



T.C.
KONYA TEKNİK ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ



OTONOM SÜRÜŞ İÇİN DERİN ÖĞRENME
TABANLI YENİLİKÇİ ÇÖZÜM
YAKLAŞIMLARI

Salim AZAK

DOKTORA TEZİ

Elektrik-Elektronik Mühendisliği Anabilim Dalı

Ocak-2024
KONYA
Her Hakkı Saklıdır

TEZ KABUL VE ONAYI

Salim AZAK tarafından hazırlanan “Otonom Sürüş için Derin Öğrenme Tabanlı Yenilikçi Çözüm Yaklaşımları” adlı tez çalışması 31/01/2024 tarihinde aşağıdaki jüri tarafından oy birliği / oy çokluğu ile Konya Teknik Üniversitesi Lisansüstü Eğitim Enstitüsü Elektrik-Elektronik Mühendisliği Anabilim Dalı’nda DOKTORA TEZİ olarak kabul edilmiştir.

Jüri Üyeleri

İmza

Başkan

Doç. Dr. İlker Ali ÖZKAN
(Selçuk Üniversitesi)

.....

Danışman

Doç. Dr. Akif DURDU
(Konya Teknik Üniversitesi)

.....

Üye

Prof. Dr. Seral ÖZŞEN
(Konya Teknik Üniversitesi)

.....

Üye

Doç. Dr. S. Sinan GÜLTEKİN
(Konya Teknik Üniversitesi)

.....

Üye

Dr. Öğr. Üyesi Hakan TERZİOĞLU
(Selçuk Üniversitesi)

.....

Yukarıdaki sonucu onaylarım.

Prof. Dr. Mevlüt UYAN
Enstitü Müdürü

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Salim AZAK

04.01.2024

ÖZET

DOKTORA TEZİ

OTONOM SÜRÜŞ İÇİN DERİN ÖĞRENME TABANLI YENİLİKÇİ ÇÖZÜM YAKLAŞIMLARI

Salim AZAK

**Konya Teknik Üniversitesi
Lisansüstü Eğitim Enstitüsü
Elektrik-Elektronik Mühendisliği Anabilim Dalı**

Danışman: Doç. Dr. Akif DURDU

2024, 115 Sayfa

Jüri

Doç. Dr. Akif DURDU

Prof. Dr. Seral ÖZŞEN

Doç. Dr. S. Sinan GÜLTEKİN

Doç. Dr. İlker Ali ÖZKAN

Dr. Öğr. Üyesi Hakan TERZİOĞLU

Derin öğrenme alanındaki ilerlemeler, son yıllarda otonom sürüş teknolojilerinde önemli gelişmelere yol açmıştır. Bu gelişmeler, çeşitli endüstriyel ve akademik alanlarda büyük heyecan uyandırmış ve otonom araç teknolojilerinin gelişimine ivme kazandırmıştır. Derin öğrenme yöntemlerinin kullanımı, otonom araçların çevresel algılama, karar alma ve yol takibi gibi temel yeteneklerini geliştirmede kritik bir rol oynamıştır. Günümüzde geliştirilen birçok otonom araç, genellikle LiDAR, radar ve gelişmiş kameralar gibi maliyetli sensörlerle donatılmış zengin bir sensör altyapısı kullanmaktadır. Ancak, insan sürücüler, pahalı sensörlere ihtiyaç duymadan yalnızca görsel algılarıyla gündüz, gece ve hatta zorlu hava koşullarında etkileyici bir sürüş performansı sergileyebilmektedirler. Hem otonom araçlar hem de insan sürücüler, çevresel algılamalarına dayanarak hareket kontrol komutları üretirler. Bu komutlardan direksiyon açısı ve hızın tahmini, otonom sürüşün yerine getirilmesi gereken temel görevlerindendir. Bu tez çalışması, direksiyon açısını ve araç hızını tahmin etmek için monoküler kamera görüntüsünden elde edilen çeşitli anlamlı ipuçları kullanan uçtan uca bir yöntem sunmaktadır. Önerilen model çerçevesinde üç giriş görüntüsü kullanılmaktadır. Sahne dokusu ve görünüm detaylarını aktaran renkli görüntünün yanı sıra, uzamsal ve anlamsal çevre yapıları hakkında bilgi veren monoküler derinlik görüntüsü ve anlamsal segmentasyon görüntüsü de dahil edilmiştir. Ayrıca, zamansal özellikleri elde etmek için de LSTM birimleri kullanılmıştır. Önerilen çok girdili çok görevli model, gerçek dünya şartlarında insan sürücüler tarafından oluşturulmuş Udacity ve Sully Chen veri kümelerinde ve CARLA Sürüş Simülatörü üzerinden toplanan veri kümesinde değerlendirilmiş ve mevcut yöntemlerle karşılaştırılmıştır. Bu karşılaştırmalar sonucunda, önerilen modelin, Udacity veri kümesinde direksiyon açısı için %44,96 ve hız için %4,39 iyileştirme sağlayarak RMSE değerlerinde literatürdeki en başarılı sonuçlara ulaştığı gözlemlenmiştir. Benzer şekilde Sully Chen ve CARLA veri kümeleri üzerinde elde edilen sonuçlar da literatürdeki en başarılı sonuçlardır. Modelin her bir bileşenin etkinliğini değerlendirmek ve modele katkısını incelemek amacıyla kapsamlı bir eksiltme çalışması da gerçekleştirilmiştir. Elde edilen sonuçlar, yalnızca görsel girdi kullanan otonom sürüş sistemlerinin potansiyelini vurgulamaktadır. Araştırmanın sonunda, farklı amaçlar doğrultusunda otonom sürüş alanında çalışmak isteyen araştırmacılar için kolayca erişilebilecek, çeşitli amaçlara hitap eden kapsamlı bir veri kümesi oluşturulmuştur. Elde edilen veri kümesi, sensör çeşitliliği, toplanan veri miktarı ve veri çeşitliliği açısından mevcut literatürdeki en kapsamlı CARLA veri kümesi olmuştur. Bu veri kümesi, kullanıma hazır bir şekilde sunulmuştur, böylece araştırmacılar verilere ulaştıktan sonra hemen kullanabilirler. Söz konusu yaklaşım, araştırmacıların veri kümesi oluşturmak için

zaman ve efor harcamadan hızlı bir şekilde CARLA veri kümesi üzerinde çalışmalarını sürdürmelerine olanak tanır.

Anahtar Kelimeler: Derin öğrenme, direksiyon açısı tahmini, evrimsel sinir ağıları, hız tahmini, otonom sürüş, veri kümesi



ABSTRACT

PhD THESIS

INNOVATIVE SOLUTION APPROACHES BASED ON DEEP LEARNING FOR AUTONOMOUS DRIVING

Salim AZAK

**Konya Technical University
Institute of Graduate Studies
Department of Electrical and Electronics Engineering**

Advisor: Assoc. Prof. Dr. Akif DURDU

2024, 115 Pages

Jury

**Assoc. Prof. Dr. Akif DURDU
Prof. Dr. Seral ÖZŞEN
Assoc. Prof. Dr. S. Sinan GÜLTEKİN
Assoc. Prof. Dr. İlker Ali ÖZKAN
Asst. Prof. Dr. Hakan TERZİOĞLU**

Advances in deep learning have led to significant developments in autonomous driving technologies in recent years. These developments have aroused great excitement in various industrial and academic fields and accelerated the development of autonomous vehicle technologies. The use of deep learning methods has played a critical role in improving the basic capabilities of autonomous vehicles such as environmental sensing, decision making and lane following. Many autonomous vehicles developed today utilise a rich sensor infrastructure, often equipped with costly sensors such as LiDAR, radar and advanced cameras. However, without the need for expensive sensors, human drivers can achieve impressive driving performance in daytime, nighttime and even in extreme weather conditions using only their visual perception. Both autonomous vehicles and human drivers generate motion control commands based on their environmental perceptions. Among these commands, steering angle and speed estimation are the main tasks of autonomous driving. This dissertation presents an end-to-end method for estimating steering angle and vehicle speed using various meaningful cues extracted from a monocular camera image. Three input images are used in the proposed model. In addition to the colour image, which conveys scene texture and appearance details, the monocular depth image and the semantic segmentation image, which provide information about spatial and semantic structures of the environment, respectively, are also included. In addition, LSTM units are used to obtain temporal features. The proposed multi-modal multi-task model is evaluated and compared with existing methods on the Udacity and Sully Chen datasets generated by human drivers in real-world conditions and on the dataset collected through the CARLA Driving Simulator. As a result of these comparisons, it is observed that the proposed model achieves the state-of-the-art results in the literature in RMSE values by providing 44,96% improvement for steering angle and 4,39% improvement for speed in the Udacity dataset. Similarly, the results obtained on the Sully Chen and CARLA datasets are the most successful results in the literature. A comprehensive ablation study was also performed to evaluate the effectiveness of each component of the model and to examine its contribution to the model. The results obtained emphasise the potential of autonomous driving systems using only visual input. At the end of the research, a comprehensive dataset has been created for researchers aiming to work in the field of autonomous driving for various purposes. The obtained dataset has become the most comprehensive CARLA dataset in the current literature in terms of sensor diversity, collected data quantity, and data variety. This dataset has been made available ready-to-use, allowing researchers to utilize it immediately

upon accessing the data. This approach enables researchers to work quickly on the CARLA dataset without spending significant time and effort to create a dataset.

Keywords: Autonomous driving, convolutional neural networks, dataset, deep learning, speed estimation, steering angle estimation



ÖNSÖZ

Bu doktora çalışması, Doç. Dr. Akif DURDU danışmanlığında gerçekleşmiştir. Kendisinin bu çalışma boyunca gösterdiği rehberlik, eleştiri, tavsiyelerine ve bana her zaman sağladığı desteğe minnettarım. Kendisiyle çalışmak benim için büyük bir şanstı.

Prof. Dr. Seral ÖZŞEN, Doç. Dr. İlker Ali ÖZKAN ve Doç. Dr. S. Sinan GÜLTEKİN'e bu araştırma sırasında verdikleri destek ve tavsiyeler için teşekkür etmek istiyorum.

ODTÜ İnşaat Mühendisliği Bölümü'nün önceki başkanlarından Prof. Dr. Güney ÖZCEBE'ye, mevcut başkanı Prof. Dr. Erdem CANBAY'a ve Prof. Dr. Barış BİNİCİ'ye; bu çalışmanın tamamlanması için sundukları imkanlardan dolayı derin teşekkürlerimi sunmak isterim.

Çalışmalarım sırasında bana hem bir iş arkadaşı hem de bir dost olarak daima destek olan Fırat BOZKAYA'ya teşekkür ederim.

Ayrıca Bekir Özer AY, Ramazan ÖZÇELİK, Turan ERDOĞAN, Sinan KARADUMANLI, Ümit ÖZPOLAT, Okay ÖZCAN, Oktay ÖZCAN, Erhan SEVİM ve Veli TAŞÇI'ya dostlukları ve her ihtiyacım olduğunda verdikleri destek için teşekkür borçluyum.

Bu süreçte beni daima cesaretlendiren ve fikir alışverişlerimde bana ilham veren kıymetli hocam Prof. Dr. S. Tanvir WASTİ'ye teşekkürü bir borç bilirim.

Bana daima güç ve enerji veren, hayatımı güzelleştiren sevgili oğlum Emre'ye, Değerli eşim Tuba EROĞLU AZAK'a, sevgisi, sabrı, anlayışı ve çalışmanın en zor zamanlarında verdiği desteği için teşekkürlerimi sunarım.

Son olarak bugünlere gelmemde büyük desteği olan, annem ve babam Huriye AZAK ve Yakup AZAK ile sevgili kardeşim Nigar BIYIKLI'ya, hayatım boyunca gösterdikleri sonsuz sevgi, anlayış ve destek için şükranlarımı sunarım.

Salim AZAK
KONYA-2024

İÇİNDEKİLER

ÖZET	iv
ABSTRACT.....	vi
ÖNSÖZ	viii
İÇİNDEKİLER.....	ix
ŞEKİLLER LİSTESİ	xi
ÇİZELGELER LİSTESİ	xiii
SİMGELER VE KISALTMALAR.....	xiv
1. GİRİŞ	1
1.1 Genel Bakış	1
1.2 Tezin Amacı ve Kapsamı	4
1.3 Tezin Organizasyonu	6
2. LİTERATÜR TARAMASI.....	8
2.1 Görme Tabanlı Sürüşte Model Mimarileri.....	9
2.2 Otonom Sürüşte Amaçlanan Görevler	12
2.3 Eğitim ve Test Ortamları.....	12
3. RENK DOKUSU, UZAYSAL VE SEMANTİK SEGMENTASYON BİLİNÇLİ SÜRÜŞ MODELİ	14
3.1 Model Mimarisine Genel Bakış	17
3.2 Uzaysal Çıkarım ve Semantik Segmentasyon İçin Ön-İşleme Teknikleri.....	19
3.2.1 Uzaysal çıkarım – monoküler derinlik tahmini	19
3.2.2 Semantik segmentasyon ile özellik çıkarımı	21
3.3 Evrimsel Özellik Çıkarımı.....	23
3.3.1 Evrimsel Sinir Ağları (ESA)	23
3.3.2 EfficientNetV2 ağ yapısı	26
3.4 Zamansal Özellik Çıkarımı	31
3.5 Uçtan Uca Komut Tahmini	33
4. VERİ KÜMELERİ VE EĞİTİM VERİLERİNİN HAZIRLANMASI	35
4.1 Udacity Veri Kümesi.....	35
4.1.1 Udacity eğitim verisinin hazırlanması	39

4.2	CARLA Veri Kümesi ve Simülasyon Platformu	44
4.2.1	CARLA otonom sürüş simülatörü	45
4.2.2	CARLA veri kümesi	48
4.3	Sully Chen Veri Kümesi	51
5.	DENEYSEL SONUÇLAR	53
5.1	Eğitim Ayrıntıları	53
5.2	Eksiltme Çalışması	58
5.3	Sonuçlar ve Analiz	61
5.3.1	Direksiyon açısı tahmini	63
5.3.2	Hız tahmini	71
5.3.3	Karşılaştırmalı değerlendirme	76
5.3.4	Çalışma süresi analizi	83
6.	RAPIDSENSE VERİ KÜMESİ	87
6.1	Sensör Düzeni	88
6.2	Veri Tanımlaması	90
6.3	Veri Yapısı	100
7.	SONUÇLAR, TARTIŞMA VE ÖNERİLER	103
7.1	Sonuçlar ve Tartışma	103
7.2	Öneriler	107
KAYNAKLAR		108

ŞEKİLLER LİSTESİ

Şekil 3.1. Önerilen çok girdili çok görevli uçtan uca modelin mimarisi. Ağda her bir görüntü için ayrı ESA ağı eğitilmiştir. EfficientNetV2 modülleri, uzaysal özellik kodlaması için kullanılırken, LSTM modülü zamansal özelliklerin elde edilmesi için kullanılmıştır.....	16
Şekil 3.2 Üst sırada Udacity veri kümesinden rastgele alınmış üç adet orijinal RGB görüntü ve alt sırada bu görüntülerin Monodepth2 ile elde edilmiş derinlik görüntüleri20	
Şekil 3.3 Üst sırada Udacity veri kümesinden rastgele alınmış üç adet orijinal RGB görüntü ve alt sırada bu görüntülerin YOLOPv2 ile elde edilmiş anlamsal bölümlenmiş görüntüleri.....	22
Şekil 3.4 EfficientNet'in model ölçeklemesi. a) temel ağ örneği; (b)-(d), ağ boyutlarından genişliğin, derinliğin, veya çözünürlüğün yalnızca birini artıran geleneksel ölçeklendirme (e) üç boyutun tamamını sabit bir oranda eşit şekilde ölçeklendiren bileşik ölçeklendirme yöntemi (Tan ve Le, 2019)	28
Şekil 3.5 EfficientNetV2'nin literatürdeki diğer çalışmalarla parametre sayısı, FLOPs ve gecikme süresine göre kıyaslanması (Tan ve Le, 2021).....	30
Şekil 3.6 LSTM biriminin iç yapısı	32
Şekil 4.1 Udacity veri kümesinden alınmış sürüşün farklı sahnelerine ait zorlu görüntüler: a) Sürüş yüzeyi aşırı aydınlanma nedeniyle çok parlak olduğu için ayırt edilememektedir. b) Yol yüzeyindeki parçalı gölgeler algılamayı zorlaştırır. c) Sürüş yüzeyi yine aydınlanma saatine bağlı olarak sıradan asfalt rengi yerine mavimsi bir renkte görülmektedir. d) Sürüş yüzeyinde hem çok parlak alanlar hem de koyu gölgeler mevcuttur	36
Şekil 4.2 Udacity veri kümesine ait yol haritası. Bu harita, veri kümesinin beş ayrı bölümünü temsil etmekte olup, her renk farklı bir video serisini belirtmektedir. Görseldeki beyaz bölge ise test verisi olarak seçilen kısmı göstermektedir.....	38
Şekil 4.3 Udacity veri kümesinin arazi üzerinde elde edilen üç boyutlu görüntüsü. Toplanan veri kümesi hem düz yolları hem de virajlı bölgeleri içermektedir ve engebeli bir arazide elde edilmiştir	38
Şekil 4.4 Direksiyon açılarının histogramı. Udacity veri kümesinin ham verilerinde, düz gidişi temsil eden küçük değerli direksiyon açısı etiketlerinin sayısı oldukça fazladır, bu da dengesiz bir dağılıma yol açmaktadır	40
Şekil 4.5 Bütün veri kümesinin (eğitim + test) direksiyon açılarının zaman serisi. Sarı renkle vurgulanan bölüm, tüm verinin %10'unu oluşturan test verisini temsil etmektedir	40
Şekil 4.6 Orijinal görüntü (a) üzerinde veri artırma yöntemlerinin uygulanmasıyla elde edilmiş rastgele gölgeli görüntü (b), düşük parlaklığa sahip görüntü (c), Gauss gürültü eklenmiş görüntü (d) ve orijinal görüntüden elde edilmiş derinlik görüntüsü (e) ile anlamsal bölümlenmiş görüntü (f).....	44
Şekil 4.7 CARLA simülatöründe farklı kasabalara ait görüntüler. Üst sıradaki görüntüler soldan sağa <i>Town01</i> , <i>Town02</i> , <i>Town03</i> ; alt sıradaki görüntüler soldan sağa <i>Town04</i> , <i>Town05</i> ve <i>Town06</i> 'dan alınmıştır	45
Şekil 4.8 Soldan sağa <i>Town04</i> , <i>Town05</i> ve <i>Town06</i> kasabalarının havadan görünüşleri	46
Şekil 4.9 CARLA sürüş simülatörü ile toplanmış veri kümesinden farklı hava (ıslak zemin, orta yağmurlu, bulutlu, açık) ve aydınlanma şartlarına (gündüz, gün batımı) ait görüntüler.....	47

