünkü kare hızları tam sayı değildir. Bu uyumsuzluk, hareket hesaplamasında önceki kareler kullanıldığında küçük bir sapmaya yol açmaktadır. Bazı örneklerde, hareket N+6. karelerden hesaplanmaktadır. N arttıkça hem ortalama hata hem de iz başına hata oranı artmaktadır. Bu hata özellikle uzun izlerde, nispeten yavaş hareket eden nesnelerde veya zaman aralığı içindeki ani değişimlerde daha belirgin hale gelmektedir.

Her video için ufuk tahmini, her görüntüdeki nesne sınırlayıcı kutularının ortalama üst koordinatını hesaplayarak belirlenmiştir. Bu yöntem, ufuk hesaplama hatalarını azaltarak nesne ve hareket tespiti bölümlerinde daha iyi karşılaştırma metrikleri sağlar. Kemer boyutu olarak 45 piksel belirlenmiştir, çünkü bu boyut çerçevedeki yayalar ve araçlar için en geniş kapsama alanını sunmaktadır.

Videoların kare hızları, veri setinde önemli bir parametredir. Kare hızları 30 ila 60 fps arasında değişir ve bu, sonraki hesaplamaları önemli ölçüde etkiler. Bu parametre, hareket hesaplamasında kullanılan önceki karelerin sayısını (N) ve MPI oluşturulmasında kullanılan zaman adımını (dt) etkiler. MPI'nin yüksekliği değiştirilmeden 32 piksel olarak yeniden boyutlandırılmıştır. Bu nedenle, farklı fps değerlerine sahip videolar için hareket profili görüntüsünün yüksekliği sabit olarak 32 piksel tutulmuştur. Benzer şekilde, hareket hesaplamasında N değeri, N=18*fps/30 formülü ile belirlenmiştir.



Şekil 5.3. *Doğrusal hareket varsayımına bağlı hareket hatası histogramı*

Her iki durumda da hareket profili görüntüsü her zaman 1 saniyelik bilgiyi içerir ve hareket 0.5 saniye içinde hesaplanır. Bu durum hareket dinamiklerinin kapsamlı bir şekilde anlaşılmasını garanti etmektedir.

5.2. TwoStrea[M] Modelini Test Etme

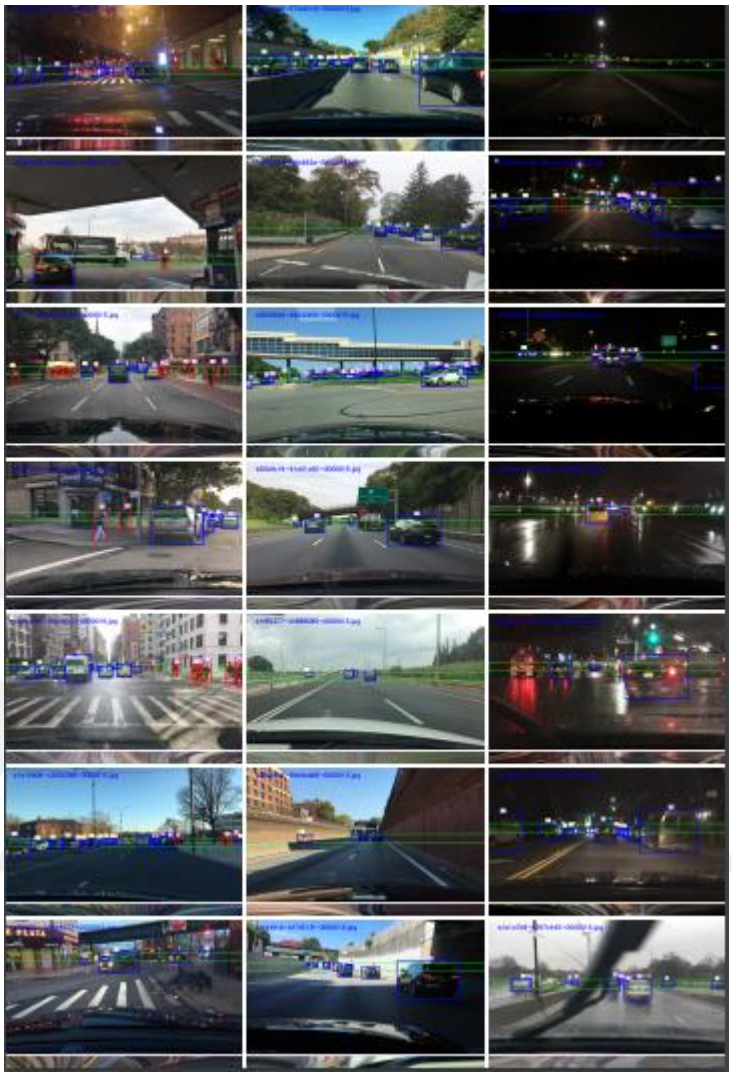
Eğitim için kullanılan girdi kaynakları, bir hareket profili görüntüsü ve bir video görüntüsüdür, model de Two-Strea[M] YOLOv8'dir. Şekil 5.4, daha genel senaryolarda elde edilen tespit sonuçlarını göstermektedir. Bu sonuçlar, gündüz ve gece, kentsel ve otoyol ortamlarında, kalabalık ve seyrek ortamlarda, farklı hava koşullarında çekilen videoları içerir. TwoStrea[M] modelinin eğitim ve değerlendirmesi, farklı hareket yönlerindeki tüm örneklerin eşit şekilde ele alındığından emin olmak için 2Hz örnekleme hızında gerçekleştirilmiştir. Ancak, test videosu orijinal video hızında üretilmiştir.

Hareket algılama, hareket yönündeki ani değişikliklere karşı hassas olmalıdır. Bu nedenle, önerilen yöntem, nesne örnekleme kuşağında olduğu sürece, tüm derinliklerdeki hızlı nesne hareketlerini içeren hareket değişikliklerine hızlı yanıtlar elde etmiştir. Bu hızlı yanıt, aracın olası çarpışmalardan kaçınmak için hızlı frenleme ve direksiyon hareketleri yapmasını sağlar.

Hem eğitim hem de test aşamasında, örtüşme ve kalabalık seviyelerine sahip sınırlayıcı kutular hariç tutulmamıştır. Hariç tutma kriterleri şunlardır:

(1) bir sınırlayıcı kutunun en az %50'sinin örnekleme kuşağı bölgesinde yer alması ve

(2) nesne yüksekliklerinin dh/2 = 22.5'ten büyük olmasıdır. Bu kriterler, hareket profili görüntüsünde nesne izinin görünürlüğünü sağlar. Böylece, yakından uzağa tüm mesafelerdeki nesneler tespit sonuçlarına dahil edilir.



Şekil 5.4. *Sürüş senaryolarında Two-Strea[M] modeli kullanılarak nesne ve hareket algılama sonuçları*

MPI hareket izleri, nesne sınıfından bağımsız olarak (araç ve kamyon gibi) benzer yapılara sahiptir. Bu model, hareket yönlerini başarılı bir şekilde tespit ederken, aynı zamanda benzer göreceli hareketlere sahip farklı nesne sınıfları için de başarı gösterir, bu da onu mevcut yöntemlerden ayıran bir özellik olmaktadır. Yağmurlu bir günde veya olumsuz hava koşullarında, silecek hareketiyle bile nesneler MPI'de tespit edilebilir. MPI'deki silecek nedeniyle oluşan yatay karanlık çizgiler, genel görüntüde yalnızca birkaç çizgiyi değiştirir. Ayrıca, yöntem, hareket profili görüntülerindeki yapısal benzerlikleri kesin bir şekilde ayırt etme konusunda başarılı sonuçlar verir; bu durum, yayaları veya araçları içeren nesnelerin görünüm özelliklerinin video karelerinde doğru bir şekilde yakalanmasından kaynaklanmaktadır. Bu yöntem, yalnızca hareket profili

görüntülerinde tespit etmeye kıyasla yanlış pozitif (false-positive) oranlarını önemli ölçüde ve etkili bir şekilde azaltır, bu da doğruluğunu kanıtlar. Ayrıca, veri setinde yalnızca farlarıyla görülebilen araçların bulunduğu gece videoları bulunmaktadır. Bu farlar, hareket profili görüntülerinde güçlü bir kontrast oluşturur ve gece sürüş sahnelerinde araçların ve hareketlerinin doğru bir şekilde tespit edilmesine yardımcı olur.

Şekil 5.5, bazı başarısızlık durumlarını ve yanlış pozitif tespit sonuçlarını özetlemektedir. Kamera yerleşimi, ufuk hesaplamasındaki hatalar ve geçici olarak tıkanan nesnelerden kaynaklanan birkaç başarısızlık durumu söz konusudur. Yanlış pozitifler, esas olarak dikey nesnelerin yayalar olarak tespit edilmesinden kaynaklanmaktadır. En bariz başarısızlık durumu, kamera eğimi nedeniyle farklı derinlikleri kaplayan örnekleme bandıdır. Başarısızlığın diğer nedenleri,
(1) MPI hareket verilerinin nesneye ait olmadığı ufuk hesaplama hataları ve
(2) hareket verilerinin bir veya daha fazla nesne ile karıştığı geçici olarak örtülen nesnelerdir. Önceden eğitilmiş TwoStrea[M] modeli ile hareketli nesne tespiti genel olarak %78 mAP ve 3.09˚ MAE elde etmiştir.