Şekil 4.10 CARLA veri kümesinden alınan bir RGB görüntüye ait farklı derinlik görüntüleri.....	50
Şekil 4.11 Sully Chen veri kümesinden alınmış farklı ışıklandırma, farklı yol yapıları gibi değişik sürüş koşullarını içeren bazı görüntüler	51
Şekil 5.1 Direksiyon açısı ve hızdaki zamansal bağımlılıkları çıkartmak için kullanılan LSTM dalı.....	58
Şekil 5.2 Udacity veri kümesi üzerinde direksiyon açısı tahminlerinin sonuçları ve bunlara karşılık gelen gerçek açıları. Direksiyon açısı tahminleri mavi renkle gösterilirken, gerçek direksiyon açısı değerleri kırmızı renkle gösterilmiştir	64
Şekil 5.3 Udacity veri kümesi üzerinde tahmin edilen direksiyon açıları ve her tahmin sırasında oluşan sapma miktarları. Direksiyon açısı tahminleri mavi çizgi ile gösterilirken, sapma miktarları sarı renkli bant ile gösterilmiştir	65
Şekil 5.4 Direksiyon açısı tahminlerinin gerçek verilere göre dağılımı	66
Şekil 5.5 Frekans bazında direksiyon açısı sapma miktarının dağılımı.....	67
Şekil 5.6 CARLA veri kümesinde Monodepth2 kullanılarak elde edilmiş direksiyon açısı tahminleri	68
Şekil 5.7 Monodepth2 derinlik tahmin yöntemi yerine CARLA sürüş simülatörünün sağladığı derinlik görüntüleri kullanılarak elde edilmiş direksiyon açısı tahminleri.....	69
Şekil 5.8 Sully Chen veri kümesinde elde edilmiş direksiyon açısı tahminleri.....	70
Şekil 5.9 Udacity veri kümesi üzerinde elde edilmiş hız tahminleri ve bunlara karşılık gelen gerçek hız değerleri. Hız tahminleri yeşil renkle, gerçek hız değerleri ise kırmızı renkle gösterilmiştir	72
Şekil 5.10 Tahmin edilen araç hızı sonuçları ve her tahmin sırasında oluşan sapma miktarları. Araç hızı tahminleri mavi renkle gösterilirken, sapma miktarları yeşil renkle gösterilmiştir	73
Şekil 5.11 Hız tahminlerinin gerçek verilere göre dağılımı.....	74
Şekil 5.12 Frekans bazında araç hızı sapma miktarının dağılımı	74
Şekil 5.13 CARLA veri kümesi üzerinde Monodepth2 monoküler derinlik tahmin metodu kullanılarak elde edilmiş hız tahminleri ve bunlara karşılık gelen gerçek hız değerleri. Tahminler yeşil renkle gösterilirken, gerçek hız değerleri kırmızı renkle gösterilmiştir	75
Şekil 5.14 CARLA veri kümesi üzerinde Monodepth2 metodunun sağladığı görüntüler yerine CARLA'nın kendi sağladığı derinlik görüntüleri ile elde edilmiş hız tahminleri ve bunlara karşılık gelen gerçek hız değerleri. Tahminler yeşil renkle gösterilirken, gerçek hız değerleri kırmızı renkle gösterilmiştir	76
Şekil 5.15 Önerilen modelin eşleşen zaman damgalarını doğrulayarak alt ağ verilerini senkronize eden ROS çerçevesi.....	85
Şekil 6.1 Aracın dört yönünü gösteren RGB kameralar. Üst sıra, soldan sağa, ön ve sağ kamera; alt sıra soldan sağa, arka ve sol kamera	88
Şekil 6.2 Sol üst köşede DVS kamera, sağ üst köşede LiDAR görüntüsü, altta ise kamera görüntüsü üzerine yansıtılmış radar verisi görülmektedir	90
Şekil 6.3 Town02 kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)	91
Şekil 6.4 Town03 kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)	92
Şekil 6.5 Town04 kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)	93
Şekil 6.6 Town05 kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)	94
Şekil 6.7 Town06 kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)	95
Şekil 6.8 Town07 kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)	96
Şekil 6.9 Town10 kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)	97
Şekil 6.10 Town12 kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)	98
Şekil 6.11 Veri kümesinin dosya düzeni ve veri yapısı.....	99

ÇİZELGELER LİSTESİ

Tablo 3.1 EfficientNetV2-S'in mimarisi	27
Tablo 3.2 EfficientNetV2'nin ImageNet üzerinde performans karşılaştırması (Tan ve Le, 2021)	29
Tablo 4.1 Udacity sürüş datası ve açıklamaları	37
Tablo 4.2 CARLA sürüş simülatörü ile toplanan verinin ayrıntıları	49
Tablo 5.1 Eksiltme çalışması: Modellerin ve ayar değişkenlerinin model performansına katkıları	59
Tablo 5.2 Udacity veri kümesi üzerinde elde edilmiş niceliksel sonuçlar.....	77
Tablo 5.3 CARLA veri kümesi üzerinde elde edilmiş niceliksel sonuçlar.....	81
Tablo 5.4 Sully Chen veri kümesi üzerinde elde edilmiş niceliksel sonuçlar	83
Tablo 6.1 Kamera veri yapısı.....	89
Tablo 6.2 Radar veri yapısı.....	89
Tablo 6.3 LiDAR veri yapısı	89
Tablo 6.4 DVS Kamera veri yapısı.....	89
Tablo 6.5 GNSS veri yapısı	89
Tablo 6.6 IMU veri yapısı.....	90
Tablo 6.7 Farklı aydınlanma durumu ve farklı hava şartlarına göre kasabalardan toplanan veri miktarı.....	101
Tablo 6.8 CARLA ile oluşturulmuş veri kümelerine ait özet bilgiler	102

SİMGELER VE KISALTMALAR

Simgeler

b	: Sapma
β_1	: Nadam ilk moment tahminleri üstel azalma oranı
β_2	: Nadam ikinci moment tahminleri üstel azalma oranı
C	: Görüntü kanal sayısı
$\mathbf{c}_i$	: Direksiyon açısı ve hızdan oluşan kontrol çifti
$\mathbf{c}_t$	: LSTM hücre durumu
$\tilde{\mathbf{c}}_t$	: LSTM hücre aday durumu
$\mathcal{D}$	: Algı- kontrol çifti
D_t	: Yoğun derinlik haritası
e	: Matematiksel sabit
ϵ	: Nadam sayısal kararlılık sabiti
f_i	: LSTM unutma kapısı
$\mathcal{F}$	: Ana model fonksiyonu
$\mathcal{G}^{ht \times wd \times C}$	: $\mathcal{G} = \{0, 1, \dots, 255\}$ değer aralığında $ht \times wd \times C$ boyutlarında gri değerler
$\mathcal{H}$	: Uzaysal özellik fonksiyonu
$h^{(l)}$	: ESA v LSTM ağlarından elde edilen üst düzey vektör
ht	: Görüntü yüksekliği
$\mathcal{I}_j$	: Görüntüdeki piksel
$\mathcal{I}^d$	: Derinlik görüntüsü
$\mathcal{I}^{rgb}$	: RGB görüntü
$\mathcal{I}_{j,z}^{seg}$	: Bir $\mathcal{I}_j$ pikselinin $\mathcal{Z}$ kümesinde belirli bir anlam sınıfı z 'ye ait olma olasılığı
$\mathcal{I}^{seg}$	: Semantik bölümlenmiş görüntü
$\mathbb{I}^{ht \times wd \times \mathcal{Z} }$	: $\mathbb{I} = [0,1]$ aralığında $ht \times wd \times \mathcal{Z} $ boyutlarında olasılık değerleri
I_t	: Hedef görüntü
$I_{t'}$	: Kaynak görüntü
i	: Tam sayı indeksi
i_t	: LSTM giriş kapısı
k	: Zamansal özellik çıkarımı için dikkate alınan son görüntü sayısı
K	: Önceden hesaplanmış esaslar
$\mathcal{L}$	: Kayıp fonksiyonu
$\mathcal{L}_{direksiyon}$	: Direksiyon açısı kayıp fonksiyonu
$\mathcal{L}_{hız}$	: Araç hızı kayıp fonksiyonu
$\mathcal{L}_{toplam}$	: Toplam kayıp fonksiyonu
L_p	: Fotometrik yeniden yansıtma hatası
λ_{dr}	: Direksiyon kaybı ağırlığı
$\lambda_{hız}$	: Araç hızı kayıp ağırlığı
m_i	: Sınıf indeksi
n	: Tam sayı indeksi
o_t	: LSTM çıkış kapısı
pe	: Fotometrik yeniden yapılandırma hatası

$\mathbf{p}_i$	: RGB görüntülerden oluşan algı
R^2	: Doğrusal regresyon denklemi ölçütü
s_i	: i indeksli direksiyon açısı gerçek değeri
$\tilde{s}_i$	: i indeksli direksiyon açısı tahmini
s_t	: t anındaki gerçek direksiyon açısı
$\tilde{s}_t$	: t anındaki tahmin edilen direksiyon açısı
S_p	: Kaybın hız bileşeni
S_t	: Kaybın direksiyon açısı bileşeni
σ^2	: Varyans
$\sigma(x)$	: Sigmoid fonksiyonu
$\sum$	: Toplam sembolü
t	: Zaman değişkeni
$T_{t' \rightarrow t}$	: Göreceli poz
$\mathcal{U}$	: Zamansal özellik fonksiyonu
v_i	: i indeksli gerçek hız
$\tilde{v}_i$	: i indeksli tahmin edilen hız
v_t	: t anındaki hız
$\tilde{v}_t$	: t anındaki tahmin edilen hız
W	: Ağırlık matrisi
$w^{(i)}$	: Özellik vektörleri için ağırlık katsayısı
wd	: Görüntü genişliği
$\mathbf{x}_t$	: LSTM giriş vektörü
z	: Birleştirilmiş ağırlıklı üst düzey özellikler
$\mathcal{Z}$	: $\mathcal{Z} = \{1, 2, \dots, Z \}$ anlamsal bölütleme sınıf kümesi
$\circ$	: Hadamard çarpımı
$*$	: Bulunduğu yerde özel biri durum olduğunu ifade eder. Açıklamalara bakılmalıdır.
$\dagger$	: Sonuçların başka çalışmadan alındığını gösterir. Tablo açıklamasına bakılmalıdır.
$\ddagger$	: Sonuçların başka çalışmadan alındığını gösterir. Tablo açıklamasına bakılmalıdır.
$\S$	: Sonuçların başka çalışmadan alındığını gösterir. Tablo açıklamasına bakılmalıdır.

Kısaltmalar

ABS	: Anti-lock Braking System (Kilitlenme karşıtı fren sistemi)
API	: Application Programming Interface (Uygulama Programlama Arayüzü)
CAN bus	: Controller Area Network (Kontrol Alanı Ağı)
CSV	: Comma-separated values file (Virgülle ayrılmış değerler dosyası)
CPU	: Central Processing Unit (Merkezi işlem birimi)
DARPA	: (Defense Advanced Research Projects Agency)
ESA	: Evrimsel Sinir Ağları (Convolutional Neural Networks)
FCN	: Fully Convolutional Networks (Tamamen Evrimsimli Ağlar)

FLOPs	: Floating Point Operations Per Second (saniye başına yapılan kayan nokta işlemlerinin sayısı)
FPS	: Frames per second (Saniyede Kare Sayısı)
GPS	: Global Positioning System (Küresel Konumlandırma Sistemi)
GPU	: Graphics processing unit (Grafik işlem birimi)
GRU	: Gated Recurrent Unit (Geçitli Tekrarlayan Birim)
GZIP	: Dosya sıkıştırma ve açma için kullanılan bir dosya formatı ve yazılım uygulamasıdır
H5	: Hierarchical Data Format version 5 (Hiyerarşik Veri Formatı sürüm 5)
HDF5	: Hierarchical Data Format version 5 (Hiyerarşik Veri Formatı sürüm 5)
IMU	: Inertial Measurement Unit (Ataletsel Ölçüm Birimi)
JPG	: Joint Photographic Experts Group (Birleşmiş Fotoğraf Uzmanları Grubu)
LiDAR	: Light detection and Ranging (Işıkla Algılama ve Uzaklık Ölçümü)
LSTM	: Long Short-Term Memory
MAE	: Mean Absolute Error (Ortalama Mutlak Hata)
MBConv	: Mobile Inverted Bottleneck (Mobil Ters çevrilmiş Darboğaz)
MSE	: Mean Squared Error (Ortalama Kare Hata)
NAG	: Nesterov Accelerated Gradient (Nesterov'un hızlandırılmış gradyanı)
NAS	: Neural Architecture Search (Nöral Mimari Arama)
Navlab	: Navigasyon laboratuvarı
ONNX	: Open Neural Network Exchange (Açık Sinir Ağı Değişimi)
ÖKU	: Özellik Kümesi Uzunlukları
PLY	: Polygon File Format (Polygon Dosya Formatı)
RGB	: Red, Green and Blue (Kırmızı, Yeşil ve Mavi)
RMSE	: Root Mean Squared Error (Kök Ortalama Kare Hata)
RNN	: Recurrent Neural Network (Tekrarlayan sinir ağı)
ROS	: Robot Operating System (Robot İşletim Sistemi)
RTX	: Ray Tracing Technology (Işın İzleme Teknolojisi)
SGD	: Stochastic Gradient Descent (Stokastik Gradyan İnişi)
TPU	: Tensor Processing Unit (Tensor İşlem Biri)
TS2DNet	: Texture, Spatial, and Semantic Aware Driving Network (RGB dokusu, uzayal ve anlamsal farkındalığı olan sürüş ağı)
YOLOP	: You Only Look Once for Panoptic Driving Perception (Panoptik Sürüş Algısı için Sadece Bir Kez Bakarsınız)
YOLOPv2	: Better, Faster, Stronger for Panoptic driving Perception (Panoptik Sürüş Algısı için Daha İyi, Daha Hızlı, Daha Güçlü)

1. GİRİŞ

Bu bölümde, otonom araçların zaman içinde gelişimi ve uçtan uca kontrol konularına genel bir bakış sunulacak, ardından tezin amacı ve kapsamı hakkında bilgi verilecek ve son olarak tezin organizasyonu açıklanacaktır.

1.1 Genel Bakış

Otonom sistemlerin yolunu açan elektromekanik sistemlerin ortaya çıkması ve devrim niteliğinde bir buluş olan otomobillerin geliştirilmeye başlanması ile birlikte kendi kendine sürüş yapabilen (otonom) otomobiller fikri hem araştırmacıların hem de endüstri dünyasının daima ilgisini çeken bir konu olmuştur. Sürücü desteği olmadan her türlü şartlarda ve her konumda hareket edebilen tam otonom araçlar henüz geliştirilmemiş olsa da gelinen noktada teknoloji buna çok yaklaşmış durumdadır. Bugün dünyada otomotiv sektöründe var olan büyük markaların birçoğu otonom araç geliştirmek için Ar-Ge birimleri kurmuştur. Ayrıca mevcut otomobiller üzerinde yazılım ve donanım eklemeleri yaparak araca otonom kabiliyet kazandırmak üzere kurulmuş şirketler de bu alanda araştırma ve geliştirme faaliyetleri yürütmektedir. Bunların dışında bir de endüstri dünyasına ışık tutan ve gelişmelerde öncülük eden akademik birimler vardır. Dünyada birçok üniversitede veya araştırma enstitülerinde ya doğrudan otonom sürüş laboratuvarları ya da otonom sürüşle ilgili çalışmalar yürüten bilgisayarlı görü, kontrol ve algılama laboratuvarları kurulmuştur. Elde edilmeye başlanan sonuçlarla birlikte, özellikle son on yılda bu alana yapılan yatırımlar çok hızlı bir şekilde artmıştır. Sonuç olarak bugün dünya üzerinde pek çok kişi ve kurum yakın zamanda yollara çıkacak tam otonom bir araç üretmek için doğrudan veya dolaylı olarak çalışmaktadır.

Otonom araçların ortaya çıkmasındaki talep, öncelikli olarak sürüş güvenliğinin artırılması ve insan kaynaklı meydana gelen kazaların minimuma indirilmesi amacıyla doğmuştur. Dünya Sağlık Örgütü'nün 2018'de yayınladığı son rapora göre dünya genelinde trafik kazalarından dolayı yılda yaklaşık 1,35 milyon kişinin hayatını kaybettiği tahmin edilmektedir (Anonymous, 2018). Bu da yaklaşık her 23 saniyede bir insanın ölmesi anlamına geliyor. Hayatını kaybeden insanların büyük bir çoğunluğu 5-29 yaş arasında yer alırken, 20 ila 50 milyon insanın ölümcül olmayan yaralanmalara maruz kaldığı ve birçoğunun da yaralanma sonucu engelli kaldığı rapor edilmektedir. Meydana gelen trafik kazaları, can kayıpları ve yaralanmaların yanı sıra ciddi ekonomik kayıplara

da yol açmaktadır. Bu tür kazalar, çoğu ülkenin gayri safi milli hasılasının %3' üne denk gelen bir maliyet oluşturmaktadır. Bu kazaların ardından yaşanan yaralanmalar, birçok insanın iş gücü kaybına yol açmakta ve üretime katkıları azalmaktadır.

Otonom araçların gelişimi, 20. yüzyılın ikinci yarısında araç güvenliği ve sürüş konforu için geliştirilen iki önemli özellik olan kilitlenmeyi engelleyen fren sistemi (ABS) ve hız kontrolü (cruise control) gibi özelliklerle başladı. ABS, frenleme sırasında tekerleklerin kilitlenmesini önleyerek aracın kaymasını engeller ve yol yüzeyiyle çekiş temasını korur. Böylece sürücünün frenleme esnasında da aracı kontrol etmesine olanak tanır. Bu sistemler günümüzde daha da gelişti; frenleme esnasında tekerleklerin kilitlenmesini önlemenin yanında elektronik fren gücü dağılımı, çekiş kontrol sistemi, elektronik kararlılık kontrolü ve virajda denge kontrolü gibi özellikler eklenerek sürüş güvenliği artırıldı. İkinci önemli buluş olan hız kontrolü ise 1948 yılında kör bir makine mühendisi olan Ralph R. Teetor tarafından geliştirildi ve patentlendi (Teetor, 1948). Sürücü tarafından kontrol edilen bir mekanizma ile devreye alınan ve aktif edildiğindeki araç hızını koruyan hız kontrol sistemi frene basıldığında devre dışı kalmakta, daha fazla hızlanmak istendiğinde ise buna izin vermektedir.

Bu iki temel özellikten sonra 2000'li yıllarda kullanılmaya başlanan özellikleri akıllı özellikler olarak sınıflandırabiliriz. Bunlardan bazıları öndeki araca göre aracın hızını ayarlayan uyarlanabilir hız kontrol sistemi, acil frenleme sistemi, trafik işareti tespit sistemi ve şerit takip sistemidir. Bu sistemler genellikle kamera ve radar gibi sensörlerle çalışırlar ve sinyal işleme, karar verme ve kontrol gibi akıllı yapıları bünyesinde barındırırlar. Bugün pek çok otomobil bu özellikleri standart donanım olarak bulundurmaktadır. Hatta günümüzde otonom olmayan birçok araçta bile kamera ve benzeri sensörler mevcuttur. Özellikle şerit takip sistemi ve otomatik park etme sistemlerinin ortaya çıkması otonom araçlar konusunda beklentiye daha da arttırmıştır.

Otonom araç konusundaki gelişmeleri mobil robotlarda yaşanan gelişmelerden ayrı tutmak doğru olmayacaktır. Mobil robotlarda 1960'lardan itibaren kaydedilen ilerlemeler otonom araç çalışmalarının da temellerini oluşturmuştur. Bir mobil robotun sensörler yardımıyla çevreyi algılaması, buradan anlamlı bilgiler çıkartarak çarpışma sakınımı yapması, boş bir güzergahta yol planlaması yaparak ilerleyebilmesi, harita oluşturabilmesi gibi özellikler, bu özelliklerin otomobillere de uyarlanıp uyarlanamayacağı sorusunu hep akla getirmiştir. Mobil robotların otomobillere kıyasla daha yavaş hareket ettiği, otomobillerin daha yüksek süratlerde ve halka açık bir trafik içinde insan taşıyarak seyrettiği göz önünde bulundurulduğunda, bilgisayar

kapasitelerinin henüz yeterli seviyeye ulaşmaması nedeniyle mobil robotlardaki gelişmeler, bu yıllarda otonom araçlara aktarılamamıştır.

Gerçek anlamda otonom hareket eden ilk araçlar 1970'lerin sonuna doğru ortaya çıkmaya başlamıştır. Tsukaba Üniversitesi Makine Mühendisliği Laboratuvarı 1977 yılında dünyada kendi kendine gidebilen ilk yolcu aracını Japonya yollarında test etmeyi başarmıştır. Araç bir stereo kamera yardımıyla nesne algılaması yapabilmekte ve yol çizgilerine göre direksiyon açısı kontrolü yaparak 30 km/h hızla gidebilmekteydi. DARPA (Defense Advanced Research Projects Agency) tarafından desteklenen ve Carnegie Mellon Üniversitesi Navigasyon Laboratuvarı (Navlab) tarafından geliştirilen Navlab 1 isimli otonom araç ilk sürücüsüz deneyimini 1986 yılında gerçekleştirmiştir (Dowling et al., 1987). İlerleyen yıllarda proje devam etmiş ve aracın üst versiyonları da geliştirilmiştir. DARPA'nın bu konudaki katkısı ilerleyen yıllarda da devam etmiş, önemli çalışmaların ortaya çıkmasına katkı sağlamıştır. DARPA'nın düzenlediği ve ilki 2004 yılında gerçekleştirilen yarışmada (DARPA Grand Challenge) hiçbir araç tam otonom olarak bitiş noktasına varamamış, ancak 2005 yılında düzenlenen yarışta 5 araç 212 km'lik parkuru tam otonom sürüşle tamamlamıştır. Burada ortaya çıkan başarı otonom araçlar için bir dönüm noktası olmuştur. Stanford Üniversitesi takımı tarafında geliştirilen ve bu yarışmada birinci olan araç (Stanley), LiDAR (Light detection and Ranging- Işıklı Algılama ve Uzaklık Ölçümü), GPS (Global Positioning System- Küresel Konumlandırma Sistemi), IMU (Inertial Measurement Unit- Ataletsel Ölçüm Birimi) ve diğer sensörlerle donatılmıştı. Bu sensörler, çevresel verilerin sürekli olarak algılanmasını ve değerlendirilmesini sağlayarak aracın konumunu belirleme, çevresel engelleri algılama ve manevra yapma yeteneği sağlıyordu. İkinci yarışma ve neticeleri bu alanda büyük heyecan yarattı. Derin öğrenmenin henüz sistemlere dahil olmadığı bu yıllarda, araçlar üzerindeki algılama sistemi ile son komut üretici arasında çok sayıda alt işlevsel grup çalışıyordu. Bu sistemlerin haberleşmesi, birbirlerinin çıktılarını alıp ürettikleri sonuçları başka alt birimlere iletmesi otonom sürüşü karmaşık bir mimari haline getiriyordu.