Şekil 5.5. *Başarısızlık durum örnekleri*

5.3. Temel Çalışmalar ve Karşılaştırmalar

Ubuntu 22.04 üzerinde Intel(R) i9(R) CPU @ 2.20GHz ve 2xRTX3060 12GB VRAM donanımlı bir makinede birkaç deney gerçekleştirilmiştir. Tüm deneylerde verileri ve modelleri GPU VRAM'ine sığdırmak için eğitim sırasında 10 epoch ve farklı batch boyutları kullanılmıştır. YOLOv8'de tüm varsayılan parametreler kullanılıp, hareket kaybı hiper parametresi 15 olarak ayarlanmıştır.

TwoStrea[M] YOLOv8 modelinde bir video görüntüsü ve MPI'de ortalama test süresi 110 fps'dir. Buna karşılık, yalnızca nesne tespiti için kullanılan YOLOv8n modeli 160 fps, izleme için ise 100 fps hızında çalışmaktadır. Bu nedenle, önerilen yöntem

izleme tabanlı modeller kadar hızlı çalışmaktadır. Ayrıca, video görüntüsündeki nesne sayısına bağlı olarak izleme süresi orantılı olarak artsa da yöntemin performansı bu durumdan etkilenmemektedir. Algılanan sınırlayıcı kutunun, doğru ile 0.5'ten büyük IoU varsa, doğru pozitif (true-positive) olarak değerlendirilir. Kesinlik ve duyarlılık metrikleri her zamanki gibi hesaplanır ve nesne tespiti için birincil metrik, ortalama hassasiyet (MAP@0.5)'tir. Tahmin edilen hareket değerleri MAE metriği ile değerlendirilir. Önerilen yöntem, performansının mevcut çerçevelerle ilişkili kapsamlı bir anlayış sağlamak amacıyla iki farklı yönüyle değerlendirilmiş ve karşılaştırılmıştır. İlk olarak, geleneksel YOLOv8 modeli ile karşılaştırılarak tespit performansını mAP metriğine dayanarak değerlendirilmiştir. İkinci olarak, hareket doğruluğu, Bytetrack (Zhang vd., 2019) ve Botsort (Zhang vd., 2021) ile karşılaştırılarak MOT-BDD100K veri setinde eğitilmiş YOLOv8 kullanılarak MAE metrikleri ile karşılaştırılmıştır.

Nesne algılama: Veri kümesini ve model karmaşıklığını anlamak için orijinal YOLOv8 ve Two-Strea[M] modelleri kullanılarak çeşitli eğitim deneyleri yapılmıştır. Tablo 5.1, nesne algılama deneylerinin özetini sunmaktadır. YOLOv8 modeli, önceden eğitilmiş ağırlıklar kullanılmadan araçlar ve yayalar için sırasıyla %77, %87 ve %67 mAP elde etmiştir. Ancak ImageNet veri setindeki önceden eğitilmiş ağırlıklar kullanıldığında mAP %80'e, araç ortalama hassasiyeti %89'a ve yaya ortalama hassasiyeti %71'e yükselmiştir. Omurga katmanlarının dondurulması bu ölçümleri önemli ölçüde değiştirmemiştir. Bu deney, beklendiği gibi veri kümesine daha fazla veri eklemenin doğruluğu artırdığını ve hatalı pozitifleri azalttığını göstermektedir.

Tablo 5.1. *YOLOv8n ve Two-Strea[M] nesne tespiti üzerine eğitim sonuçları*

	Model	Source	Dimension	Pre-train	Freeze	mAP	AP(V.)	AP(P.)
1	YOLOv8n	Frame	768 × 1280	-	-	%77	%87	%67
2	YOLOv8n	Frame	768 × 1280	✓	✓	%77	%87	%68
3	YOLOv8n	Frame	768 × 1280	✓	-	%80	%89	%71
4	YOLOv8n	MPI	32 × 1280	-	-	%52		
5	YOLOv8n	MPI	32 × 1280	-	-	%38	%57	%20
6	Two-Strea[M]	Frame MPI	768 × 1280 32 × 1280	✓ -	✓ -	%79	%89	%70

Hareket profili görüntülerinde nesne algılamaya bakmak için YOLOv8 modeli yalnızca MPI üzerinde eğitilmiştir. Yalnızca MPI'dan öğrenmek için araç ve yayanın iki sınıf olarak kullanılması, araçlar ve yayalar için sırasıyla %38, %57 ve %20 mAP ile sonuçlanır. Bu sonuç her iki sınıfın da özellikle yaya sınıfının sadece MPI'dan öğrenilemeyeceğini göstermektedir. Deneylerin ortaya koyduğu gibi, asıl zorluk, bu görüntülerde nesne sınıflarının kolayca ayırt edilememesidir, bu da tüm yayaları ve araçları tek bir sınıf olarak ele almaya yöneltmektedir. Bu nedenle bu deneyin mAP puanı %52 olarak elde edildi. Bu nedenle, bu iki deney MPI'daki nesne sınıfı belirsizliklerini göstermektedir.