Otonom sürüşte uçtan uca kontrolün ortaya çıkışıyla birlikte (LeCun et al., 2005), çevresel algılama üzerinden doğrudan komut tahmini yapan çalışmaların sayısı önemli ölçüde artmıştır. Bu sistemler, ara fonksiyonlara gerek duymadan giriş verileriyle doğrudan kontrol komutları üretebilmektedirler. NVIDIA'dan bir araştırma grubunun 2016'da gerçek bir araç üzerinde uçtan uca mimari ile direksiyon açısı kontrolü yaptığı çalışması (Bojarski et al., 2016), derin öğrenmenin önemli derecede ilerleme kaydetmesi

ve bilgisayar kapasitelerinin artması gibi etkenlerle birlikte otonom sürüş alanındaki uçtan uca çalışmaların 2020’li yıllara doğru önemli ölçüde artmasına yol açmıştır.

Günümüzde, otonom sürüşle ilgili olarak nesne algılama, sahne anlama, sürülebilir alan tahmini, komut tahmini, planlama, hareket tahmini gibi çeşitli alanlarda yoğun bir araştırma faaliyeti sürdürülmektedir. Bu araştırmaların büyük bir kısmı, derin öğrenme tabanlı uçtan uca mimariler üzerinde yoğunlaşmaktadır. Çalışmaların sayısındaki ve otonom araç araştırmalarına ayrılan bütçelerdeki artışlara rağmen, günümüzde tamamen otonom araçlar henüz yaygın olarak kullanılmamaktadır. En gelişmiş otonom araçlar, genellikle sınırlı harita alanlarında veya diğer kısıtlı koşullarda operasyonel olabilmektedir. Bununla birlikte, dünyanın pek çok yerinde trafikte test edilmeye başlanmış araçlar artık yollarda görülmektedir. Bu noktada, otonom sürüş alanında birçok sorunun hala çözülmeye muhtaç olduğu ve tamamen otonom araçların günlük trafikte geniş ölçekte kullanılabilmesi için önemli adımların atılması gerektiği açıktır. Mevcut teknolojik gelişmelerle birlikte, bu alandaki çalışmalar ve araştırmalar, otonom sürüşün daha güvenli, daha erişilebilir ve topluma daha fazla fayda sağlayacak şekilde ilerlemesi için önemli bir potansiyele sahiptir.

1.2 Tezin Amacı ve Kapsamı

Bu tez, otonom sürüş alanında görsel algılamayı temel alan bir modelin kullanılmasıyla sürüş sahnesinin analiz edilmesi ve sürüş kararlarının geliştirilmesini amaçlamaktadır. Birincil odak noktası, kamera tarafından yakalanan RGB görüntü aracılığıyla sürüş sahnesini analiz ederek ve bunu türetilmiş görüntülerle, yani derinlik görüntüsü ve anlamsal olarak bölümlere ayrılmış görüntü ile entegre ederek sürüş doğruluğunu ve güvenliğini artırmaktır. Önerilen model, bu üç girdiden elde edilen görsel ipuçlarından yararlanarak bir derin öğrenme ağı aracılığıyla uçtan uca direksiyon açısı ve hız tahminleri yapmayı amaçlamaktadır.

Bu araştırmanın arkasındaki temel fikir, önceki çalışmalarda kullanılan yüksek maliyetli sensörler yerine tek bir uygun maliyetli kamera görüntüsüne dayanarak güvenli sürüş koşulları oluşturmaktır. Birden fazla sensörü entegre eden geleneksel sensör füzyon yöntemlerinin aksine, bu çalışma kamera görüntülerini kullanarak yalnızca derin öğrenmeye dayalı kontrol komutlarının tahmin edilmesine vurgu yapmaktadır.

Bu araştırmayı yönlendiren ana motivasyon, insan sürücülerin LiDAR veya radar gibi pahalı sensörlere ihtiyaç duymadan yalnızca görsel algıyı kullanarak araç

kullanabilme potansiyelidir. Özellikle, insan sürücülerin geceleri sadece araç farlarını veya yol aydınlatmasını kullanarak araçları başarılı bir şekilde kullanabilmeleri, görsel algının etkinliği için önemli bir kanıt teşkil etmektedir. Bu durum, güvenli araç kullanımının sağlanmasında görsel algının oynayabileceği rolün önemini vurgulamaktadır.

Temel mimari hedef, üç ayrı girdi oluşturarak sistemi besleyen üç farklı sensörün varlığını simüle etmektir: bir RGB görüntü, bir derinlik görüntüsü ve bir anlamsal bölümlenmiş görüntü. Bu yaklaşım, modele farklı veri türleri sunarak farklı hassasiyet seviyelerine sahip sensörlerden elde edilen bilgilerin birleştirilmesini simüle etmeyi amaçlamaktadır. Bu tasarım tercihi, çalışmanın temel motivasyonu ile uyumludur ve sisteme birden fazla perspektif sağlayarak genel performansı artırmayı ve daha sağlam sonuçlar elde etmeyi amaçlamaktadır.

Daha önce de belirtildiği gibi, insan sürücüler güvenli sürüş için görüş dışında lazer veya radar gibi teknolojik sensörlere ihtiyaç duymazlar. İnsan görsel sistemi, derinlik tahmini ve semantik bölümlendirme gibi olağanüstü yeteneklere sahiptir. İnsan beyni her iki gözden gelen bilgileri entegre ederek çevredeki dünyanın tutarlı, üç boyutlu bir algısını oluşturur. Buna ek olarak, insan görme sistemi görsel sahne içindeki anlamlı bileşenleri ayırt etmek için bir tür semantik segmentasyon gerçekleştirir. İnsan görüşünün bu temel yönleri, sürüş ortamının kapsamlı bir şekilde algılanmasını ve anlaşılmasını sağlar. Aslında, önerilen modelin yapısı bu ilkelere yakından benzemektedir. İnsan gözünün bilgiyi yakalayıp dahili olarak işlemesine benzer şekilde, model kamera görüntüsünü elde ettikten sonra temel mimarisi dahilinde derinlik tahmini ve anlamsal segmentasyon gerçekleştirir. Daha sonra zenginleştirilmiş bu bilgiler ışığında direksiyon açısı ve hız tahmini yapar.

Önerilen model öncelikle monoküler kamera ile görsel algıya dayansa da sahne tespiti için bir LiDAR sensöründen derinlik bilgisi dahil edilerek ve sensör füzyon teknikleri uygulanarak da gerçekleştirilebileceğini belirtmek gerekir. Ayrıca monoküler kamera yerine stereo görüş teknikleri ile de derinlik bilgisi elde etmek mümkündür. Yine tahmin doğruluğunu artırmak adına radar veya IMU benzeri sensörler de dahil edilebilir. Ancak bu tezin birincil odak noktası, görsel algının karar vermede tek başına etkinliğini göstermek ve geleneksel sensör tabanlı yaklaşımlara uygun maliyetli bir alternatif olarak potansiyelini sergilemektir. Dolayısıyla farklı sensör tipleri ile yapılacak çalışmalar, bu tezin kapsamı dışındadır.

Bu araştırma genel olarak, görsel algının gücünden faydalanarak doğru sürüş kararları alabilen bir model geliştirme amacını taşımaktadır. Derin öğrenme algoritmalarını kullanarak ve monoküler bir kameradan elde edilen çeşitli görsel ipuçlarını entegre ederek, pahalı sensörlere olan bağımlılığı azaltmayı hedeflemektedir. Bu yaklaşım, sürüş doğruluğunu ve güvenliğini artırarak daha erişilebilir ve verimli otonom sürüş sistemlerinin gelişimine katkı sağlamayı amaçlamaktadır.

Çalışmanın son bölümünde, otonom sürüş alanında çalışan araştırmacılar için kolay erişilebilir bir sentetik veri kümesi sunulması amaçlanmıştır. Veri kümelerinin oluşturulmasıyla ilgili zorluklar, literatürde mevcut olan sınırlı sayıdaki veri kümesi ve bu veri kümelerinin veri çeşitliliği bakımından yetersiz kalması nedeniyle geniş kapsamlı, çok sayıda sensör içeren ve topluluğun erişimine açık bir veri kümesi oluşturulması amaçlanmıştır.

1.3 Tezin Organizasyonu

Bu tez çalışması, yedi bölümden oluşmaktadır. Birinci bölüm, otonom sürüşün zaman içindeki evrimine genel bir bakış sunmaktadır. Bu bölüm ayrıca otonom sürüş mimarilerindeki değişimi ve uçtan uca görsel sürüş mimarisinin gelişimini günümüze kadar ele almaktadır.

İkinci bölüm, görsel otonom sürüşle ilgili kapsamlı bir literatür taraması sunmaktadır. Bu bölümde, derin öğrenme sonrası yapılan çalışmalar incelenmiş ve sahne bilgisinden kontrol çıktısı üreten çalışmalar farklı alt başlıklar altında özetlenmiştir. Bu alanda yapılmış çalışmalar, kullanılan modellerin mimari yapıları, amaçlanan görevler ve tercih edilen veri kümeleri bakımından sınıflandırılmıştır.

Giriş ve literatür taramasının ardından üçüncü bölümde, önerilen yöntem detaylı bir şekilde ele alınmıştır. Burada önce model mimarisi genel hatlarıyla anlatılmış, ardından modelin her bir alt bloğu detaylı bir şekilde açıklanmıştır. Bu bölümde, kullanılan yardımcı yöntemler, çalışma yapıları, tercih edilme nedenleri ve ürettikleri çıktılar bakımından incelenmiştir.

Dördüncü bölüm ise modelin test edildiği üç veri kümesi üzerinedir. Burada veri kümeleri ve bütün aşamaları detaylı olarak anlatılmıştır. Veri kümelerinin elde edilişi, yapıları, eğitim için hazırlanmaları bu bölümde detaylandırılmıştır. Uygulanan veri artırma ve ön işleme teknikleri de bu bölümde yer almaktadır.

Beşinci bölümde geliştirilen modelin veri kümeleri üzerinde yapılan testlerinin kapsamlı sonuçları sunulmaktadır. Öncelikli olarak modelin her bir veri kümesi üzerindeki eğitim ayrıntıları anlatılmıştır. Seçilen platformlar, eğitim parametreleri ve kullanılan kayıp fonksiyonu ayrıntıları bu bölümde açıklanmıştır. Daha sonra modelin bileşenlerinin sonuca olan katkısını görmek ve bileşenlere ait en iyi parametreleri belirlemek amacıyla detaylı bir eksiltme çalışması yapılmıştır. Bu çalışmanın sonuçları da beşinci bölümde tartışılmıştır.

Beşinci bölümde son olarak, tez çalışması kapsamında elde edilen bütün sonuçlar ve bu sonuçların detaylı analizi sunulmuştur. Her bir veri kümesi üzerinde elde edilen direksiyon açısı ve hız sonuçları irdelenmiş, elde edilen sonuçlar literatürdeki diğer sonuçlarla karşılaştırılmıştır. Bölüm sonunda ise modelin çalışma süresi analiz edilmiştir. Burada, daha önceki çalışmalardan farklı olarak, modele ait her bir alt bileşenin çalışma süreleri sunulmuştur. Ayrıca bu analiz, tez çalışması boyunca kullanılmış ve modelin alt birimlerinin belirlenmesinde önemli rol oynamıştır.

Altıncı bölümde, CARLA sürüş simülatörü kullanılarak farklı amaçlar doğrultusunda otonom sürüş alanında çalışmak isteyen araştırmacılar için kolayca erişilebilecek, çeşitli amaçlara hitap eden kapsamlı bir veri kümesi sunulmuştur. Elde edilen veri kümesi, sensör çeşitliliği, toplanan veri miktarı ve veri çeşitliliği açısından mevcut literatürdeki en kapsamlı CARLA veri kümesi olmuştur. Bu bölümde, veri kümesinde kullanılan sensörler ve temel özellikleri, verinin içeriği ve yapısı detaylı olarak anlatılmıştır.

Tez çalışması neticesinde varılan sonuçlar ise yedinci bölümde yer almaktadır. Burada, çalışma süresince yapılan değişiklikler ve modelin ortaya çıkma süreci anlatılmış, elde edilen yapı kullanılarak varılan sonuçlar tartışılmıştır. Gelecekte yapılabilecek iyileştirmeler veya alternatif olabilecek çalışmalar da bu bölümde aktarılmıştır.

2. LİTERATÜR TARAMASI

Derin öğrenme alanındaki önemli ilerlemelerle birlikte, otonom sürüşle ilgili görsel algı, karar verme mekanizmaları ve sürüş stratejileri gibi farklı alanlarda birçok çalışma ortaya çıkmıştır. Bu etkileyici gelişmelerin temel nedenlerinden biri, bilgisayar kapasitelerinin son yıllarda hızla artmasıdır, bu da işlenebilir veri sayısının ve işleme hızının dramatik bir şekilde yükselmesine olanak tanımıştır. Bu, aynı zamanda derin öğrenmenin gelişimine de büyük katkı sağlamış ve görsel sürüş alanında önemli bir ivme kazanılmasına öncülük etmiştir. Direksiyon açısı ve hız tahmini, otonom sürüşle ilgili birçok uygulamada yerine getirilmesi gereken temel görevlerdendir. Bu tez çalışmasının da konusu olan uçtan uca öğrenme yöntemiyle bu görevlere yönelik farklı çalışmalar yapılmıştır (Abou-Hussein et al., 2019; Bojarski et al., 2016; L. Chen et al., 2023; Chi and Mu, 2017; Codevilla et al., 2018; Cultrera et al., 2020; Hecker et al., 2018; Hou et al., 2019; Johnson and Dana, 2020; Kim and Canny, 2017; Müller et al., 2018; Narayanan et al., 2020; Wan et al., 2021; T. Wang et al., 2022; Z. Yang et al., 2018).

Uçtan uca çözüm üzerine ilk çalışma, Savunma İleri Araştırma Projeleri Ajansı (DARPA) tarafından desteklenen ALVINN (Autonomous Land Vehicle in a Neural Network- Sinir Ağında Otonom Kara Aracı) projesidir (Pomerleau, 1988). Sistem, görüntü girdilerini kullanarak direksiyon açısını tahmin etmek için yalnızca tek bir gizli katmanlı ileri beslemeli bir sinir ağı kullanmaktadır. Derin öğrenmedeki hızlı ilerlemelere kadar, 1990'larda ve 2000'lerin başında bu alanda yapılan araştırma miktarı günümüze kıyasla oldukça sınırlı kalmıştır. LeCun ve arkadaşları (LeCun et al., 2005), mobil bir robotla engellerden kaçınma görevleri için 6 katmanlı bir Evrişimsel Sinir Ağı (ESA) kullanarak uçtan uca bir yaklaşım önermiştir. Bu proje aynı zamanda bir DARPA projesinin ön çalışmasıdır. Sonraki on yıl içinde, derin ağlarla ilgili önemli gelişmeler kaydedilmiş ve otonom sürüşle ilgili çalışmalar, kendi kendine sürüş için uçtan uca bir çözüme dayanan NVIDIA projesiyle (Bojarski et al., 2016) ivme kazanmıştır. NVIDIA grubu, yalnızca tek bir ön kameradan gelen görüntü verilerini kullanarak direksiyon komutları üretmek için beşi evrişimsel katman olmak üzere 9 katmanlı bir ESA önermiştir. Bu çalışmada, sadece görsel girdiye dayalı çok görevli bir uçtan uca öğrenme sistemi tasarlanmıştır. Bu model, sınırlı sayıda veriyle bile yol çizgisi ve kaldırım gibi belirleyici unsurların olmadığı yollarda dahi başarılı tahminler yapabilme yeteneğiyle birçok araştırmacının, enstitünün ve otomotiv endüstrisinin dikkatini çekmiştir. İlerleyen yıllarda hızlı bir şekilde pek çok üniversitede otonom sürüş laboratuvarları oluşturulmuş,

ticari şirketler ortaya çıkmış ve büyük otomobil firmaları tarafından buraya yapılan yatırımlar artırılmıştır.

2.1 Görme Tabanlı Sürüşte Model Mimarileri

Derin öğrenme sayesinde, otonom sürüş teknolojileri için hızla adapte edilen yeni teknikler geliştirmiştir. Özellikle ESA'lar, genellikle kameralarla yapılan birçok uygulamada tercih edilen uçtan uca mimarilerin öncüleri olmuştur (Bojarski et al., 2016; Codevilla et al., 2018; L. Han et al., 2022; Jiang et al., 2020; Z. Yang et al., 2018). Otonom sürüşte, kamera-LiDAR (Caltagirone et al., 2019; Sallab et al., 2018), kamera-radar (Long et al., 2021; Zuo et al., 2021), kamera-radar-LiDAR (Cai et al., 2020; Jahromi et al., 2019b) ve kamera-IMU (L. Han et al., 2019) gibi farklı sensörlerin kombinasyonları, sahnedeki çeşitli özellikleri yakalamak için kullanılmıştır. Çoklu modalite ile LSTM (Long Short-Term Memory) yapıları (Hochreiter and Schmidhuber, 1997) da mimariye dahil edilerek zamansal bilgiden faydalanılması ve neticede ortaya çıkan kararların daha sağlam olması, sistemin önemli hatalara neden olacak kararlardan kaçınması sağlanmıştır (Hecker et al., 2018; Narayanan et al., 2020). Ayrıca, eylem komutlarını tahmin etmek için kodlayıcı-kod çözücü yapıları, uçtan uca öğrenme sürecinde tercih edilen bir diğer mimari olarak öne çıkar (Huang et al., 2021; Tai et al., 2019). Bu yaklaşımlar, derin öğrenme tekniklerinin otonom sürüşte daha güvenilir ve adaptif bir şekilde kullanılmasına olanak tanır.

ESA tabanlı mimarilere sahip, kontrol komutu tahmini yapan bazı çalışmaların detayları şu şekildedir:

Deep Steering (Chi and Mu, 2017) modeli, ardışık 15 görüntüyü alarak uzamsal-zamansal konvolüsyona sokar. Elde ettiği özellik vektörünü iki farklı kolda kullanır. Birinci kol, tam bağlı dört katman sonunda bir özellik vektörü oluşturur. İkinci kol, bir LSTM biriminden sonra aynı boyutta özellik vektörü oluşturur. Daha sonra hepsi aynı boyutta olmak üzere, evrişim katmanlarının her birinden elde ettiği özellik vektörlerini, evrişimsel katmanların ucundaki tam bağlı katmanların sonucunda elde ettiği özellik vektörünü ve LSTM'den elde ettiği özellik vektörlerini birleştirerek yine aynı boyutta bir özellik vektörü oluşturur. Ardından elde ettiği bu birleşik özellik vektörünü ardışık bağlı LSTM ve tam bağlı katmandan geçirerek tahmin yapar.

MSINet (T. Wu et al., 2019) modeli ise, direksiyon açısı tahmini için farklı ölçeklerden gelen uzamsal bilgileri kodlamayı amaçlayan Çok Ölçekli Uzamsal-

Zamansal Entegrasyon (MSI) modülüne dayalı uçtan uca bir sürüş modeli önermektedir. Dört adet Conv-LSTM modülü, direksiyon açısını oluşturmak için kodlayıcının farklı seviyelerine bağlanır. Ayrıca, bu çalışma modelinin uzamsal-zamansal özelliklerini artırmak için eğitim sırasında gelecekteki sıralı bilgileri (gelecek dört görüntü) kullanmıştır. MSI-Net modeli de bir önceki çalışma gibi geçmiş n adet görüntüyü giriş verisi olarak kullanır. Bu tez çalışmasında önerilen model ise yalnızca tahmin anındaki RGB görüntüyü kullanmaktadır.

Johnson ve Dana'nın (2020)'de önerdikleri Feodal Steering modeli, iki bölümden oluşmaktadır: bir yönetici ağı ve bir işçi ağı. Yönetici ağ, üst düzey eylemlerin planlanmasından sorumluyken, işçi ağı, arabanın yönlendirilmesi gibi alt düzey eylemleri gerçekleştirir. İşçi ağın mimari yapısı ardışık evrimsel ağlar, LSTM birimi ve tam bağlı katmanlardan oluşur. Geçmişe dönük m adet direksiyon açısı tahmini yönetici ağıya verilir. İşçi ağın girişi ise ardışık m görüntü, yönetici ağdan çıkış olarak alınan hedef direksiyon açısı değeri ve geçmişe dönük m adet direksiyon açısından oluşur. Benzer şekilde buradaki ağ yapısında da giriş verisi olarak çok sayıda görüntü kullanılması durumu vardır.

Multimodal RNN-LSTM modeli (Abou-Hussein et al., 2019), beş katmanlı bir ESA ağından ve ardından gelen bir LSTM biriminden oluşur. Burada giriş verisi olarak t anındaki görüntü ve bu görüntüden elde edilmiş optik akış görüntüsünün yatay ve dikey bileşeni birleşik üç görüntü olarak kullanılmıştır.

Z. Yang ve diğ. (2018) tarafından önerilen MMMTN modeli, bir görsel kodlayıcı ve bir hız kodlayıcı içerir. Görsel kodlayıcı bir görüntüyü girdi olarak alırken, hız kodlayıcı önceki hız dizilerini alır. Direksiyon açısı tahmini, bir evrimsel sinir ağı ile yapılırken, hız tahmin bloğu farklı bir yapıya sahiptir. Burada ardışık hız verileri girdi olarak alınıp bir LSTM biriminden geçirilir. Daha sonra, iki tam bağlı katmandan geçen özellikler, görüntü tahmin ağının ara katmanından gelen özellik vektörü ile birleştirilip hız tahmini yapılır. Bu durumda, hız ve direksiyon açısı tahmin ağları birbirinden ayrı olarak çalışmaktadır. Her iki bloğun tam bağlı katmanları ayrıdır. Ancak bu tez çalışmasında önerilen modelde, LSTM kolundan ve görüntü kollarından gelen yüksek seviye özellikler birleştirildikten sonra tam bağlı katmanlardan geçilmekte ve sonrasında iki tahmin koluna ayrılıp direksiyon açısı ve hız tahmini yapılmaktadır. Bu durumda hem direksiyon açısı hem de hız tahmini yapılırken modelin bütün girişlerinden faydalanılmaktadır.

N²C (Neural Network-based Controller) (Azam et al., 2021) çalışmasında, tahmin için iki ayrı model sunulmuştur. Birinci model, giriş görüntüsünden bir ESA aracılığıyla direksiyon açısını ve hızı tahmin eder. Daha sonra bu modelin çıktıları, düşük seviyeli kontrol parametreleri oluşturmak için kontrolör modeline girdi olarak verilir. LSTM tabanlı kontrolör, bu direksiyon açısı ve hız girdilerine dayalı olarak gaz, fren ve torku tahmin eder.

Hou ve diğ. (2019) tarafından önerilen FM-Net modeli, ana yapı olarak ardışık evrişimsel katmanları takip eden tam bağlı katmanlar ve ardından gelen bir LSTM bloğu içermektedir. Ancak FM-Net modeli, ana omurga yanında (D. Wang et al., 2019)'da olduğu gibi yardımcı ağlar içerir. Burada yardımcı ağlardan biri semantik segmentasyonu diğeri ise optik akışı hesaplamaktadır. Yazarlar, yardımcı ağların ara evrişimsel katmanları ile ana ağın ara evrişimsel katmanları arasında bir ilişki kurmuşlardır. Burada yardımcı ağların katmanlarındaki özelliklerin ana omurga tarafından taklit edilmesi için eğitim sırasında kayıp hesaplanmıştır. (D. Wang et al., 2019)'da olduğu gibi, burada da test aşamasında yardımcı ağlar kullanılmamıştır. FM-Net'te giriş verisi olarak ardışık 10 görüntü kullanılmış, çıkışta ise direksiyon açısı, hız ve tork tahmin edilmiştir.

SCMTL (Abbas et al., 2021) modeli, istatistiksel olarak ilişkili görevler için bir öğrenme mimarisi önermektedir. Ancak, çalışmada anlatılan ve grafikleri sunulan veriler arasında (negatif korelasyonlu hızlanma-fren çifti dışında) istatistiksel bir ilişki gözlemlenmemiştir. Örneğin, direksiyon açısı ve hız arasında bir ilişki bulunmamaktadır. Bu modelde tek bir giriş görüntüsü kullanılarak belirli bir katmana kadar evrişimsel konvolüsyon uygulanmıştır. Daha sonra buradaki özellik vektörü, direksiyon açısı, fren basıncı ve hızlanma için üç farklı görev koluna ayrılmış ve her bir görev için bireysel kollarındaki evrişimsel katmanlarla tahmin yapılmıştır. Mimarisi, göreve özgü üç kola ayrılmış ESA'ya dayanmaktadır.

Yuan ve diğ. (2021)'de dengesiz eğitim kümeleri için tasarlanan SteeringLoss'un etkinliğini üç farklı model üzerinde göstermiştir. Bu kayıp fonksiyonu, genellikle düz gidiş açısı etrafında yoğunlaşan direksiyon açısı verisine özgü olarak tasarlanmıştır. Daha sonra bu fonksiyon, üç farklı model ile değerlendirilmiş ve üç tip deney yapılmıştır: Birinci deney, önerilen kayıp fonksiyonunun dengesiz eğitim kümesinde kullanımını; ikinci test, kayıp fonksiyonunun kullanımı ile ilgili tarifleri ve ince ayarları; üçüncü test ise kayıp fonksiyonunun farklı modellerle kullanımında elde edilen sonuçları göstermektedir.

2.2 Otonom Sürüşte Amaçlanan Görevler

Bir aracın bir insan tarafından sürüldüğü düşünüldüğünde, birden fazla görevin aynı anda gerçekleştirildiği görülmektedir. Araçtan da aynı davranış beklenmektedir. Bu nedenle çalışmalar hem ana görev hem de yan görevler açısından çeşitlilik kazanmıştır. Örneğin, şerit takibi (Z. Chen and Huang, 2017; Yuan et al., 2018), navigasyon (Hecker et al., 2018), güzergâh planlama (Cai et al., 2019; C. Chen et al., 2015) ve sürüş politikası (Müller et al., 2018; Xu et al., 2017) gibi konular, eylem komutları tahminiyle doğrudan ilişkilendirilen ve araştırmalarda detaylıca incelenen temel alanlardan sadece bazılarıdır.

Son dönemlerde, algısal sistemlere dayalı olarak eylem komutlarını tahmin etmeye yönelik yapılan çalışmaların yanı sıra otonom sürüş için faydalı çıktılar üreten farklı yöntemler de önerilmiştir. Bu araştırmalar, genellikle lokalizasyon/ poz tahmini (C. Li et al., 2021; Wagstaff and Kelly, 2021), haritalama (Wong et al., 2021), görsel odometri (C. Chen et al., 2019), nesne tespiti (Jahromi et al., 2019a; Kim et al., 2018; Person et al., 2019), yaya tespiti (Person et al., 2019), yol tespiti (Caltagirone et al., 2019; C. Han et al., 2022; Shao et al., 2022), derinlik tahmini (Lee et al., 2022; Sun et al., 2023; Wagstaff and Kelly, 2021; Wei et al., 2022; Xie et al., 2023; J. Zhang et al., 2021) ve segmentasyon (Caltagirone et al., 2019; Jahromi et al., 2019a; Shao et al., 2022; Sirohi et al., 2023) gibi alanlara odaklanmaktadır. Farklı disiplinler arasındaki bu yoğun etkileşim, otonom sürüşte karar mekanizmalarını, güvenilirliği ve çeşitliliği artırarak daha sağlam bir zeminde geliştirmeye yardımcı olmaktadır. Bu çalışmalar, sadece tek bir bileşen üzerinde değil, bütünsel bir sistem yaklaşımıyla otonom araçların performansını iyileştirmeyi hedeflemektedir.

2.3 Eğitim ve Test Ortamları

Otonom sürüş alanındaki çalışmaların bir kısmı son yıllarda gerçek araçlar üzerinde değil, simülasyon ortamlarında ya da gerçek ortam verileri kullanılmadan gerçekleştirilmiştir (Cai et al., 2020; Codevilla et al., 2019; Tai et al., 2019; T. Wang et al., 2023). Gerçek araçlarla, gerçek yollarda veri toplamak hem çok maliyetlidir hem de uzun süren bir işlemdir. Ayrıca veri toplanan alandaki araç yoğunluğu, yaya durumu, gün ışığı ve hava durumu gibi etkenler geliştiriciler tarafından kontrol edilemez. Ancak gerçek ortam verilerinin kullanılmadığı çalışmalar, bu değişkenleri kontrol etme imkânı sunan simülasyon ortamlarında (Dosovitskiy et al., 2017; Eric Espié et al., 2005)

uygulanmaktadır. CARLA sürüş simülatörü bugün en çok kullanılan, otonom sürüş sistemlerinin geliştirilmesi ve test edilmesi için tasarlanmış bir açık kaynaklı simülasyon platformudur. Bu platform, gerçek dünya koşullarını sanal bir ortamda simüle ederek, yapay zekâ modellerinin ve sürüş stratejilerinin geliştirilmesine olanak tanır. CARLA’da kamera, LiDAR, radar, IMU gibi çok sayıda farklı senör vardır. Sürüş çevresi, hava koşulları, araç ve yaya sayısı gibi etkenler kontrol edilebilir. Ayrıca CARLA, kullanıcıların kendi haritalarını da oluşturmalarına olanak tanır.

Diğer yandan, bazı çalışmalar ise kamera görüntüleri, LiDAR verileri ve hız, tork ve direksiyon açısı gibi bazı kullanışlı kayıtları içeren ve gerçek bir araçla oluşturulmuş veri kümeleri (Hecker et al., 2018; Sully Chen, 2023; Udacity, 2017; Xu et al., 2017) üzerinde uygulanmıştır.

Önerilen uçtan uca yöntem, gerçek araç verisi kullanan Udacity ve Sully Chen veri kümeleri ile CARLA sürüş simülatöründe oluşturulan veri kümesini kullanılarak üç farklı ortamda değerlendirilmiştir. Bu değerlendirme sürecinde, modelin hem gerçek dünya koşullarındaki davranışını hem de simülasyon ortamındaki davranışını görmek ve elde edilen sonuçların veri kümesine özgü olmadığını doğrulamak için üç farklı veri kümesi kullanılmıştır

Gerçek araç verisi ile oluşturulan yaklaşımlar, otonom sürüş teknolojilerinin gerçek dünya senaryolarına ne kadar uygun olduğunu belirlemede önemli bir rol oynar. Bu çalışmalar, sistemlerin gerçek koşullarda ne kadar güvenilir olduğunun anlaşılmasını sağlayarak otonom sürüşün pratik uygulamalarına katkıda bulunur. Sanal ortam verileri ile yapılan testler de daha geniş bir yelpazede veri çeşitliliği sunması ve değişkenlerin kontrol edilebiliyor olması nedeniyle sistemin başarısını artırmada kritik bir rol oynar. Bu ortamlarda yapılan testler, algoritmaların farklı senaryolarda nasıl tepki verdiğinin anlaşılmasına yardımcı olur. Bu da otonom araçların çeşitli ve nadir senaryolarda nasıl davrandıklarını öngörmek için önemlidir. Her ikisinin de kendine göre avantajları ve dezavantajları olan bu iki yaklaşım bir arada kullanılarak, otonom sürüş teknolojilerinin daha güvenli, etkili ve geniş bir senaryo yelpazesinde çalışabilirliğine imkân sağlanmış olur.

3. RENK DOKUSU, UZAYSAL VE SEMANTİK SEGMENTASYON BİLİNÇLİ SÜRÜŞ MODELİ

Otonom sürüşün başarısı, çevresel unsurları doğru anlama ve bu bilgiler doğrultusunda doğru kararlar üretme yeteneğiyle doğrudan ilişkilidir. Bu çalışmanın odak noktası, otonom araçların çevresel algılama süreçlerini derinlemesine incelemek ve özellikle monoküler kameraların sunduğu görüntü verileri üzerinden sürüş sahnelerinin anlamlandırarak sağlıklı kontrol komutları üretmektir.

Görsel algılamayı temel alan bu model, monoküler bir kameradan alınan görüntü üzerinden sürüş sahnesini analiz eder. Model, bu kamera görüntüsünü yine kendisinden türetilen diğer görüntülerle bir araya getirip, derin öğrenme algoritmalarıyla sürüş kararları üretmeyi hedeflemektedir. Çalışmanın asıl amacı, sadece tek bir sensör yardımıyla sürüş sahnesini algılamak ve elde edilen RGB görüntü ve bu görüntüden türetilen derinlik görüntüsü ve anlamsal olarak bölümlenmiş görüntüleri entegre ederek sürüş doğruluğunu ve güvenliğini artırmaktır. Böylece RGB görüntü, derinlik görüntüsü ve anlamsal olarak bölümlenmiş görüntü olmak üzere üç farklı görüntü sisteme girdi olarak verilir. Önerilen model, üç girdiden elde edilen görsel ipuçlarından yararlanarak bir derin öğrenme ağı ile uçtan uca direksiyon açısı ve hız tahmini yapar.

Buradaki temel fikir, önceki araştırmalarda kullanılan yüksek maliyetli sensörler yerine, basit kamera görüntüsü üzerinden güvenli görsel sürüş şartlarını oluşturmaktır. Bu çalışma, çoklu sensörlerin geleneksel sensör füzyon yöntemleri ile entegrasyonu yerine, sadece kamera görüntüsü kullanarak derin öğrenmeye dayalı kontrol komutlarını tahmin etme üzerine yoğunlaşmaktadır.

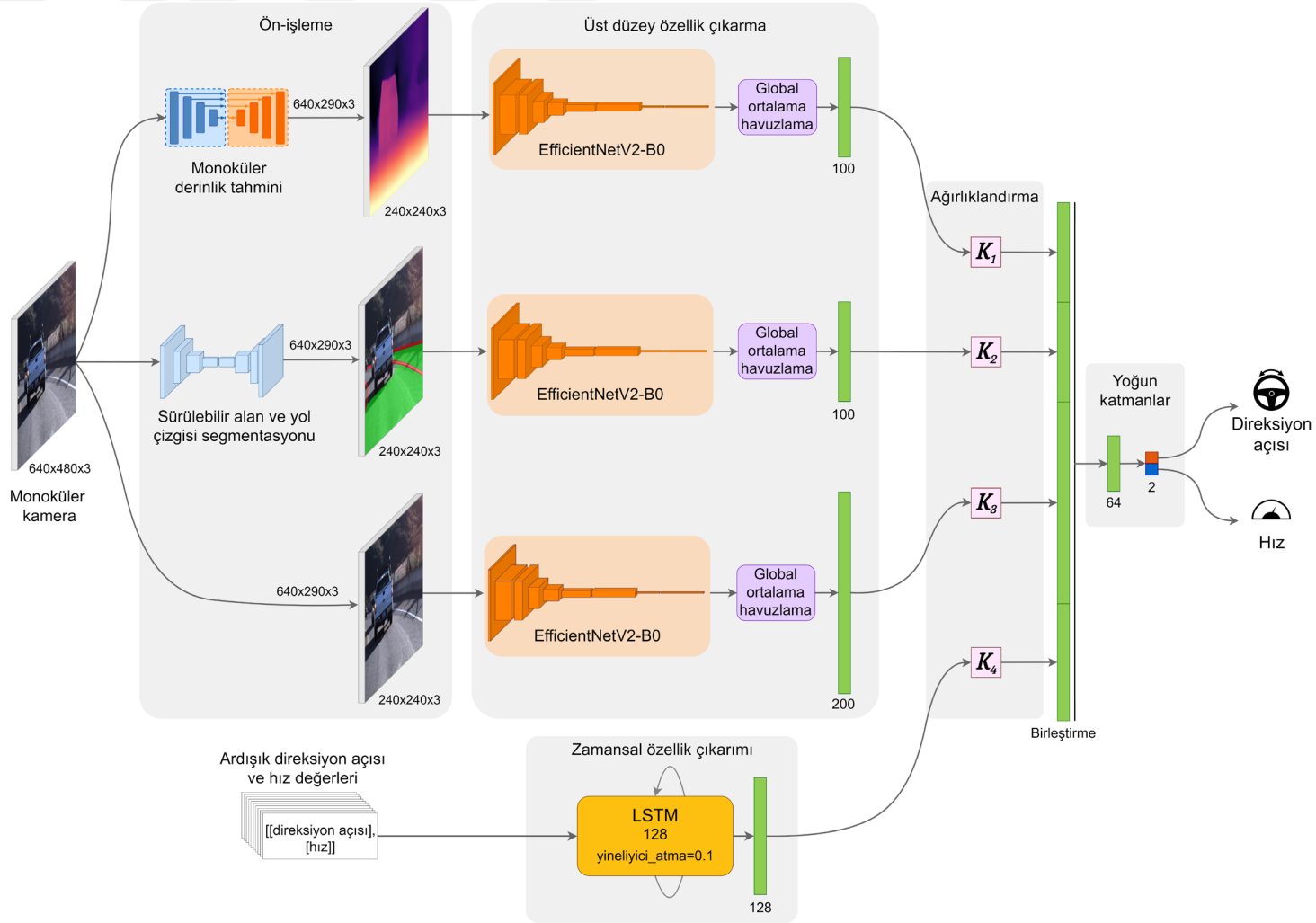
Çalışmanın temel motivasyonu, insan sürücülerin LiDAR veya radar gibi yüksek maliyetli sensörlere ihtiyaç duymaksızın sadece görsel algılarıyla araç kullanabilme potansiyelidir. Özellikle, insan sürücülerin gece şartlarında dahi sadece araç farlarından veya yol aydınlatmasından yararlanarak başarılı bir şekilde sürüş yapabilmesi, görsel algıların etkinliğini göstermesi açısından önemli bir kanıttır. Bu durum, görsel algıların araç kullanımında ne kadar kritik bir rol oynayabileceğini göstermektedir.

Bugün mevcut olan bilgisayarlı görü teknikleri insan görme sistemi kadar başarılı sonuçlar elde edemediği için, yardımcı sensörler kullanılarak otonom sürüş başarısı arttırılmaya çalışılmaktadır. Genellikle otonom araç geliştirme süreçlerinde, kamera, radar ve LiDAR gibi çeşitli sensörler kullanılmaktadır. Ancak, özellikle Tesla gibi

sektörde öncü konumda olan bazı şirketlerin, LiDAR gibi sensörleri kullanmama tercihi oldukça dikkat çekicidir.

Otonom araçlarda görsel algılamada yaşanan eksiklik kameraların yetersiz veri sunmasından çok, bu sensörlerden alınan verinin anlamlı bilgiye dönüştürülmesindeki başarının henüz bir insanın yerini dolduracak kadar yeterli olmamasındandır. Bilgisayarlı görü çalışmalarının sonuçlarındaki başarı özellikle son yıllarda hızlı bir şekilde artmaktadır. Kısa bir zaman diliminde bu teknolojinin insan görme sistemindeki başarıyı yakalaması ve hatta geçmesi şaşırtıcı olmayacaktır. Sürüş güvenliğini arttırmak adına farklı sensör tiplerinin de kullanılması elbette faydalı olacaktır. Ancak burada maliyet de üretici firmalar için önemli bir parametredir. Kamera dışındaki sensörlerin maliyetleri görece yüksektir. Ayrıca çok sayıda sensör kullanıp sensör füzyon yöntemi ile birleştirmek de gerçek zamanlı çalışan otomobiller için zaman açısından problem teşkil edebilir. Araştırma firmalarının test araçlarında görüldüğü gibi üzerinde çok sayıda sensör bulunan bir araç yerine bugün şerit takip özelliğine sahip araçlarda olduğu gibi sadece dikiz aynası civarında basit bir gömülü kamerası olan bir araç çok daha kabul edilebilir bir tasarım olacaktır. Aracın üzerinde ekstra çıkıntılı çok sayıda sensörün aracın aerodinamiğini olumsuz etkilemesinin yanında hem sensörler darbeye açıktır hem de bu pahalı cihazların açıkta oluşu güvenlik zafiyeti yaratabilir.

Bilgisayar sistemlerinin her geçen gün artan veri işleme kapasitesi yakın gelecekte sensör füzyon işlemi için gerekli süreyi çok rahat sağlayacak olsa da görme sistemlerinin otonom araçlar için tek başına yeterli olması ve insan görme sistemi kadar başarılı sonuçlar üretmesi birçok zorluğu ortadan kaldıracak gibi üretici maliyetlerini de düşürecektir. Bu durumda, diğer tip sensörler sürüş güvenliğini ekstra arttıran yardımcı ekipmanlar olarak gerekli durumlarda kullanılabilir.