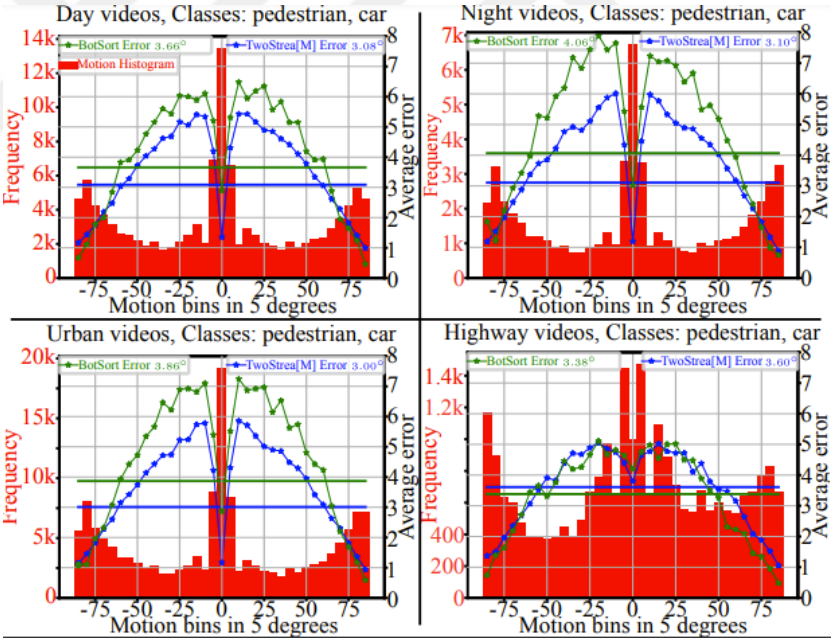
Son olarak, TwoStream[M] modeli, video görüntüsü ve MPI olmak üzere iki akışlı girişlerle ve yalnızca sınırlayıcı kutuları çıkış olarak kullanan bir tespit başlığıyla eğitildi. Model, ImageNet veri kümesinde önceden eğitilmiş ağırlıkları kullandı ve Çerçeve Ağı'nın katmanlarını dondurdu. Bu eğitimde, araçlar için %89 ve yayalar için %70 AP ile %79 mAP başarısı elde edildi. Bu sonuçlar, tek akışlı YOLOv8 modeliyle MOT-BDD100k veri kümesinde elde edilen sonuçlarla benzerlik göstermektedir. Modelin MPI'yi girdi olarak eklenmesi veri setinin karmaşıklığını artırmasına rağmen, mAP sonuçları YOLOv8 nesne tespit modelleriyle kıyaslanabilir düzeydedir.

Nesne ve hareket algılama: Yöntem, hareket hataları açısından nesne algılama ve takibi yöntemleri ile karşılaştırılmıştır. Bu amaçla iki izleme yöntemi seçilmiştir: Bytetrack ve Botsort. Tablo 5.2, hareket algılama hatalarının özetini sunmaktadır. Tüm deneylerde omurga modeli olarak YOLOv8 kullanılmıştır. Değerlendirme sırasında, profil oluşturma bandıyla %50'den daha az örtüşen sınırlayıcı kutular hariç tutulmuştur. Önerilen yöntem, araçlar ve yaya sınıfları için ortalama olarak 3,09˚ MAE, 3,09˚, 3,05˚ elde etmiştir. Bu sonuçlar, tespite dayalı yöntemlerle yapılan takipten daha iyi performans göstermektedir. Ayrıca nispeten basit ve küçük bir veri setinde rapor edilen hata ∼ 3˚'dir. Veri kümesindeki hareket değerleri arttıkça, hareket tahminleri de daha doğru olmaktadır. Sonuç olarak, önerilen yöntem, mevcut izleme yöntemlerine kıyasla daha yüksek doğruluk ve tutarlılık sağlamaktadır.

Tablo 5.2. *Two-Strea[M] ve tespit-izleme tabanlı hareket tespit sonuç karşılaştırmaları*

	Model	Source	Dimension	Pre-train	Freeze	MAE	MAE (V.)	MAE (P.)
1	Bytetrack	Frame	768 × 1280	MOT-bdd	-	4.02˚	3.99˚	4.34˚
2	Botsort	Frame	768 × 1280	MOT-bdd	-	3.79˚	3.73˚	4.28˚
3	Two-Strea[M]	Frame MPI	768 × 1280 32 × 1280	✓ -	✓ -	3.09˚	3.09˚	3.05˚

Farklı çevresel koşullar altında hata metriklerini detaylı analiz edilmiştir. Önerilen yöntem, gündüz, gece ve kentsel ortamlarda daha iyi performans gösterirken, otoyol videolarında BotSort daha düşük MAE skorlarına sahiptir. Şekil 5.6'da, her iki yöntemin farklı sürüş ortamlarındaki karşılaştırması görselleştirilmiştir.



Şekil 5.6. *Hareket hata histogramı ve 5˚'lik aralıklarda ortalama hata, (a) Two-Strea[M] (b) BotSort*

5.4. Ablasyon Çalışması

Ablasyon çalışması kapsamında, YOLOv8 modeli hem hareket kafası hem de TwoStrea[M] modeli hareket ve algılama kafalarıyla eğitilmiştir. Bu süreçte, önceden eğitilmiş (pre-trained) modeller ve dondurma (freeze) kombinasyonları kullanılmıştır. Dondurma seçeneği uygulandığında, yalnızca Çerçeve Omurgası (Frame Backbone)

dondurulur. İlk olarak, video görüntüsü geleneksel YOLOv8 mimarisi ile eğitilmiş, ardından TwoStrea[M] video görüntüsü akışında öğrenilen ağırlıklar kullanılmıştır. Dondurma seçeneğinin eklenmesi, öğrenilebilir parametrelerin sayısını önemli ölçüde azaltarak modelin aşırı uyum (overfitting) riskini düşürmüştür. Genel olarak, TwoStrea[M]'ye ikinci bir akışın eklenmesi, algılama süresini çerçeve başına 6 ms'den 9 ms'ye hafifçe artırmaktadır. Tablo 5.3, tek akışlı ve TwoStrea[M] YOLOv8 modellerinin sınırlayıcı kutu lokalizasyonunu mAP ve MAE değerleri açısından karşılaştırmaktadır. Bu tablo, farklı modellerin performansını net bir şekilde karşılaştırmaktadır.

Tablo 5.3. *YOLOv8n ve Two-Strea[M] hareket ve nesne algılama sonuçlarının ablasyon çalışması*

YOLOv8

	Model	Source	Dimension	Pre-train	Freeze	mAP	AP(V.)	AP(P.)	MAE
1	Motion	Frame	768 × 1280	-	-	%62	%78	%45	14.97˚
2	Motion	Frame	768 × 1280	✓	✓	%71	%83	%58	11.56˚
3	Motion	Frame	768 × 1280	✓	-	%69	%80	%58	16.62˚
4	Motion	MPI	32 × 1280	-	-	%50	-	-	3.11˚
5	Motion	MPI	32 × 1280	-	-	%37	%54	%20	3.00˚

TwoStrea[M] YOLOv8

6	Detect	Frame MPI	768 × 1280 32 × 1280	- -	- -	%77	%88	%66	-
7	Detect	Frame MPI	768 × 1280 32 × 1280	✓ -	✓ -	%80	%89	%70	-
8	Detect	Frame MPI	768 × 1280 32 × 1280	✓ -	- -	%79	%89	%70	-
9	Motion	Frame MPI	768 × 1280 32 × 1280	- -	- -	%73	%85	%61	3.30˚
10	Motion	Frame MPI	768 × 1280 32 × 1280	✓ -	✓ -	%77	%87	%68	3.29˚
11	Motion	Frame MPI	768 × 1280 32 × 1280	✓ -	- -	%78	%88	%69	3.09˚

YOLOv8 modeli hareket başlığıyla nesne tespitinde hala iyi sonuçlar vermektedir, ancak çok yüksek hareket hataları bulunmaktadır. Tek bir görüntü, hareketi etkili bir şekilde tespit etme yeteneğine sahip değildir. MPI kullanımı, hareket hatalarında en düşük MAE değerlerini üretse de nesne tespiti için mAP puanlarında istenilen kaliteyi sağlayamamaktadır. MPI'dan hareket bilgisi öğrenilebilir ancak nesne sınıfları MPI üzerinden doğrudan öğrenilemez. Bu sonuçlar ne görüntünün ne de MPI'nin tek başına nesne ve hareketi doğru bir şekilde tespit etmek için yeterli olmadığını göstermektedir.