Şekil 3.1. Önerilen çok girdili çok görevli uçtan uca modelin mimarisi. Ağda her bir görüntü için ayrı ESA ağı eğitilmiştir. EfficientNetV2 modülleri, uzaysal özellik kodlaması için kullanılırken, LSTM modülü zamansal özelliklerin elde edilmesi için kullanılmıştır

3.1 Model Mimarisine Genel Bakış

Bu mimarideki temel amaç, sistemi sanki üç farklı sensör besliyormuş gibi üç farklı girdi yaratmak, böylece modele bir RGB görüntüsü, bir derinlik görüntüsü ve bir de anlamsal olarak bölümlenmiş görüntü olmak üzere üç ayrı giriş sunmaktır. Bu şekilde, sisteme farklı türde veriler sunulurken, özellikle farklı duyarlılık seviyelerine sahip sensörlerden elde edilen bilgilerin birleştirilmesi hedeflenir. Bu tercih aslında çalışmanın ana motivasyonunu yansıtmaktadır. Böylece, birden fazla girişle sisteme farklı bakış açıları kazandırarak, genel performansı artırmak ve daha sağlam sonuçlar elde etmek amaçlanmaktadır.

Daha önce bahsedildiği gibi insan sürücüler otomobil kullanırken güvenli sürüş için görme yetileri dışında başka algılara doğrudan gerek duymazlar. Benzer şekilde lazer veya radar gibi teknolojik sensörlerden de faydalanmazlar. İnsan görme sistemi, derinlik tahmin edebilme ve semantik segmentasyon yapabilme gibi olağanüstü yeteneklere sahiptir. İnsan beyni her iki gözden gelen bilgileri birleştirerek çevredeki dünyaya ilişkin tutarlı ve üç boyutlu bir algı oluşturur. İnsan görme sistemi ayrıca görsel sahneyi anlamlı bileşenlere ayırmak için bir tür semantik segmentasyon gerçekleştirir. İnsan görüşünün bu temel bileşenleri sürüş ortamının sağlıklı bir şekilde algılanmasını ve anlaşılmasını sağlar. Aslında önerilen modeldeki yapı da buna çok benzerdir. İnsanda, gözden bilginin alınıp içeride işlenmesine benzer şekilde, kameradan görüntü alındıktan sonra arkadaki yapıda derinlik tahmini ve semantik segmentasyon yapılır. Burada bir LiDAR sensöründen de derinlik bilgisi alınıp sensör füzyonu yaparak sahne algılaması yapmak mümkündür (Caltagirone et al., 2019; Sallab et al., 2018).

Şekil 3.1, RGB kamera görüntüsü üzerinden Evrişimsel Sinir Ağları (ESA) ve LSTM kullanarak hız ve direksiyon açısı tahmini yapan çok girişli, çok görevli ağıın mimarisine genel bir bakış sunmaktadır. Sistem, tek bir sensör girdisi olan standart bir monoküler kameradan elde edilen RGB görüntü (I^{rgb}) üzerinde işlem yapmaktadır. Ön-işleme aşamasında bu RGB görüntüden bir derinlik görüntüsü (I^d) ve bir de semantik olarak bölümlenmiş görüntü (I^{seg}) elde edilir. Bu ön-işleme adımlarının ayrıntıları, hangi yöntemlerle görüntülerin nasıl elde edildiği Bölüm 3.2’de detaylı bir şekilde açıklanacaktır.

Ön işleme aşamasında elde edilen derinlik görüntüsü ve semantik olarak bölümlenmiş görüntüler, RGB görüntü ile birlikte, $\langle \mathcal{I}^{gb}, \mathcal{I}^d, \mathcal{I}^{seg} \rangle$ olarak sisteme girdi olarak sunulurken, aynı zamanda özellik çıkartma alt ağlarının girdilerini oluşturur. Bu aşamadan sonra, temelde karşılaşılan problem, çok girdili ve çoklu görevli bir öğrenme problemine dönüşür. Önerilen model RGB dokusu (Texture), uzaysal (Spatial) ve anlamsal (Semantic) ipuçlarını dikkate aldığı için TS2DNet olarak isimlendirilmiştir.

Çalışmada kullanılan eğitim verisi bir insan sürücünün kullandığı araç ile toplanan $\mathcal{D} = \{(\mathbf{p}_i, \mathbf{c}_i)\}_{i=1}^N$ algı-kontrol çiftlerinden oluşur; burada $\mathbf{p}_i = \{\mathcal{I}^{gb}\}$, RGB görüntülerden oluşan algıyı temsil ederken, $\mathbf{c}_i = \{s_i, v_i\}$ ise direksiyon açısı ve hızdan oluşan kontrol çiftini ifade eder. Eğitim veri kümesi sağlandığında önerilen ağ, RGB görüntüden ($\mathcal{I}^{gb}$) direksiyon açısını (s_i) ve araç hızını (v_i) tahmin etmek üzere eğitilir. t anında sağlanan RGB görüntü ($\mathcal{I}^{gb}$) göz önünde bulundurulduğunda genel ağ modeli şu şekilde yazılabilir:

$$(\tilde{s}_t, \tilde{v}_t) = \mathcal{F}(\mathcal{H}(\mathcal{I}_t^{rgb}, \mathcal{I}_t^d, \mathcal{I}_t^{seg}), \mathcal{U}(s_{t-k:t-1}, v_{t-k:t-1})). \quad (3.1)$$

Denklem (3.1)'de, $\tilde{s}_t$ tahmin edilen direksiyon açısını, $\tilde{v}_t$ tahmin edilen araç hızını, $s_{t-k:t-1}$ ve $v_{t-k:t-1}$ sırasıyla son k direksiyon açısını ve son k hız değerini belirtir. $\mathcal{U}(\cdot)$, $\langle s_{t-k}, \dots, s_{t-1} \rangle$ ve $\langle v_{t-k}, \dots, v_{t-1} \rangle$ dizilerindeki zamansal bağımlılıkları yakalarken, $\mathcal{H}(\cdot)$, uzaysal öğrenme için üç görsel girdi üzerinde işlem yapar. $\mathcal{F}(\cdot)$ olarak gösterilen ana model, direksiyon açısı ve araç hızını tahmin edebilmek için bu bileşenleri birleştirir. Burada amaç hem direksiyon açısı hem de araç hızı tahmininde her üç girdiden en iyi şekilde faydalanmaktır. Bu hedefe ulaşmak için her bir görev için ayrı ayrı eğitilen farklı modeller kullanmak yerine çok görevli öğrenme yaklaşımı tercih edilmiştir.

Çok görevli öğrenme yöntemi, paylaşılan bir temsili kullanırken farklı görevleri paralel olarak eğitmek suretiyle genelleme performansını artırır. Tüm görevler için eğitilmiş paylaşılan bir katman kullanan bu yapı, her görevin öğrenilen bilgilerinin diğer görevlere olumlu katkılarda bulunabileceği bir etkileşim sağlar (Naik and Rangwala, 2018). Bu da genel sistem performansını olumlu yönde etkiler. Bu yapı, özellikle sürüş ortamlarındaki karmaşıklık ve çeşitlilikle başa çıkabilen, daha etkili ve genel bir otonom sürüş modeli oluşturmayı amaçlar. Ayrıca, bu yaklaşım hesaplama süresini de azaltan önemli bir faktördür.

3.2 Uzaysal Çıkarım ve Semantik Segmentasyon İçin Ön-İşleme Teknikleri

Otonom sürüş sistemlerinin geliştirilmesi, sürüş güvenliği ve etkinliği için önemli bir hedef olarak belirlenmiştir. Bu bağlamda, uzaysal çıkarım ve semantik segmentasyon, araçların çevresini daha iyi anlamalarına ve çeşitli yol koşullarında daha güvenli bir sürüş sağlamalarına olanak tanır.

3.2.1 Uzaysal çıkarım – monoküler derinlik tahmini

İnsan sürücüler açısından düşünüldüğünde, sürüşün çok önemli bir yönü olan sahnenin derinliğini algılama becerisi, stereo görüş aracılığıyla sahnenin uzaysal yapısı hakkında öngörüler sunar. Böylece sürücüler çevrelerini daha iyi algılayarak özellikle karmaşık trafik koşulları, kötü aydınlatma ve yetersiz yol çizgisi gibi durumlarda daha güvenli bir şekilde hareket edebilirler. Ayrıca, stereo görüş derinlikle birlikte ölçek bilgisi de sunduğu için nesnelerin konumunu ve hareketlerini daha iyi yorumlamaya yardımcı olur, bu da yine sürüş güvenliğini arttıran önemli parametrelerdendir. Canlılarda kusursuz çalışan bu sistem araçlar için düşünüldüğünde bazı dezavantajlara sahiptir. Stereo görüntü elde etmek için birden fazla kamera kullanımının tek bir kameraya kıyasla daha yüksek maliyeti vardır. Stereo görüş sistemleri, kameralar arasında hassas kalibrasyon ve senkronizasyon gerektirir, bu da karmaşıktır ve hata olasılığını arttırabilir. Monoküler derinlik tahmini, sistemi basitleştirerek bu karmaşıklıkları ortadan kaldırır. Ayrıca, stereo görüş sistemi, düşük dokulu veya tekrarlayan desenlerin olduğu sahnelerde ve nesnelerin her iki görüşte de farklı gözüktüğü veya bir kamerada görünmediği durumlarda sınırlı performans sergileyebilir. Kullanılan kameraların çözünürlüğü ve aralarındaki mesafe, doğru derinlik tahmininin etkili aralığını belirler (Martins et al., 2018).

Yukarıda bahsedilen dezavantajlar ve kullanılması düşünülen gerçek dünya veri kümelerinde görüntü olarak yalnızca monoküler kamera verisinin bulunması nedeniyle derinlik tahmininde monoküler derinlik tahmini yönteminin tercih edilmesine karar verilmiştir. Bu çalışmada, ağa ikinci girdi olarak verilen derinlik görüntülerini oluşturmak için Monodepth2 (Godard et al., 2019) monoküler derinlik tahmin yöntemi kullanılmıştır. Monodepth2, evrimsel sinir ağı mimarisinin hâkim olduğu bir kodlayıcı-dekoder yapısına sahiptir. Monodepth2'nin temel amacı, tek bir görüntüden sahnenin üç boyutlu temsilini sağlayan doğru derinlik haritaları tahmin etmektir. Model, kendi kendine

gözetimli öğrenme yaklaşımını kullanır. Bu yaklaşımla model, gerçek derinlik etiketlerine ihtiyaç duymadan, nesneler arasındaki geometrik ilişkiden derinlik öğrenir.

Monodepth2 modeli, tek bir görüntü üzerinde derinlik haritasını tahmin etmek için eğitilir. Bu süreçte model, derinlik tahminini optimize etmek için bir kayıp fonksiyonu olan yeniden yansıtma hatasını azaltmaya odaklanır:

$$L_p = \sum_{t'} pe(I_t, I_{t' \rightarrow t}) \quad (3.2)$$

$$\text{ve } I_{t' \rightarrow t} = I_{t'} \langle proj(D_t, T_{t' \rightarrow t}, K) \rangle. \quad (3.3)$$

Denklem (3.2) ve (3.3)'te $I_{t'}$ kaynak görüntü, I_t hedef görüntü ve $T_{t' \rightarrow t}$ de göreceli pozdur. D_t , yoğun derinlik haritasını temsil ederken, L_p fotometrik yeniden yansıtma hatasını ifade eder; burada pe fotometrik yeniden yapılandırma hatasına karşılık gelirken, $proj()$ ise $I_{t'}$ içindeki D_t öngörülen derinliklerinin iki boyutlu koordinatlarını hesaplar, K , önceden hesaplanmış esasları ifade eder ve $\langle \rangle$ örnekleme operatörünü belirtir.



Şekil 3.2 Üst sırada Udacity veri kümesinden rastgele alınmış üç adet orijinal RGB görüntü ve alt sırada bu görüntülerin Monodepth2 ile elde edilmiş derinlik görüntüleri

Fotometrik yeniden yansıtma hatası, modelin tahmin ettiği derinlik haritasını kullanarak kaynak görüntüden yeniden projeksiyon yapılarak elde edilen görüntü ile tahmin edilen derinliğe dayalı olarak oluşan farklı ölçer. Bu hata, derinlik tahminlerinin doğruluğunu değerlendirmek ve iyileştirmek için önemli bir ölçüdür. Daha az hata, modelin gerçek dünya sahnelerini daha doğru bir şekilde anlamasına ve işlemesine olanak

tanır. Monodepth2 modeli, verimli işlem hızı ve alternatif çalışma yöntemlerine göre daha yüksek başarı oranı nedeniyle monoküler derinlik tahmini için seçilmiştir.

Şekil 3.2’de, Udacity veri kümesinden alınmış bazı sürüş sahnelerinin Monodepth2 ile elde edilmiş derinlik görüntüleri görülmektedir.

3.2.2 Semantik segmentasyon ile özellik çıkarımı

Bir sahneyi anlamak, renk dokusunu yorumlamak ve derinlik tahmini yapmaktan daha fazlasını içerir; semantik (anlamsal) segmentasyon da bir görüntünün görsel içeriğini kavramak için çok önemlidir. Semantik segmentasyon modelleri, görüntüdeki her bir piksel için anlamsal bir etiket olasılığı hesaplar. Bu da sahnedeki ayrıntılar ve görüntünün kapsamı hakkında zengin bilgiler sunar. Semantik segmentasyon için genellikle kodlayıcı-kod çözücü yapısında evrimsel sinir ağları kullanılır. Kodlayıcı, görüntü anlamak için tasarlanan çoğu ESA’da olduğu gibi, ardışık evrim katmanlarından oluşur. Hesaplama maliyetinin yüksek olmaması için çözünürlük her aşamada azaltılır. Kodlayıcı bu nedenle giderek artan bir alanda daha özet anlamsal ve görsel kavramları öğrenir. Kod çözücü kısmında ise öğrenilen özellikler büyütülür ve segmentasyon sonucuna dönüştürülür. Bu yöntem, semantik bölümlenmede ilk olarak (Shelhamer et al., 2017) tarafından kullanılmıştır. Tamamen Evrimsel Ağlar (Fully Convolutional Networks- FCNs) olarak isimlendirilen bu yapı daha sonra çok popüler olmuş ve sonraki segmentasyon çalışmalarına (L.-C. Chen et al., 2018; X. Li et al., 2017; Zhao et al., 2017) ışık tutmuştur.

Segmentasyon işlemi uygulanacak $\mathcal{I} \in \mathcal{G}^{ht \times wd \times C}$ görüntüsünde, $\mathcal{G}, \{0, 1, \dots, 255\}$ değer aralığını, ht görüntü yüksekliğini, wd görüntünün genişliğini, $C=3$ ise üç renk kanalını ifade etsin. Semantik segmentasyon, görüntüdeki her bir piksele ($\mathcal{I}_j$), ait olduğu nesneyi veya bölgeyi belirten bir kategori etiketi atar. Bu yöntem, girdi görüntüsü ile çıktı olasılıkları arasında $\mathcal{I}^{seg} \in \mathbb{I}^{ht \times wd \times |Z|}$ tensörü ile temsil edilen, giriş görüntüsü ile çıkış arasında doğrusal olmayan bir eşlemeyi içerir. Burada, olasılık $\mathbb{I} = [0,1]$ ve semantik sınıf kümesi $\mathcal{Z} = \{1, 2, \dots, |Z|\}$ olarak tanımlanır. $\mathcal{I}^{seg}$ içindeki her bir eleman $\mathcal{I}_{j,z}^{seg}$, bir $\mathcal{I}_j$ pikselinin $\mathcal{Z}$ kümesinde belirli bir anlam sınıfı z ’ye ait olma olasılığını temsil eder. Tüm olası sınıflar üzerinde maksimum olasılığı seçerek, her bir piksel $\mathcal{I}_j$ ’ye, bir ayrık sınıf indeksi $m_i \in \mathcal{Z}$ ataması yapılabilir (Klingner et al., 2020).

Her pikseli bir anlamsal sınıflandırmayla etiketlemek ve bu etiketleri analiz etmek, bir bilgisayar görüş modelinin görsel içeriği kapsamlı bir şekilde anlamasını sağlar. Bu süreç, modelin sahnede nesneleri ve bunların uzamsal ilişkilerini tanımasını mümkün kılar. Otonom sürüş için bu yetenek oldukça önemlidir; çünkü otonom araçların sürülebilir alanı, yol çizgilerini, diğer araçları, yayaları ve trafik işaretlerini tanımasına olanak sağlar.



Şekil 3.3 Üst sırada Udacity veri kümesinden rastgele alınmış üç adet orijinal RGB görüntü ve alt sırada bu görüntülerin YOLOPv2 ile elde edilmiş anlamsal bölünmüş görüntüleri

Bu çalışma, monoküler kamera tarafından yakalanan sürüş sahnesine ait görüntüdeki nesneleri tanımlamak ve bunların semantik yapısını anlamak için YOLOPv2 (C. Han et al., 2022)'yi anlamsal sınıflandırma yöntemi olarak kullanmaktadır. YOLOPv2, nesnelerin algılanması, sürülebilir alan segmentasyonu ve şerit tespiti yapabilen çok görevli bir öğrenme modelidir, ancak bu çalışmada nesne tespit görevi dikkate alınmamıştır. Yöntem, özelliklerin çıkartılması için paylaşılan tek bir kodlayıcıdan ve her bir göreve ayrılmış üç ayrı kod çözücünden oluşur. İlk versiyona göre, YOLOPv2'de farklı katmanların ağırlıklarının daha çeşitli özellikler öğrenmesine izin veren ELAN'ın (Verimli Katman Birleştirme Ağları) (C. Wang et al., 2022) tasarımı omurga olarak benimsenmiştir. Modelin boyun bölümü, farklı aşamalardan elde edilen özellikleri bağlayarak entegre eder. YOLOP (D. Wu et al., 2022) ile benzer bir şekilde çok ölçekli özellikleri bir araya getirmek için Uzamsal Piramit Havuzu (Spatial Pyramid Pooling) (SPP) modülü (He et al., 2014) kullanılır. Boyun bölümünden sonra, farklı

semantik seviyelerden özellikleri alan üç ayrı görev başlığı bulunmaktadır. YOLOPv2 modeli, diğer alternatif segmentasyon yöntemlerine kıyasla mevcut literatürdeki en başarılı hız ve doğruluk performansını sergilemesi nedeniyle tercih edilmiştir. Bölümlenmiş görüntü, önerilen modelin üçüncü girdisi olarak kullanılmaktadır. Şekil 3.3'te Udacity veri kümesinden alınmış bazı sürüş sahnelerine ait RGB görüntüler ve bu görüntülere YOLOPv2 uygulandıktan sonra elde edilen semantik bölümlenmiş görüntüler görülmektedir.

3.3 Evrişimsel Özellik Çıkarımı

Görüntü gibi, ızgara yapısında olan verilerin en önemli özelliği uzamsal konumları yakın olan datanın (piksellerin) benzer renk değerlerine sahip olmasıdır. Bu yüzden görüntülerde, komşu piksellere olan ilişkiler ve komşu pikseller arasındaki matematiksel işlemler önemli sonuçlar doğurur. Görüntü içindeki nesnelerin kenarları ve alanları komşu piksellerle olan benzerliklerine veya farklılıklarına göre tespit edilir. Görüntü içinde tek başına anlam ifade etmeyen bir piksel, görüntüdeki komşu pikseller ve bulunduğu konum göz önünde bulundurulduğunda anlam kazanır. Bu yüzden görüntü işleme teknikleri var olduğundan beri komşu piksellerle olan ilişkiler her zaman önemli olmuştur.

3.3.1 Evrişimsel Sinir Ağları (ESA)

Görüntü verilerinde ızgara yapısının karakteristikleri, ızgaranın yerel bölgelerindeki veri birimleri arasında güçlü bağlar oluşturur. Örneğin, bir otonom aracın ön kamerasından alınan yol görüntüsündeki arabayı oluşturan pikseller, çevresindeki piksel değerlerine göre önemli farklılıklar gösterirken, arabayı oluşturan pikseller kendi aralarında yakın değerlere sahiptirler. Benzer şekilde, ağaç gibi nesneleri oluşturan piksellerin renk değerleri birbirine yakındır. Ancak ağacın arka planındaki çevre görüntüsü, farklı renk değerleriyle ağaçtan ayırır. Bu durum, piksellerin konumlarına göre ve komşu piksellerle birlikte değerlendirilmesinin, sahne anlama sürecinde ne kadar önemli olduğunu vurgulamaktadır.

Evrişimsel Sinir Ağları (ESA) (LeCun et al., 1989) ızgara yapısındaki görüntüler üzerinde etkili çalışabilen bir mimari olarak tasarlanmıştır. Bu ağlar, benzer desenlere sahip yerel alanlardan benzer özellik değerleri çıkarma eğilimindedir. LeCun ve diğerleri

tarafından yapılan ilk uygulamalardan biri, el yazısı rakamları tanıma üzerineydi. Bugün ESA'lar, nesne algılama, tanıma, sınıflandırma, görüntü sentezi, segmentasyon gibi pek çok konuda kullanılmaktadır. ESA'ların gelişimi, bilgisayarlı görü çalışmalarında büyük bir değişime ve hatta farklı bilim dallarında ESA'ların kullanımına yol açmıştır.