İkinci modelde, yalnızca sınırlayıcı kutuları regresyon yapan tespit başlığıyla, en iyi sonuçlar MOT-BDD100K veri setinde önceden eğitilmiş ağırlıkların kullanılmasıyla ve herhangi bir katmanın dondurulmamasıyla elde edilmektedir. Ayrıca, Çerçeve omurgasının dondurulması da iyi sonuçlar vermektedir. Bu sonuçlar, %80, %89 ve %71 oranlarında olan çerçeve tabanlı YOLOv8 nesne tespitine oldukça benzerdir. Ayrıca, nesne tespiti için hareket başlığının kullanılmasıyla, en iyi sonuçlar önceden eğitilmiş bir model ve çerçeve omurga'nın dondurulmasıyla elde edilmektedir. Genel olarak, bu model nesne tespitinde mAP ve hareket tespitinde MAE metrikleri arasında iyi bir denge sağlamıştır.

6. TARTIŞMA

Çalışmada önerilen yöntem, mAP ve MAE ölçümleri açısından halka açık büyük bir veri kümesi üzerinde test edilmiştir. Yöntem, çerçeve tabanlı (frame-based) nesne dedektörü kadar doğru olduğunu doğrulamıştır. Ayrıca yöntem, TwoStrea[M] ağı orijinal YOLOv8 ağından iki kat daha kapsamlı olduğundan, video için kareler arası hareket algılamalarına göre daha az veri kullanır. Ayrıca birleşik bir derin öğrenme modelinde nesne ve hareket algılama sağlar. Yöntemin avantajları şunlardır:

(1) Yöntem, araç kontrolü için çok önemli olan yavaş veya hızlı olmasına bakılmaksızın nesne hareketini doğru bir şekilde algılayabilmektedir.

(2) Yöntem, hareket profili görüntülerindeki izlerle hareket değerlerini doğrulamak için yorumlanabilir bir sonuç üretir.

(3) Yatay kuşak nesnenin yarısını kapladığı sürece hareketli profil görüntülerinde yüksek kontrastlı izlere sahiptir. Ancak bazı kamera yerleşimleri nedeniyle bir nesnenin hareketi kısa zaman aralıklarında hatalı olarak algılanabilmektedir.

(4) Eğitim ve test sırasında kapatılan hiçbir nesne hariç tutulmamıştır. Bu seçim bazı durumlarda nesne algılama performansının düşmesine neden olabilir.

Sonuç olarak, nesnelerin anlık hareketle algılanması, onları hızlı bir şekilde güvenli ve tehlikeli seviyeler olarak sınıflandırabilir. İnsan gözü, sürüş sırasında yol yönüne odaklanarak zıt hareketlere dikkat eder. Nesnenin hareketi ve görünüm özelliklerinin tümü insanın dikkatini etkiler ancak hareket, nesnenin şeklinden daha önemlidir. Önerilen yöntem, hareketi geleneksel nesne algılama yöntemleriyle aynı doğrulukta bir öznitelik olarak algılar ve gerçek zamanlı yanıt verir. Bu yöntem, geleneksel video karesi tabanlı nesne algılama yöntemlerinin yerini alabilir ve eksiksiz bir sistem olarak araç kontrol sistemlerine entegre edilebilir.

7. SONUÇ

Bu çalışmada, alanında önemli bir ilerleme kaydeden yeni bir yaklaşım olan Two-Strea[M] uzamsal-zamansal nesne ve hareket tespit modeli tanıtılmıştır. Bu, sürüş videolarında çevre algısı için uçtan uca hareket farkındalığına sahip bir nesne tespit modelidir. Önerilen yöntem, izleme ve optik akış tabanlı yöntemlerden daha az veri kullanır. Tüm derinlik aralıklarında %78 genel mAP ve 3.09˚ MAE değerlerine ulaşır. Nesnelerin hareketleri, hareket profili görüntülerinde anında tespit edilir ve video kareleri üzerinde İki Akışlı model kullanılarak nesnelerin konumları belirlenip sınıflandırılır. Modelde aşırı öğrenmeyi (overfitting) önlemek ve nesnelerin hareketini daha etkili bir şekilde öğrenmek amacıyla transfer öğrenme (transfer learning) tekniği kullanılmıştır. Transfer öğrenme, mevcut bir modelin daha önce başka bir görevde öğrenilmiş bilgilerini yeni bir görevde kullanarak modelin eğitim sürecini hızlandırır ve performansını artırır. Bu durumda, transfer öğrenme çerçeve omurgasında (frame backbone) uygulanarak modelin hareket akışına (motion stream) odaklanması sağlanır. Böylece, modelin hareket tespitinde daha başarılı olması ve gereksiz detaylara odaklanmaması amaçlanır.