ESA'ların çalışma prensibi ismini aldığı evrişim (convolution) işlemine dayanmaktadır. Evrişim işlemi, özel bir lineer işlemidir. Bilgisayar görüşü çerçevesinde evrişim, sıradan matris çarpma işlemi yerine, ızgara tipi bir giriş verisi ile yine ızgara tipi bir ağırlık kümesinin nokta-çarpımı işlemidir. ESA'larda, bu işlemde ilk ifade edilen veri “giriş”, ikincisi yani ağırlık kümeleri ise filtre veya “kernel” olarak adlandırılır. Çıkışta elde edilen çarpım ise “özellik haritası” olarak tanımlanır.

Evrişimsel Sinir Ağlarının benimsenmesindeki ana etken, geleneksel yapay sinir ağlarının yüksek boyutlu ve karmaşık görüntü verilerini işlemek için uygun olmamasıdır. ESA'lar, her bir çıktı katmanındaki her nöronun, giriş katmanındaki birkaç nöronla etkileşime girmesine izin vererek hesaplama maliyetini düşürür. Ancak geleneksel ağlarla bu mümkün değildir. Örneğin $32 \times 32 \times 3$ boyutlu bir görüntüden elde edilen $28 \times 28 \times 6$ boyutundaki bir çıkış düşünüldüğünde, girişte 3.072 birim, çıkışta ise 4.704 birim mevcuttur. Bu durumda ağırlık matrisi yaklaşık olarak $3.072 \times 4.704 \approx 14M$ parametreden oluşur. Daha büyük bir görüntü için, parametre sayısı çok daha fazladır. Bu da eğitimi imkânsız hale getirir. Ancak burada örneğin 25 parametreye sahip 5×5 boyutlarında 6 adet filtre kullanarak evrişim işlemi yapılacak olursa, sapma parametresi de dikkate alındığında bir filtre 26 parametreye, 6 adet filtre ise 156 parametreye sahip olur. Görüldüğü gibi parametre sayısı sıradan ağlara göre oldukça düşüktür. Ayrıca ESA'ların sahip olduğu parametre paylaşımı özelliği, ağırlıkların farklı bölgelerdeki filtreler tarafından paylaşılmasını sağlar. Bu, ESA'ların daha az parametre ile daha fazla öznetelik öğrenmesine olanak tanır.

ESA'ların eşdeğer temsil özelliği ise, ağıın öğrendiği temsillerin giriş verisinde yapılan dönüşümlere karşı eşitlik özelliğini ifade eder. Bu, giriş verilerindeki dönüşümlere karşılık, ağıın öğrendiği temsillerin de uygun şekilde değiştiği anlamına gelir. (Aggarwal, 2018; Goodfellow et al., 2016)

Çok sayıda farklı ESA tasarımı olmasına rağmen hepsinin genel olarak bulundurduğu ortak bileşenler vardır. Bir ESA, ardışık evrişim katmanları ve tam bağlı katmanlardan oluşur. Her bir evrişim katmanı içinde, evrişim aşaması, doğrusal olmayan fonksiyon aşaması ve havuzlama katmanları bulunur.

Evrişim aşaması sonucunda elde edilen çıktılar doğrusal olmayan bir aktivasyon fonksiyonundan geçirilir. Aktivasyon fonksiyonları, yapay sinir ağlarının katmanlarında kullanılan matematiksel işlevlerdir. Bu fonksiyonlar, ağın çıktılarını belirlemek için girdi verileri üzerinde bir dönüşüm yaparlar. En önemli işlevleri, ağın öğrenme yeteneğini artırmak ve karmaşık ilişkileri öğrenmesine yardımcı olmaktır. Örneğin, ReLU (Fukushima, 1975), negatif olmayan girdileri doğrudan geçirerek daha hızlı öğrenmeye yardımcı olabilir; Leaky ReLU (Maas et al., 2013) ise negatif girdiler için küçük bir eğim ekleyerek ağın daha esnek olmasını sağlar. Aktivasyon fonksiyonları, ağın çıktılarını daha geniş bir aralıkta ifade etmesine olanak tanır, böylece ağ daha geniş bir problemler yelpazesine uyum sağlayabilir.

Havuzlama katmanları (pooling layers), yapay sinir ağlarının özellik haritalarını küçültmek ve çeşitli öznitelikleri vurgulamak için kullanılan katmanlardır. Genellikle havuzlama işlemi, bir özellik haritasındaki belli bir bölgenin özetini çıkartarak boyutunu azaltır ve önemli öznitelikleri vurgular. ESA’larda en çok tercih edilen havuzlama yöntemi “max pooling” yöntemidir (Zhou and Chellappa, 1988). Bu işlem, seçili dikdörtgen alan içindeki en büyük değeri çıktı olarak verir. Ortalama alan havuzlama fonksiyonu da vardır. Bu fonksiyonlar da seçili alandaki değerlerin ortalamasını çıktı olarak üretir. Havuzlama katmanlarının kullanılmasının temel amacı, özellik haritasındaki boyutu azaltarak hesaplama maliyetini düşürmek ve öğrenme sürecini hızlandırmaktır. Ayrıca, modelin görüntü içindeki küçük konum değişikliklerine karşı daha dayanıklı olmasına ve özniteliklerin translasyon invariansını artırmaya yardımcı olurlar.

ESA mimarilerinde genellikle son kısımda tam bağlı katmanlar bulunur. Bu katmanlar, geleneksel sinir ağlarındaki katmanlar gibidir ve her bir nöron kendisinden sonra gelen katmandaki bütün nöronlara bağlıdır. Evrişim katmanlarından gelen çıktılar genellikle 3D tensörlerdir (genişlik, yükseklik ve derinlik kanalları). Tam bağlı katmanlar, ağın öğrenilmiş özelliklerini kullanarak bu 3D tensörleri düz bir vektöre dönüştürüp sınıflandırma yapmak için kullanılır. Örneğin, bir evrişim katmanının çıkışı $4 \times 4 \times 256$ boyutlarındaysa, bu çıkışlar tam bağlı katmana beslenmeden önce 4096 uzunluğunda 1D vektöre dönüştürülür.

Düzleştirilmiş özellik haritası, genellikle bir veya daha fazla tam bağlı katmana iletilir. Bu katmanlar, ağın öğrendiği özelliklere dayanarak, ya giriş verilerini belirli sınıflara atarlar ya da giriş verisini işleyerek bir değer için tahmin yaparlar. Sınıf atamalarında çıkışta softmax fonksiyonu kullanılırken, tahmin modellerinde genellikle

lineer aktivasyon fonksiyonu kullanılır. Burada sürekli bir çıktı üretilir ve tahmin edilen değer sürekli bir sayıdır.

Bu tez çalışmasının ön işleme aşamasında, RGB görüntüden türetilen derinlik görüntüsü ile anlamsal bölümlenmiş görüntü ve işlem görmemiş RGB görüntü olmak üzere toplam üç görüntüden özellik haritası çıkartılmaktadır. Bu haritaların çıkartılmasında Evrimsel Sinir Ağları (ESA) kullanılmıştır. Bir sonraki bölümde tercih edilen ESA'nın yapısı detaylı şekilde anlatılmıştır.

3.3.2 EfficientNetV2 ağ yapısı

Bu bölümde model için omurga ağ olarak seçilen EfficientNetV2 (Tan and Le, 2021) ağının yapısı ve işleyişi anlatılacaktır. EfficientNetV2, farklı amaçlara yönelik tasarlanmış birden çok versiyona sahiptir. Şekil 3.1'de görüldüğü gibi, bu tez çalışmasında özellik çıkarma yöntemi olarak EfficientNetV2-B0 modeli benimsenmiştir. EfficientNetV2, EfficientNet'in birinci versiyon temel alınarak geliştirilmiş bir Evrimsel Sinir Ağıdır. Eğitim hızını ve doğruluğu artırmak, parametre boyutunu optimize etmek için eğitim odaklı Sinir mimarisi araması (Neural Architecture Search-NAS) ve ölçeklendirmenin bir karışımını kullanır.

EfficientNetV2'nin ana bileşenleri, mobil ters çevrilmiş darboğaz MBConv ve Fused-MBConv (Sandler et al., 2018; Tan et al., 2019)'dur. MBConv bloğu, mobil ve gömülü cihazlar için tasarlanmış yüksek verimli bir ESA bloğudur. Modelin karmaşıklığını azaltırken doğruluğunu koruyan bir darboğaz yapısına sahiptir. Buna ek olarak ikinci versiyonda Fused-MBConv bloğu eklenmiştir. Bu yapı, EfficientNetV2'nin birinci versiyonla arasındaki en önemli farklardan biridir. Fused-MBConv bloğunda, MBConv bloğunda bulunan derinlemesine $\text{Conv}3 \times 3$ ve genişletme $\text{Conv}1 \times 1$ alt birimleri değiştirilmiştir. Tamamı MBConv bloklarından oluşan EfficientNet-B4 modelinde başlangıçtaki bloklar (1-3) Fused-MBConv ile değiştirildiğinde parametreler ve FLOP'lar (Floating Point Operations Per Second- saniye başına yapılan kayan nokta işlemlerinin sayısı) üzerinde küçük bir ek yük ile eğitim hızının arttığı gözlemlenmiştir. Ancak bütün bloklar (1-7) Fused-MBConv ile değiştirildiğinde ise eğitim hızı düşerken parametre ve FLOP'ların da önemli ölçüde arttığı gözlemlenmiştir. Bu nedenle yazarlar, sinirsel mimari araştırma (NAS) yöntemlerini kullanarak bu işlemi otomatikleştirmiş ve en iyi birleşimle modeli oluşturmuşlardır. Elde edilen son modelde, 1-3 arası bloklar Fused-MBConv yapılarından oluşurken, 4-6 arası bloklarda MBConv yapısı tercih

edilmiştir. EfficientNetV2-S'in mimarisini oluşturan alt birimler ve ardışık dizilimleri Tablo 3.1'de detaylı bir şekilde görülmektedir.

EfficientNetV2'de ayrıca, sıkıştırma ve uyarma optimizasyon tekniği kullanılmıştır. Bu teknik, özellik haritalarını önemlerine göre uyarlanabilir bir şekilde yeniden kalibre ederek ESA'ların temsil gücünü artırmaktadır. Fused-MBConv'un yanı sıra, sıkıştırma ve uyarma optimizasyon tekniğinin kullanımı, son derece verimli ve doğru bir model ortaya çıkmasını sağlamıştır.

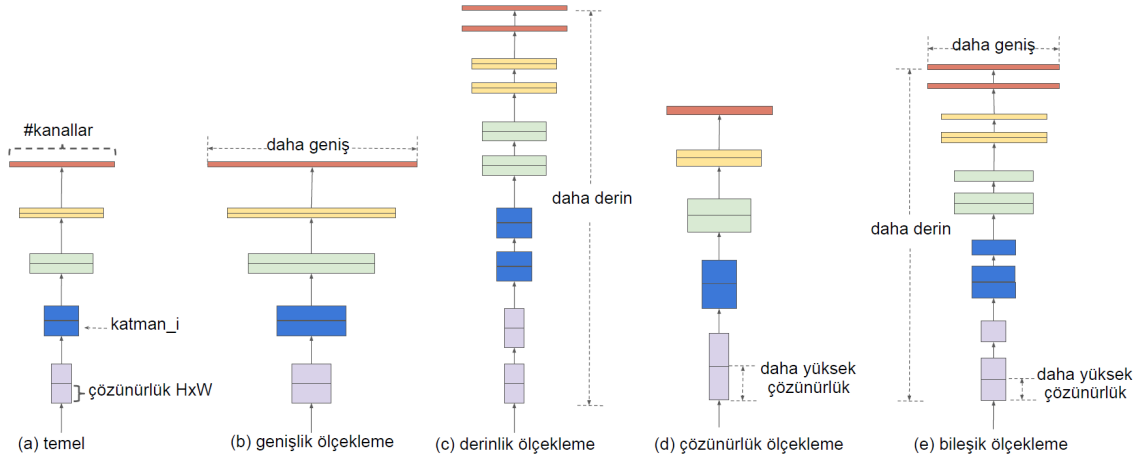
Tablo 3.1 EfficientNetV2-S'in mimarisi

Kademe	Operatör	Filtre adımı (stride)	Kanal sayısı	Katman
0	Conv3×3	2	24	1
1	Fused-MBConv1, k3×3	1	24	2
2	Fused-MBConv4, k3×3	2	48	4
3	Fused-MBConv4, k3×3	2	64	4
4	MBConv4, k3×3, SE0.25	2	128	6
5	MBConv6, k3×3, SE0.25	1	160	9
6	MBConv6, k3×3, SE0.25	2	256	15
7	Conv1x1 & Pooling & FC	-	1280	1

EfficientNetV2'de ikinci değişiklik olarak bellek yükünü azaltmak adına MBConv için daha küçük genişleme oranı tercih edilmiştir. Bu sayede hem eğitim süresi hem de parametre ve FLOPs değerleri düşük olan ve daha kısa sürede çıkarım yapabilen bir model geliştirilmiştir. Bu nedenle, otonom araçlar gibi, mobil ve gömülü cihazlarda yüksek performans gerektiren sistemler için oldukça uygun bir seçenektir.

Son yıllarda hesaplama gücünün artması nedeniyle, yalnızca doğruluğu arttırmaya yönelik yüksek sayıda parametre içeren veya aşırı uzun eğitim süresine sahip çalışmalar ortaya çıkmıştır. Bu durum, büyük model boyutlarının neden olduğu zorlukları ve eğitim süreçlerinin karmaşıklığını beraberinde getirmiştir. EfficientNetV2, bu bağlamda yalnızca doğruluğu artırmakla kalmayıp, aynı zamanda bunu daha az parametre kullanarak ve daha hızlı bir eğitim süreciyle başarmasından dolayı diğer modellere üstünlük sağlamaktadır. Ayrıca eğitim sırasında eğitim verimliliğini artırmak için düzenlemeyi ve görüntü boyutunu dinamik olarak ayarlamak üzere geliştirilmiş bir aşamalı öğrenme yaklaşımını benimser. Bu durum, EfficientNetV2'nin sadece sonuçları değil, aynı zamanda eğitim sürecindeki verimliliği de artırmaya odaklandığını göstermektedir.

EfficientNetV2, birleşik ölçeklendirme yoluyla elde edilmiş çeşitli boyutlarda modeller sunar. Bu ölçeklendirme tekniği, genişlik, derinlik ve çözünürlük gibi ağ boyutlarını sabit bir oran kullanarak ayarlar. Bu modellerin boyutları, isimlendirme olarak B0'dan B7'ye ve S, M, L, XL gibi harf kodlamalarına sahiptir. İngilizce temelleri olan bu harfler, boyutları temsil eder: Baseline (B), Small (S), Medium (M), Large (L), Xlarge (XL). EfficientNetV2 modellerinin doğruluğu model boyutlarıyla ilişkilidir. Daha fazla parametreye sahip olan daha büyük modeller, genellikle daha yüksek doğruluk sağlarlar; ancak daha uzun süre eğitim gerektirirler. Buna karşılık, daha küçük modeller daha az parametreye sahip oldukları için daha kısa sürede eğitilirler, ancak daha düşük bir doğruluk seviyesine sahip olabilirler. EfficientNet'in model ölçeklemesi Şekil 3.4'de detaylı bir şekilde görülmektedir. Burada (a)'da basit bir ESA yapısı görülürken, (b)-(d) aralığında sırasıyla genişlik, derinlik ve çözünürlüğün artırıldığı ölçeklemeler görülmektedir. (e) ise üç boyutu sabit bir oranda eşit şekilde ölçeklendiren bileşik ölçeklendirmedir.



Şekil 3.4 EfficientNet'in model ölçeklemesi. a) temel ağ örneği; (b)-(d), ağ boyutlarından genişliğin, derinliğin, veya çözünürlüğün yalnızca birini artıran geleneksel ölçeklendirme (e) üç boyutun tamamını sabit bir oranda eşit şekilde ölçeklendiren bileşik ölçeklendirme yöntemi (Tan and Le, 2019)

Parametre verimliliğini korumak ve eğitim süresinin kısa olmasını sağlamak için bu çalışmada EfficientNetV2-B0 modeli benimsenmiştir. Bu seçim tüm modellerin düşük parametre sayısına sahip olmasını sağlar. Tablo 3.2, EfficientNetV2'nin ImageNet (Russakovsky et al., 2015) üzerinde diğer çalışmalarla kıyaslamasını sunmaktadır. Çalışmada tercih edilen model, gri ile vurgulanmıştır. B0 modeli, 7,4 milyon parametre ile ImageNet (Jia Deng et al., 2009) Top-1'de %78,7 doğruluğa sahiptir. Bu düşük

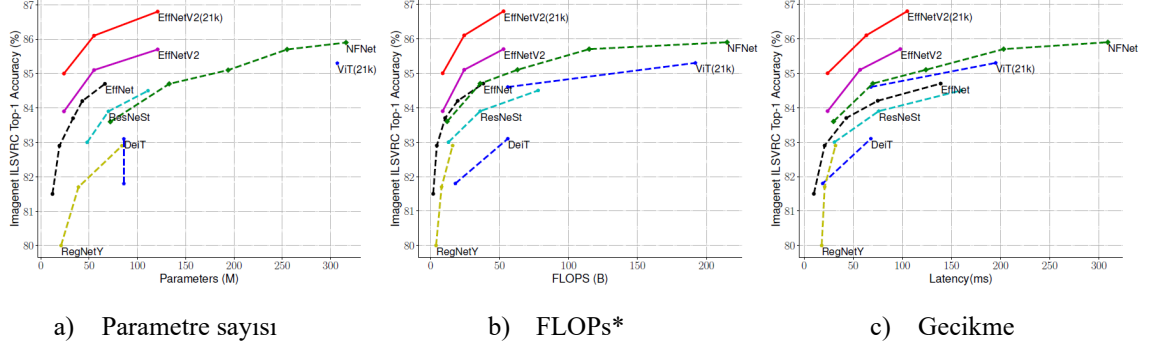
parametre sayısı ile elde edilen doğruluk oranı, literatürdeki en başarılı sonuçlardan biridir. Örneğin ResNetSt-269 modeli %84,5 doğruluğa sahipken, bu sonucu 111 milyon parametre ile elde etmiştir. EfficientNetV2'nin kendi içinde en büyük modeli olan XL modeli, 208 milyon parametre ile %87,3 doğruluk oranı elde etmiştir. Ancak, bu büyüklükte bir modelin eğitimi oldukça zordur. Bu nedenle, tercih edilecek ağ otonom araçlarda kullanılacağı düşünüldüğünde, mobil ve gömülü sistemlerde rahatlıkla kullanılabilmesi için yüksek parametrelili bir model yerine düşük parametrelili ancak yüksek verimli bir model tercih edilmiştir.

Tablo 3.2 EfficientNetV2'nin ImageNet üzerinde performans karşılaştırması (Tan and Le, 2021)

Model	Top-1 Doğruluk	Param.	FLOPs	Çıkarım süresi (ms)	Eğitim süresi (saat)
EfficientNet-B3 (Tan and Le, 2019)	%81,5	12M	1.9B	19	10
EfficientNet-B7 (Tan and Le, 2019)	%84,7	66M	38B	170	139
ResNetSt-269 (H. Zhang et al., 2022)	%84,5	111M	78B	160	-
NFNet-F4 (Brock et al., 2021)	%84,7	316M	215B	309	126
ViT-B/16 (21k) (Dosovitskiy et al., 2020)	%84,6	87M	56B	68	-
EfficientNetV2-B0 (Tan and Le, 2021)	%78,7	7,4M	0,7B	-	-
EfficientNetV2-S (Tan and Le, 2021)	%83,9	22M	8,8B	24	7,1
EfficientNetV2-L (Tan and Le, 2021)	%85,7	120M	53B	98	24
EfficientNetV2-XL(21k) (Tan and Le, 2021)	%87,3	208M	94B	-	45

Burada "Top-1 Doğruluk", modelin ImageNet test kümesinde en yüksek olasılığa sahip sınıf tahmininin, görüntülerin gerçek etiketleriyle tam olarak eşleştiği doğruluk yüzdesini ifade eder. "Param." ifadesi parametre sayısını, FLOPs (Floating Point Operations Per Second) ifadesi ise saniye başına yapılan kayan nokta işlemlerinin sayısını temsil eder

Şekil 3.5'te EfficientNetV2'nin farklı özellikler için literatürdeki diğer çalışmalarla kıyaslaması görülmektedir. Şekil 3.5a'da açıkça görüldüğü gibi EfficientNetV2 parametre sayısı bakımından diğer çalışmalara göre oldukça üstündür. Daha düşük parametre ile daha yüksek ImageNet Top-1 doğruluğu elde edebilmektedir. Şekil 3.5b'de ise FLOPs değerine göre karşılaştırma yapılmıştır. Burada da EfficientNetV2 çok daha düşük işlem sayısı ile rakiplerinden daha yüksek doğruluk değeri elde etmiştir. Şekil 3.5c'de, gecikme süreleri bakımından çalışmaları kıyaslamıştır. Burada, EfficientNetV2'nin süre bakımından da diğer çalışmalara göre oldukça avantajlı olduğu görülmektedir.



Şekil 3.5 EfficientNetV2'nin literatürdeki diğer çalışmalarla parametre sayısı, FLOPs ve gecikme süresine göre kıyaslanması (Tan and Le, 2021)

Evrişimsel sinir ağlarının önemli yapılarından bir tanesi de havuzlama yapılarıdır. Model, evrişimsel sinir ağının derin katmanlarından elde edilen karmaşık özellik haritasını vektörel bir forma dönüştürmek amacıyla Global Ortalama Havuzlama (Global Average Pooling) (Lin et al., 2014) tekniğini kullanır. Bu yaklaşım, modelin eğitim verilerini aşırı derecede ezberlemesini engelleyerek, yeni ve görünmeyen verilere daha etkili bir şekilde genelleme yapabilmesine yardımcı olur.

Aşırı uyum, bir modelin eğitim verilerine aşırı derecede uyarlanması sonucu ortaya çıkar ve bu durum genellikle modelin yeni ve farklı verilere iyi bir şekilde adapte olmasını engeller. Global Ortalama Havuzlama, özellik haritasını daha düşük boyutlu bir vektör haline getirerek, bilgi kaybını minimumda tutar. Bu, modelin öğrenme sürecinde gürültü veya gereksiz detayları göz ardı ederek daha temel ve genelleme yeteneği yüksek özelliklere odaklanmasını sağlar. Böylelikle, model eğitim verilerine aşırı derecede uyarlanmadan önce daha iyi bir öğrenme yeteneği geliştirir ve daha geniş bir veri kümesinde başarılı performans gösterebilir.

Model içinde bir de atma (Dropout) yöntemi kullanılmıştır. Atma (düşürme) yöntemi, eğitim sırasında ağıdaki bazı nöronları rastgele devre dışı bırakarak ağı düzenlemenin etkili bir yolunu sağlar. Bu yöntem, kalan nöronların daha sağlam özellikler öğrenmesini zorlar ve aşırı uyumu azaltır. Atma yöntemi, hangi nöronların kapatılacağını belirlemek için Bernoulli dağılımını kullanır.

Bağlama bölümünde ESA'lardan gelen üst düzey özellikler farklı boyutlarda birleştirilir. Derinlik görüntüsünün ve anlamsal bölümlenmiş görüntünün özellik vektörleri eşit boyutta olmasına rağmen, RGB görüntüsünden türetilen özellik vektörü, Şekil 3.1'de gösterildiği gibi, bunların iki katı boyutundadır.

3.4 Zamansal Özellik Çıkarımı

Otonom sürüş, sürekli akış halindeki sensör verileri ve aracın bir yol boyunca hareketi sırasında oluşan ardışık kontrol komutlarının oluşturduğu veri yapılarına dayanır. Örneğin, bir aracın köşeyi dönmeye yaklaştığı zamandan başlayıp, dönüşü tamamlayana kadar aldığı sensör verilerine bakıldığında, dönüş esnasında mevcut t zamanından önceki belirli sayıdaki (örneğin $t-20$) veri, o anki durum hakkında aracın yorum yapmasına yardımcı olacak değerli ipuçları sunar. Bu durum, sadece anlık verilere göre değil, aynı zamanda yakın geçmişe dayalı bilgilere de dayanarak nihai kararın oluşturulmasına yardımcı olur. Bu sayede sistem daha istikrarlı tahminler yapabilir.

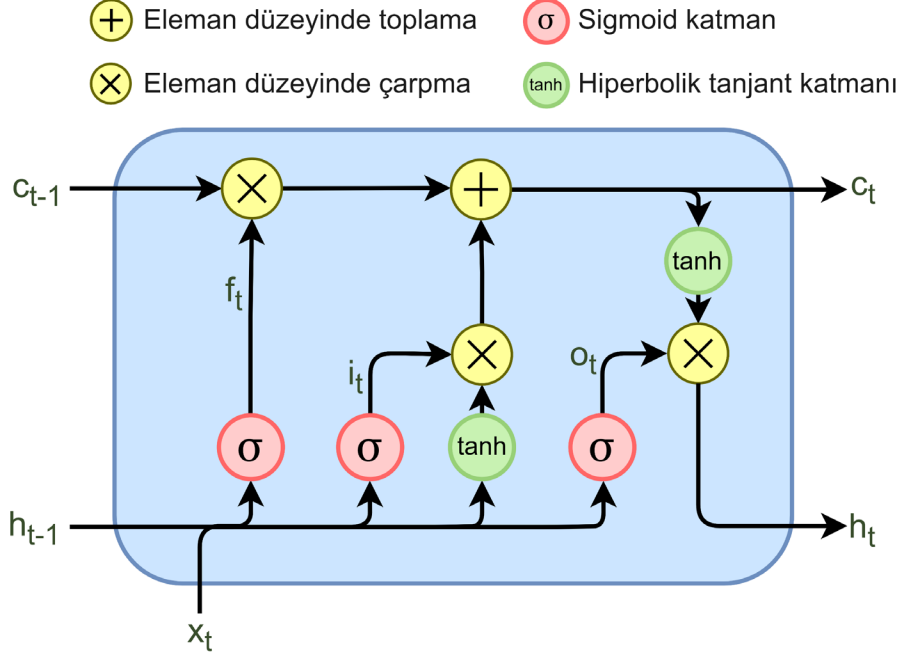
Bu çalışmada, ardışık veri bağımlılıklarından faydalanmak için Uzun Kısa Vadeli Bellek (Long Short-Term Memory) (LSTM) (Hochreiter and Schmidhuber, 1997) birimi tercih edilmiştir. Geçitli Tekrarlayan Birim (Gated Recurrent Unit) (GRU) (Cho et al., 2014) uygun bir seçenek olarak değerlendirilmiş olmasına rağmen, sonuçlar LSTM birimi kullanılarak elde edilenlerden önemli ölçüde farklılık göstermemiştir. Bu nedenle, başarılı performansı ve çeşitli ardışık öğrenme görevleri için uygunluğu nedeniyle LSTM birimi tercih edilmiştir.

LSTM birimleri, bir tekrarlayan sinir ağı (Recurrent Neural Network) (RNN) mimarisi olup, kaybolan gradyan problemlerinde ve uzun vadeli bağımlılıkları yakalama zorluğu olan durumlarda kullanılır. LSTM birimi, bir hücre durumu (c) ve üç uyarlanabilir kapı birimini içerir: giriş kapısı (i), unutma kapısı (f) ve çıkış kapısı (o). LSTM modülünün çalışmasını gösteren şema Şekil 3.6'da verilmiştir.

Giriş kapısı (i), ağa yeni bilginin eklenmesini düzenlerken, unutma kapısı (f) önceki bilgilerin ne kadarının korunacağını belirler ve gereksiz veya eski bilgileri silerek hücre durumunu günceller. Bu sayede hücre durumu, önemli ve güncel bilgileri muhafaza eder. Diğer yandan, çıkış kapısı (o), o anki durumun ağ çıkışını verir. Hücre durumu (c), ağın içindeki uzun vadeli belleği temsil ederken, gizli durum (h), ağın anlık çıkışını yansıtır. Bu şekilde, LSTM birimleri, RNN mimarisinin temel zorluklarını aşarak uzun vadeli bağımlılıkları yakalama kabiliyetleriyle dikkat çeker.

LSTM ağı, her bir t zaman adımında, bir giriş vektörü ($\mathbf{x}_t$)'yi alır. Giriş kapısı, güncellenecek değerleri belirlemek için bir sigmoid fonksiyonu kullanır. Giriş vektörü $\mathbf{x}_t$, hiperbolik tanjant fonksiyonu ($\tanh$) kullanarak bir aday durum $\tilde{\mathbf{c}}_t$ oluşturmak için önceki gizli durum $\mathbf{h}_{t-1}$ ile birleştirilir, bu durum yeni hücre durumu $\mathbf{c}_t$ içine dahil edilebilir. Unutma kapısı (f), eski hücre durumu $\mathbf{c}_{t-1}$ 'den gereksiz bilgileri seçerek kaldırır. Çıkış

kapısı (σ), hücre durumu c_t 'nin hangi öğelerinin gizli durum h_t 'ye geçeceğini belirler. Yeni gizli durum daha sonra ağırlı bir sonraki zaman adımına beslenir ve LSTM'nin iç durumunu güncellemeye devam etmesini sağlar.



Şekil 3.6 LSTM biriminin iç yapısı

Son olarak, LSTM eski hücre durumu c_{t-1} 'i yeni hücre durumu c_t 'ye çarpma ve toplama işlemleriyle güncellerken bilgiyi seçici bir şekilde saklar veya unuttur. Bu yetenek, LSTM'nin uzun vadeli bağımlılıkları etkili bir şekilde yakalamasını sağlar. LSTM birimi, t zaman adımında, hücre durumunu aşağıdaki denklemlere göre günceller:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \quad (3.4)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \quad (3.5)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \quad (3.6)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (3.7)$$

$$h_t = o_t \circ \tanh(c_t) \quad (3.8)$$

Denklem (3.8)'de, $\circ$ Hadamard çarpımını, denklem (3.4)-(3.8) arasında W , ilgili ağırlık matrislerini, b , sapma vektörünü, $\sigma(x) = (1 + e^{-x})^{-1}$ sigmoid fonksiyonu ve $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ hiperbolik tanjant fonksiyonunu simgeler.

Bu çalışmada, 128 hücreli bir LSTM bloğu kullanılmıştır. Bu LSTM bloğu son on direksiyon açısını ve hız değerini giriş olarak almakta ve yukarıda bahsedilen denklemler uyarınca zamansal özellik vektörü üretmektedir. Aynı zamanda, LSTM içinde tekrarlayan durumların doğrusal dönüşümü için düşecek birimlerin oranı, yani “recurrent_dropout” değeri %10 olarak belirlenmiştir. Bu düşük "recurrent_dropout" oranı, geçmiş bilgiyi hatırlama kapasitesini korurken aynı zamanda ağırlık genelleme yeteneğini artırır. Bu sayede, model, direksiyon açıları ve hız değerleri arasındaki kompleks ilişkileri daha başarılı bir şekilde öğrenir, özellikle zaman içindeki değişen desenlere daha duyarlı hale gelir. Bu parametre seçimi, modelin performansını optimize etme ve istenilen çıkışı üretme kapasitesini artırma amacını taşır.

3.5 Uçtan Uca Komut Tahmini

Bu çalışmada, ESA'lardan gelen uzamsal özellikleri ve LSTM'den gelen zamansal özellikleri birleştirmek için ağırlıklı birleştirme yaklaşımı kullanılmıştır. Her biri farklı veri modellerini temsil eden üç ESA ve bir LSTM tarafından üretilen yüksek düzeyli özellik vektörleri, birleştirme aşamasından önce özel katsayılar kullanılarak ağırlıklı hale getirilmiştir. ESA ağırlıklarından ve LSTM biriminden elde edilen üst düzey özellik vektörleri sırasıyla $hl^{(1)}$, $hl^{(2)}$, $hl^{(3)}$, $hl^{(4)}$ 'tür. Her bir vektör, bireysel ağırlık katsayıları $w^{(1)}$, $w^{(2)}$, $w^{(3)}$, $w^{(4)}$ ile çarpılarak ağırlıklandırılmış vektörler elde edilmiştir. Bu ağırlıklandırılmış vektörler daha sonra bağlama/ birleştirme (concatenation) işlemine tabi tutulmuştur (denklem (3.9)):

$$z = [w^{(1)}hl^{(1)}|w^{(2)}hl^{(2)}|w^{(3)}hl^{(3)}|w^{(4)}hl^{(4)}] \quad (3.9)$$

Ardından, bağlanmış ağırlıklı yapı, iki tam bağlı yoğun katman aracılığıyla işlenmiştir. Yoğun (tam bağlı) katmanlar bölümü için farklı konfigürasyonlarda katman bağlantıları denenmiştir. Bu yapılandırmaların içerdiği katmanlar, özelliklerin daha iyi bir şekilde işlenmesini ve temsil edilmesini sağlayacak şekilde optimize edilmiştir. Denemeler sonucunda, birleştirme işleminden sonra gelen 64 tam bağlantılı katmanın, iki tam bağlantılı katmana bağlandığı durum, en başarılı sonuçları vermiştir (*bağlama/64/2*). Daha sonra bu son katmanın bileşenlerinin birinden direksiyon açısı tahmini, diğerinden de hız tahmini yapılmıştır. Yoğun katmanlar bölümünde farklı konfigürasyonlar da

denenmiş (128/64/32/16/1, 128/64/1, 64/32/1 vb.), ancak hiç birisi (bağlama/64/2) konfigürasyonundan daha iyi sonuç üretmemiştir.

Son aşamada, ağın çıkışı olarak direksiyon açısı ve hız değeri, ortak katmandan elde edilmektedir. Bu, ağın derinliği ve karmaşıklığına rağmen, orijinal verilerin anlamının başarılı bir şekilde çıkarılmasında etkili olduğunu göstermektedir. Derin öğrenmenin birçok aşamasında olduğu gibi burada da genel bir doğru yoktur. Burada her bir katmandaki nöron sayılarının seçimi, modelin veriden karmaşık örüntüleri öğrenme kapasitesini etkileyebilir. Çok katman veya çok nöron kullanmak her zaman daha çok şey öğrenilebileceği anlamına gelmez. Daha fazla nöron kullanmak, modelin verilerdeki daha karmaşık ilişkileri yakalamasına olanak sağlayabilir; ancak aynı zamanda, modelin belirli bir veri kümesi için gereğinden fazla karmaşık hale gelmesi durumunda aşırı uyuma da yol açabilir. Tam bağlı katmanların konfigürasyonu genellikle deneme ve ayarlama süreçleriyle belirlenir. Regresyon, ikili veya çok sınıflı sınıflandırma gibi belirli görevlerde en iyi performansı elde etmek için, model karmaşıklığı ile genelleme arasında bir denge kurulması gerekir. Bu seçim süreci genellikle veri kümesi boyutunu, karmaşıklığını ve mevcut hesaplama kaynaklarını dikkate almayı gerektirir. Bu çalışmada, yoğun katmanların konfigürasyonu, çeşitli denemeler neticesinde belirlenmiştir.

4. VERİ KÜMELERİ VE EĞİTİM VERİLERİNİN HAZIRLANMASI

Tasarlanan derin öğrenme modelleri, doğrulama amacıyla veri kümeleri üzerinde eğitilip test edilirler. Bu tez kapsamında, öncelikle Udacity veri kümesi (Udacity, 2017) tercih edilmiştir; bu veri kümesi, gerçek yollarda insan sürücünün kullandığı araç ile toplanan gerçek sensör ve kontrol verilerini içermektedir. Modelin tek bir veri kümesine özgü olmadığını doğrulamak ve genellenebilirliğini göstermek amacıyla, geliştirilen model Udacity veri kümesinin yanı sıra iki farklı veri kümesinde daha test edilmiştir. Bunlardan biri, CARLA Otonom Sürüş Simülatörü kullanılarak bu tez çalışması için oluşturulmuştur ve dolayısıyla sentetik bir veri setidir. Modelin test edildiği bir diğer veri kümesi de yine gerçek yol koşullarında insan sürücüler tarafından toplanmış Sully Chen veri kümesidir (Sully Chen, 2023). Takip eden alt bölümlerde, üç veri kümesi de ayrıntılı olarak anlatılacaktır.

4.1 Udacity Veri Kümesi

Otonom sürüşte, ego-hareket, segmentasyon, 3D sahne anlama, hareket kestirimi ve planlama gibi uygulamalar için birçok veri kümesi bulunmasına rağmen, direksiyon açısı sunan veri setlerinin sayısı oldukça kısıtlıdır. Bu çalışma kapsamında tercih edilen veri kümelerinden biri, Udacity veri kümesi olmuştur ve tercih sebepleri çeşitli nedenlere dayanmaktadır. Udacity veri kümesi, otonom araçlar için karşılaşılan zorlukları temsil eden unsurları içererek dikkat çeker. Ayrıca, günün farklı saatlerinde toplanmış kamera görüntüleriyle öne çıkar. Bu durum, sahnede koyu gölgelerin veya aşırı parlak alanların oluşmasına neden olur ve bu da otonom sürüş için gerçek dünya senaryolarını temsil etme açısından önemlidir. Veri kümesi, yoğun trafik içeren sahnelerden, keskin virajlar içeren farklı yol tiplerine kadar geniş bir yelpazede veri sunar. Ayrıca, bazı keskin virajlarda yolun görüntüde neredeyse görünmez olduğu zorlu durumlar da bu veri kümesinde yer almaktadır.

Udacity veri kümesi başlangıçta bir çevrimiçi yarışma için oluşturulmuştur ve içerisinde aracın üç farklı yöne (sol ön- merkez- sağ ön) bakan kameralarıyla kaydedilmiş video akışları bulunur. Bu kameralarla 20fps hızında kayıt alınmıştır. Ayrıca, aracın hızı, direksiyon açısı ve tork gibi değerler de kamera görüntüleri ile eş zamanlı olarak, Kontrol Alanı Ağı (Controller Area Network) (CAN bus) üzerinden toplanmıştır. Veri kümesinin

paylaşım sayfasında bugün itibariyle, ikinci ve üçüncü yarışma için sunulan veri setleri (CH2 ve CH3) bulunmaktadır.