Model, tüm çevresel zorluklarla başa çıkabilmek için kentsel ve otoyol ortamlarını içeren büyük kamu video veri setlerinde eğitilmiştir. Yöntem, kalabalıklar, yüksek örtüşme ve çeşitli yol ortamları gibi zorlu senaryolarda nesneleri ve hareketi tespit ederek düşük yanlış pozitif oranları ve ortalama mutlak hareket hatası ile çok yönlülüğünü göstermektedir. Model, gerçek zamanlı hareket ve nesne tespiti sağlamak için iki görüntü ve sadeleştirilmiş mimari kullanır.

KAYNAKÇA

Aharon, N., Orfaig, R., & Bobrovsky, B.-Z. (2022). BoT-SORT: Robust associations multi-pedestrian tracking. *ArXiv*, abs/2206.14651. Retrieved from https://api.semanticscholar.org/CorpusID:250113384.

Akar, F., & Orman, K. (Eylül 2020). *Otonom Kara Araçlarındaki Görüş Sistemlerinin İncelenmesi*. Erzincan Binali Yıldırım Üniversitesi.

Ataelmanan, H., Puan, O. C., & Hassan, S. A. (2021). Examination of lane changing duration time on expressway. *IOP Conference Series: Materials Science and Engineering*, 1144(1), 012078.

Çevik, H. (2022). *Akıllı Ulaşım Sistemleri Unsurlarından Otonom Araçlarda Trafik Kaza Faktörlerinin Belirlenmesine İlişkin Bir Araştırma.* İstanbul Üniversitesi Fen Bilimleri Enstitüsü.

Cheng, G., & Zheng, J. Y. (2022). Sequential semantic segmentation of road profiles for path and speed planning. *IEEE Transactions on Intelligent Transportation Systems*, 23(12), 23869–23882.

Corsel, C. W., van Lier, M., Kampmeijer, L., Boehrer, N., & Bakker, E. M. (2023). Exploiting temporal context for tiny object detection. *In 2023 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)* (pp. 1–11). IEEE.

Dafrallah, S., Amine, A., Mousset, S., & Bensrhair, A. (2021). Monocular pedestrian orientation recognition based on capsule network for a novel collision warning system. *IEEE Access*, 9, 141635–141650.

Deokar, S., & Khandekar, S. (2022). Identification of pedestrian movement and classification using deep learning for advanced driver assistance system. In Proceedings - *International Conference on Augmented Intelligence and Sustainable Systems* (ICAISS 2022) (pp. 374–381).

Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., v. d. Smagt, P., Cremers, D., & Brox, T. (2015). FlowNet: Learning optical flow with convolutional networks. In 2015 *IEEE International Conference on Computer Vision (ICCV)* (pp. 2758–2766). IEEE.

Feng, C., Zhong, Y., Gao, Y., Scott, M. R., & Huang, W. (2021). TOOD: Task-aligned one-stage object detection. *In 2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (pp. 3490–3499). IEEE. Retrieved from https://ieeexplore.ieee.org/document/9710724.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2016). Region-based convolutional networks for accurate object detection and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1), 142–158.

Girshick, R. (2015). Fast R-CNN. *In 2015 IEEE International Conference on Computer Vision (ICCV)* (pp. 1440–1448). IEEE.

Gürtaş, S. (2020). *Otonom Araç Sürüş Destek Sistemleri ve Yapay Zeka Uygulamaları*. Bursa Uludağ Üniversitesi Fen Bilimleri Enstitüsü, Bursa.

Heo, B., Yun, K., & Choi, J. Y. (2017). Appearance and motion based deep learning architecture for moving object detection in moving camera. *In 2017 IEEE International Conference on Image Processing (ICIP)* (pp. 1827–1831). IEEE.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2), 386–397.

Jazayeri, A., Cai, H., Zheng, J. Y., & Tuceryan, M. (2011). Vehicle detection and tracking in car video based on motion model. *IEEE Transactions on Intelligent Transportation Systems*, 12(2), 583–595.

Jocher, G., Chaurasia, A., & Qiu, J. (2023). *Ultralytics YOLO*. Retrieved from https://github.com/ultralytics/ultralytics.

Kilicarslan, M., & Zheng, J. Y. (2014). Visualizing driving video in temporal profile. *In 2014 IEEE Intelligent Vehicles Symposium Proceedings* (pp. 1263–1269). IEEE.

Kilicarslan, M., & Zheng, J. Y. (2019). Predict vehicle collision by TTC from motion using a single video camera. *IEEE Transactions on Intelligent Transportation Systems*, 20(2), 522–533.

Kilicarslan, M., & Zheng, J. Y. (2023). DeepStep: Direct detection of walking pedestrian from motion by a vehicle camera. *IEEE Transactions on Intelligent Vehicles*, 8(2), 1652–1663.

Kilicarslan, M., & Temel, T. (2022). Motion-aware vehicle detection in driving videos. *Turkish Journal of Electrical Engineering and Computer Sciences*, 30(1), 63–78.

Kim, Y., Kang, D., & Baek, H. (2022). A 2-stage model for vehicle class and orientation detection with photo-realistic image generation. In Proceedings - *2022 IEEE International Conference on Big Data (Big Data 2022)* (pp. 6489–6493).

Knoblauch, R. L., Pietrucha, M. T., & Nitzburg, M. (1996). Field studies of pedestrian walking speed and start-up time. *Transportation Research Record: Journal of the Transportation Research Board*, 1538(1), 27–38.

Li, X., Ma, S., Shan, L., & Li, X. (2023). Multi-window transformer parallel fusion feature pyramid network for pedestrian orientation detection. *Multimedia Systems*, 29(2), 587–603.

Liu, L., Pan, Z., & Lei, B. (2017). Learning a rotation invariant detector with rotatable bounding box. *arXiv preprint arXiv*:1711.09405.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2015). SSD: Single shot multibox detector. *In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 9905 LNCS, pp. 21–37).

Monika, Singh, P., & Chand, S. (2023). Computer vision-based framework for pedestrian movement direction recognition. *Journal of Intelligent & Fuzzy Systems*, 44(5), 8015–8027.

Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. *arXiv*:1804.02767 [cs]. Retrieved from http://arxiv.org/abs/1804.02767.

Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6), 1137–1149.

Temel, T., Kilicarslan, M., & Hoscan, Y. (2024). SDA: A novel skewed-deep-architecture for vehicle motion detection in driving videos. *Kahramanmaras Sütçü İmam Üniversitesi Mühendislik Bilimleri Dergisi*, 27(1), 92–104.

Türkiye İstatistik Kurumu (TUİK). (2023). *Karayolu Trafik Kaza İstatistikleri*, 2023. Erişim adresi: https://data.tuik.gov.tr/Bulten/Index?p=Karayolu-Trafik-Kaza-Istatistikleri-2023-53479 (Erişim Tarihi: 01.06.2024).

Ullah, M., Mohammed, A., & Cheikh, F. A. (2018). PedNet: A spatio-temporal deep convolutional neural network for pedestrian segmentation. *Journal of Imaging*, 4(9), 107.

Yeong, D. J., Velasco-Hernandez, G., Barry, J. ve Walsh, J., 2021, Sensor and Sensor Fusion Technology in Autonomous Vehicles : *A Review*, *Sensors*, 21 (6), 2140.

Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., & Darrell, T. (2020). BDD100K: A diverse driving dataset for heterogeneous multitask learning. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 2633–2642). Demo video. Retrieved from https://drive.google.com/drive/u/0/folders/1tuLt6vbzucRCh3aTCMnxAy7l8kgqeHgS.

Zhang, C., & Kim, J. (2019). Modeling long- and short-term temporal context for video object detection. *In 2019 IEEE International Conference on Image Processing (ICIP)* (pp. 71–75). IEEE.

Zhang, Y., Sun, P., Jiang, Y., Yu, D., Yuan, Z., Luo, P., Liu, W., & Wang, X. (2021). *ByteTrack: Multi-object tracking by associating every detection box. In European Conference on Computer Vision*. Retrieved from https://api.semanticscholar.org/CorpusID:238744032.

ÖZGEÇMİŞ

ORCID NO: 0009-0005-6985-7824

Ad Soyad : Özlem Okur

Yabancı Dil : İngilizce

Eğitim ve Mesleki Geçmişi:

- Yüksek Lisans 2023-2024, Eskişehir Teknik Üniversitesi, Lisansüstü Eğitim Enstitüsü, Bilgisayar Mühendisliği, Bilgisayar Mühendisliği Anabilim Dalı
- Lisans 2011-2012, Varşova Teknoloji Üniversitesi, Enformatik ve Bilgisayar Mühendisliği Bölümü
- Lisans 2009-2014, Anadolu Üniversitesi, Bilgisayar Mühendisliği Bölümü
- 2022-, Yazılım ve Entegrasyon Yöneticisi, Gürok Grup
- 2019-2022, Kıdemli Yazılım Uzmanı, Gürok Grup
- 2015-2019, Yazılım Uzmanı, LAV
- 2014 (Temmuz-Ekim), Proje Stajyeri, Savronik
- 2013 (Temmuz-Ekim), Proje Stajyeri, Bitsnbrains S.L.