Şekil 4.1’de Udacity veri kümesinden alınmış sürüşün farklı sahnelerine ait zorlu görüntüler görünmektedir. Bu görüntülerin içinde sürüş yüzeyinin çok parlak olduğu sahnelerde (Şekil 4.1a) yol ayırt edilememektedir. Bazı durumlarda da yol yüzeyine düşen parçalı gölgeler (Şekil 4.1b) yanlış tespite neden olabilecek potansiyel durumlardır. Günün farklı saatlerinde veri toplanmasından dolayı güneşin az olduğu zamanlarda yol yüzeyinin farklı renklerde görüldüğü görüntüler de kaydedilmiştir (Şekil 4.1c). Bazı sahnelerde ise hem koyu gölgeler hem de parlak yüzeyler aynı anda bulunabilmektedir (Şekil 4.1d). Görüldüğü gibi Udacity veri kümesi algılama ve tespit açısından oldukça zorlayıcı görüntüler içermektedir. Bu da denenen yöntemin böyle bir veri kümesi ile eğitildikten sonra farklı aydınlanma koşullarından etkilenmeden sağlıklı algılama yapmasını sağlamaktadır.



a) Parlak sürüş yüzeyi



b) Parçalı gölgeli sürüş yüzeyi



c) Farklı renkte görünen sürüş yüzeyi



d) Koyu gölgeli sürüş yüzeyi

Şekil 4.1 Udacity veri kümesinden alınmış sürüşün farklı sahnelerine ait zorlu görüntüler: a) Sürüş yüzeyi aşırı aydınlanma nedeniyle çok parlak olduğu için ayırt edilememektedir. b) Yol yüzeyindeki parçalı gölgeler algılamayı zorlaştırır. c) Sürüş yüzeyi yine aydınlanma saatine bağlı olarak sıradan asfalt rengi yerine mavimsi bir renkte görülmektedir. d) Sürüş yüzeyinde hem çok parlak alanlar hem de koyu gölgeler mevcuttur

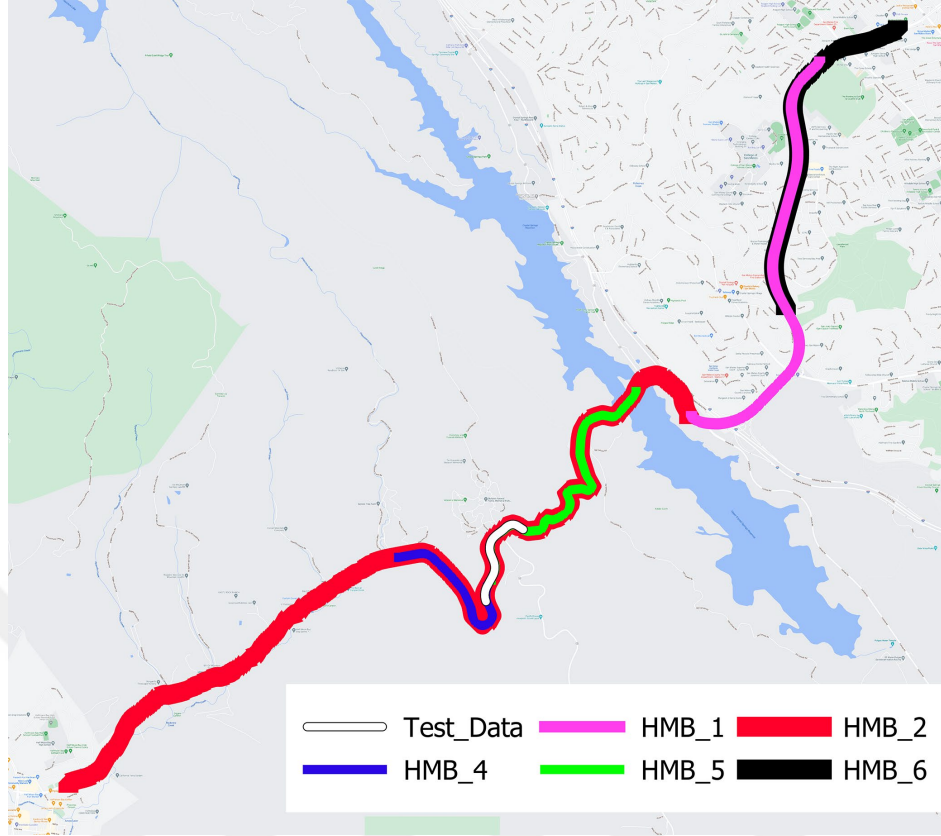
Tablo 4.1 Udacity sürüş datası ve açıklamaları

İsim	Amaç
CH2_001	Yarışma 2 için test verileri*
CH2_002	Yarışma 2 için eğitim verisi
CH3_001	IMU konumlandırılmalı El Camino verisi
CH3_002	IMU ve LiDAR verili El Camino verisi

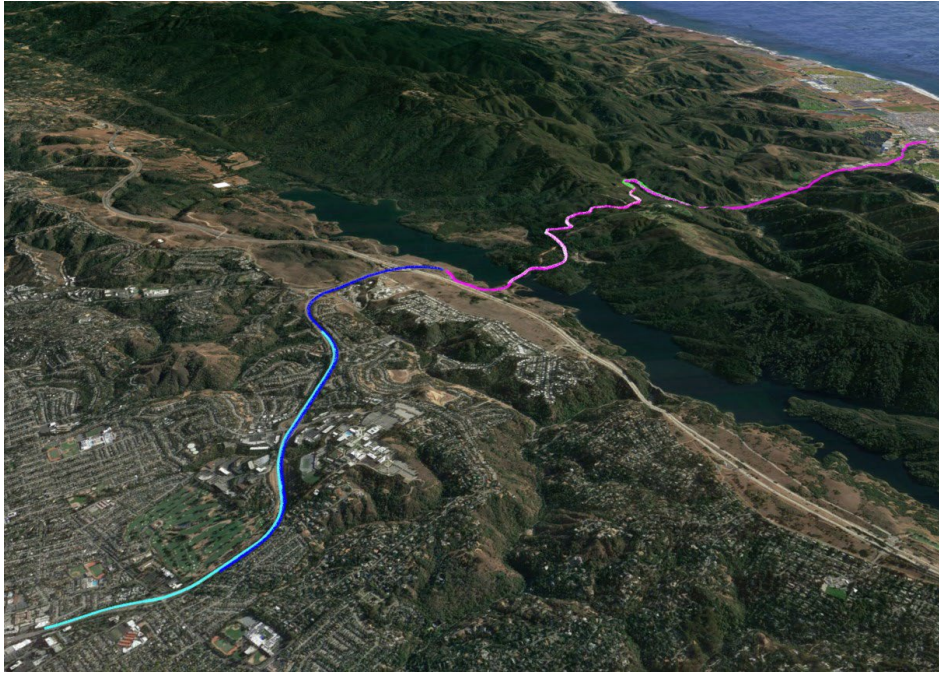
* Yarışmacıların erişimine açık olmadığı için CH2'nin test verisi kullanılamamaktadır

İki yarışmada sunulan verilerin içerikleri Tablo 4.1'de görüldüğü gibidir. Yarışmacıların test verisini görmemesi gerektiği için CH2'nin yarışma için kullanılan test verisi erişime açık değildir. Bu çalışmada Udacity veri kümesinin CH2_002 parçası kullanılmıştır. Düzenleyici, aracın hareket esnasındaki enlem ve boylam bilgilerini de sağladığı için CH2_002 veri kümesinin her bölümünde aracın geçtiği yolları Şekil 4.2'de gösterildiği gibi harita üzerine yansıtılmıştır. Bu yolların arazi üzerinde üç boyutlu olarak gösterimi de Şekil 4.3'de verilmiştir. Burada daha açık görüldüğü üzere veri engebeli ve virajlı bir arazide toplanmıştır. Aynı zamanda hem düz yolların olduğu hem de virajlı yolların olduğu bölümler vardır. Bu şekilde veri çeşitliliği, öğrenme sürecinde modelin daha verimli öğrenmesini sağlamaktadır. Veri kümesinin bu kısmı beş parçadan oluşmaktadır ve her bir renk farklı bir video dizisini temsil etmektedir. Beyaz kısım test verisi için seçilmiş kısmı göstermektedir. Udacity ile daha önce yapılan çalışmalarda buradaki gibi bir harita gösterimi hiç yapılmamıştır. Bu haritanın oluşturulması ve video parçalarının yol üzerinde gösterimi sonucunda, Şekil 4.2'de görüldüğü gibi bazı parçaların birbiri ile örtüştüğü fark edilmiştir. Eğitim kümesi içinde tekrar eden verilerin bulunması, öğrenmeyi olumsuz etkileyeceği için bu durum hemen dikkat çekmiştir. Ancak, görüntüler incelendiğinde, bu kısımların günün farklı zaman aralıklarında ve farklı yönlerde seyahat ederken kaydedildiği doğrulanmıştır. Dolayısıyla eğitim kümesi içinde birden fazla aynı görüntüden bulunmamaktadır.

Özellikle eğitim sürecinde tekrarlanan veya aynı sahnelerin birden fazla kez görüntülenmesi, modelin yanıltıcı verilerle eğitilmesine yol açabilir. Bu nedenle, veri kümesindeki benzer veya örtüşen bölgelerin tespiti ve bunların eğitim verilerinden temizlenmesi, modelin daha güvenilir ve doğru bir şekilde öğrenmesini sağlamak için önemlidir. Mevcut durumda ise aynı konumda farklı saatlerde ve farklı yönlerde toplanan görüntüler tam aksine veriye çeşitlilik katmaktadır. Zaten hem literatürde çokça bulunduğu gibi hem de bu çalışmada da uygulandığı üzere veri artırma teknikleri ile eğitim verisinin çoğaltılması öğrenmenin performansını artıran bir yöntemdir. Veri artırma ile ilgili ayrıntılar sonraki bölümde (Bölüm 4.1.1.2) detaylı olarak anlatılmıştır.



Şekil 4.2 Udacity veri kümesine ait yol haritası. Bu harita, veri kümesinin beş ayrı bölümünü temsil etmekte olup, her renk farklı bir video serisini belirtmektedir. Görseldeki beyaz bölge ise test verisi olarak seçilen kısmı göstermektedir



Şekil 4.3 Udacity veri kümesinin arazi üzerinde elde edilen üç boyutlu görüntüsü. Toplanan veri kümesi hem düz yolları hem de virajlı bölgeleri içermektedir ve engebeli bir arazide elde edilmiştir

CH2_002 bölümü, 640×480 çözünürlüğe sahip altı video içermektedir; bunlardan beşi eğitim için, biri test için oluşturulmuştur. Ancak, test bölümünde hız verisi bulunmamakta, sadece direksiyon açısı sunulmaktadır. Önerilen model hem direksiyon açısı hem de hızı tahmin ettiğinden, bu bölüm test verisi olarak kullanılamamıştır. Bu durumda, verilen eğitim kümesi, yeni bir eğitim kümesi ve bir test kümesi içerecek şekilde yeniden düzenlenmiştir. Beş parçadan oluşan orijinal eğitim verisi, %90'ı eğitim için kullanılmak üzere bölünmüş ve %10'u test verisi olarak ayrılmıştır.

Bu düzenleme ile, modelin doğruluğunu test etmek için ayrılmış özgün bir veri kümesi oluşturulmuştur. Test kümesi, modelin genellenebilirliğini ve performansını değerlendirmek amacıyla eğitim kümesinden ayrılmıştır. Bu şekilde, modelin gerçek dünya koşullarında ne kadar iyi performans gösterdiği daha sağlıklı bir şekilde değerlendirilmiştir.

Test verisi, tamamen ayrı bir set olarak tutulmuş ve eğitim süresince öğrenmeye dahil edilmemiştir. Eğitim ve doğrulama için 30,423 (%90) kare kullanılırken, test için 3,380 (%10) kare ayrılmıştır. Eğitim verisini genişletmek amacıyla çeşitli veri artırma teknikleri uygulanmış ve bunun sonucunda 53,220 kare içeren genişletilmiş bir eğitim veri kümesi elde edilmiştir. Eğitim sürecinde, artırılmış karelerin %20'si (10,644 kare) doğrulama için kullanılırken, geriye kalan 42,576 kare (%80) ise eğitim amaçlı kullanılmıştır.

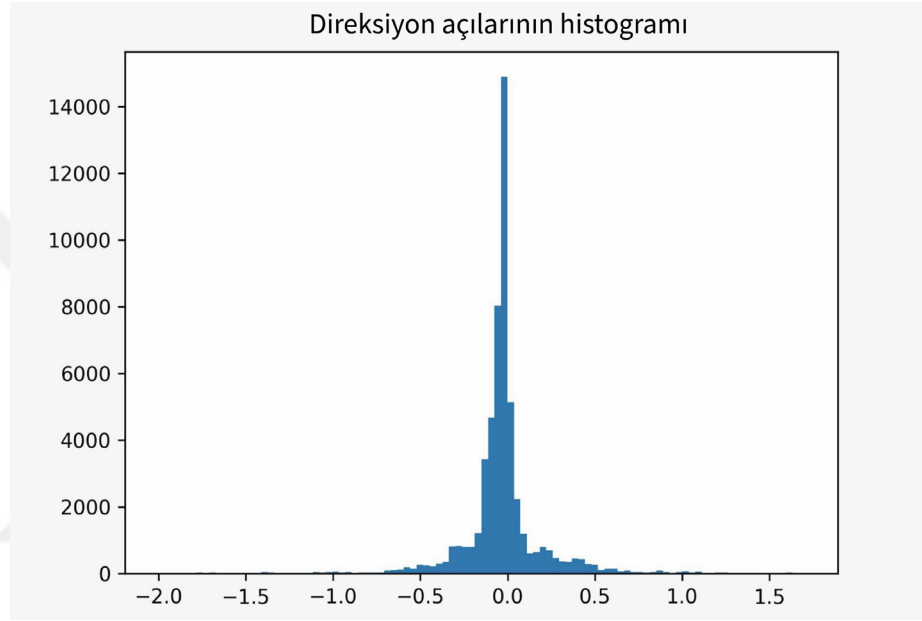
Uygulanan veri artırma teknikleri, Bölüm 4.1.1.2'de detaylı bir şekilde ele alınmıştır. Veri artırma teknikleri, eğitim verisinin çeşitliliğini artırmak ve modelin genelleme yeteneğini geliştirmek amacıyla kullanılmıştır. Bu tekniklerin detaylı olarak incelenmesi, modelin eğitim sürecinde ne tür dönüşümlerden geçtiğini ve nasıl geliştirildiğini daha iyi anlamamıza yardımcı olmuştur. Bu da modelin daha geniş bir veri kümesinde ne kadar iyi performans gösterebileceğini değerlendirmemize imkân sağlamıştır.

4.1.1 Udacity eğitim verisinin hazırlanması

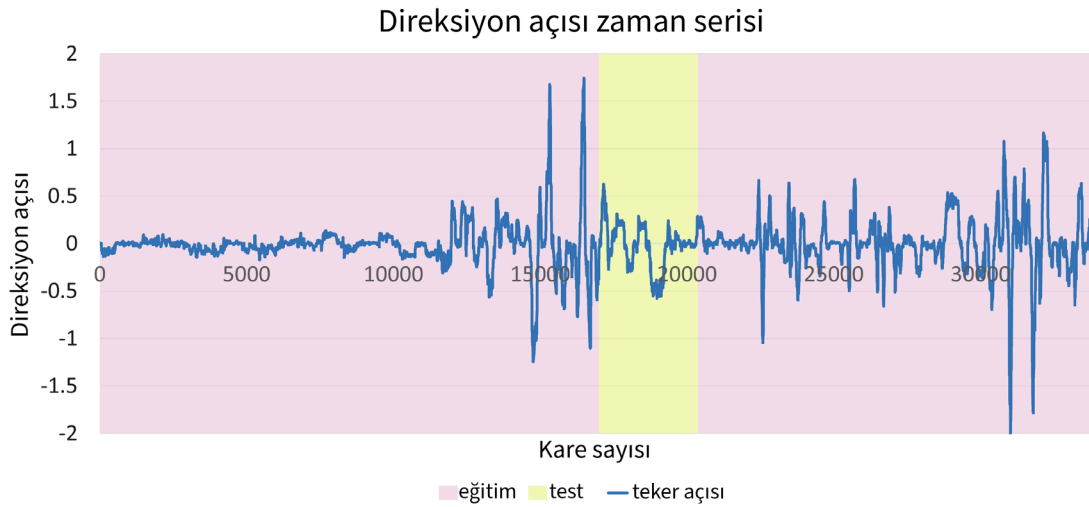
4.1.1.1 Veri ön-işleme

Direksiyon açısı veri setleri genellikle dengesiz bir dağılıma sahiptir. Araçlar, seyir esnasında dönüşlere kıyasla düz yollarda daha fazla zaman harcadığı için, düz

yollarda kaydedilen direksiyon açısı verisi, dönüşler sırasında kaydedilen veriden daha fazladır. Udacity veri kümesi de düz sürüşü temsil eden küçük direksiyon açılı birçok etiket içerir (Şekil 4.4). Bu şekilde bir veri ile eğitim yapıldığı takdirde küçük açılı değerler için aşırı öğrenme olacağından model genellikle düz gitme eğiliminde olabilir. Bu tür bir önyargıyı ortadan kaldırmak ve eğitim verisini dengelemek için küçük direksiyon açılına karşılık gelen verilerin yüzde 50'si rastgele olarak veri kümesinden çıkarılmıştır.



Şekil 4.4 Direksiyon açılarının histogramı. Udacity veri kümesinin ham verilerinde, düz gidişi temsil eden küçük değerli direksiyon açısı etiketlerinin sayısı oldukça fazladır, bu da dengesiz bir dağılıma yol açmaktadır



Şekil 4.5 Bütün veri kümesinin (eğitim + test) direksiyon açılarının zaman serisi. Sarı renkle vurgulanan bölüm, tüm verinin %10'unu oluşturan test verisini temsil etmektedir

Tüm verinin direksiyon açısı zaman serisi Şekil 4.5'te gösterilmektedir. Grafikte sarı renkle vurgulanan bölüm, tüm verinin %10'una karşılık gelen test verisini temsil etmekte olup rastgele belirlenmiş ve eğitim sırasında hiç kullanılmamıştır. Burada önerilen model için verinin ardışık olması çok önemlidir. Hem önerilen model LSTM modülüne sahip olduğu için, hem de otonom sürüşün ilkesi gereği sensör verilerinin ardışık olarak gelmesi ve ardışık verilere birlikte bakıldığında sürüş çevresini daha iyi yorumlanması nedeniyle veri her zaman ardışık olarak ele alınmıştır. Bu yüzden test verisi, rastgele seçilen bireysel görüntülerden ziyade, rastgele seçilmiş sürekli bir aralıktan oluşmaktadır.

Ağa giren bütün giriş görüntülerinin piksel değerleri $[0,255]$ aralığından $[0,1]$ aralığına normalize edilmiştir. Sinir ağları genellikle daha küçük değerlerle çalıştığında daha iyi performans gösterir. Piksel değerlerini $[0,1]$ gibi daha küçük bir aralığa ölçeklendirmek, eğitim sürecinin kararlılığını ve yakınsamasını artırabilir. Bu ölçekleme, daha büyük değerlerle çalışırken oluşabilecek sayısal taşmaları veya alt sınırlamaları da önler. Normalizasyon, eğitim sırasında optimizasyon algoritmasının daha hızlı yakınsamasını sağlar. Daha kararlı gradyanlar elde etmek de öğrenme sürecini hızlandırır. Ayrıca, sinir ağları, daha büyük değerlere karşı önyargı gösterebilir. Giriş verilerinin normalleştirilmesi, bu önyargıyı engeller ve öğrenme sürecinde bir özelliğin yalnızca büyüklüğü nedeniyle baskın olmasını önler.

Veri kümesindeki orijinal görüntü çözünürlüğü 640×480 'dir. Bu değerler, yol çizgileri, diğer araçlar ve yol kenarları gibi sürüşle ilgili özelliklere öncelik vermek amacıyla 640×290 olarak değiştirilmiş ve gökyüzünü içeren üst kısım çıkarılmıştır. Modelde kullanılan EfficientNetV2 modelleri ön eğitilidir. Bu modeller, ImageNet veri kümesindeki kare görüntüler üzerinden eğitildiği için uyumluluğu sağlamak adına giriş görüntülerinin boyutları daha sonra $240 \times 240 \times 3$ boyutuna yeniden boyutlandırılmıştır. Önemli sürüş özelliklerine odaklanmak ve önceden eğitilmiş ağlara uyum sağlamak amacıyla çözünürlük ve boyut değişimi bu model için oldukça önemlidir.

4.1.1.2 Veri artırma

Bir modelin öğrenme süreci boyunca, eğitim kümesinde bulunan veri miktarı arttıkça eğer bu veriler çeşitlilik açısından zenginse, model genellikle daha iyi bir performans sergiler. Bu durum genellikle modelin genelleme yeteneğini geliştirmesine

faydalı olur. Eğitim kümesindeki veri miktarı doğrudan veri eklenerek artırılabilirse daha öncekilerle benzeşmeyen yeni verilerin eklenmesi birinci çözümdür. Ancak bu durum derin öğrenme çalışmalarının birçoğunda mümkün değildir. Genellikle veri toplamak zor bir iştir ve uzun zaman alır. Birçok eğitim kümesinde çoğu zaman eğitim için yeterli veri olmaz. Böyle durumlarda ikinci bir alternatif olarak veri artırma yöntemi uygulanabilir. Veri artırma, mevcut verileri değiştirerek ve aynı etiketlerle yeni örnekler oluşturarak yapay olarak veri miktarını artırmak için kullanılan bir tekniktir. Görüntü işleme, veri artırma tekniklerinin uygulanması açısından oldukça elverişli bir alandır. Çünkü bir görüntü üzerinde uygulanan öteleme, döndürme gibi dönüşüm yöntemleri görüntü içindeki nesneler üzerinde temel değişikliklere neden olmaz. Böylece dönüşümlerle değiştirilmiş ancak sınıfı değişmemiş ekstra eğitim görüntüleri eklenerek ve eğitim kümesinin boyutu artırılarak modelin genelleme yeteneğinin artması sağlanır.

Bu çalışmanın da konusu olan görüntüler için veri artırımı genellikle geometrik dönüşümler veya renk manipülasyonları yoluyla gerçekleştirilir. Geometrik dönüşüm, döndürme, doğrusal olmayan geometrik dönüşümler (affine transformation), kırpma, yeniden boyutlandırma gibi yöntemleri içerir. Bu tür dönüşümler, özellikle nesnelerin farklı açılardan görüntülerini modellemek ve öğrenmek için kullanılır. Renksel dönüşümler ise parlaklık, kontrast ve renk doygunluğu (saturation) gibi özelliklerin değiştirilmesiyle yapılır. Bu tür değişiklikler, aydınlatma koşullarının çeşitliliğini modellemeye yardımcı olabilir. Öte yandan, veri artırma için kullanılan diğer yöntemler arasında rastgele silme, gölge ekleme, ayna döndürme veya optik kaydırma gibi teknikler de bulunmaktadır.

Bu farklı teknikler, özellikle sınırlı sayıdaki veri setleriyle çalışıldığında modelin daha genel ve sağlam bir şekilde öğrenmesine yardımcı olabilir. Ancak, her veri artırma işlemi modelin başarısını kesin olarak artırmayabilir. Veri artırmanın dikkatlice ve kaliteyi koruyarak yapılması gereklidir; aksi takdirde, yanıltıcı veya bozulmuş veri setleri modelin performansını olumsuz etkileyebilir. Bu noktada önemli olan husus, elde edilen yeni görüntülerin, aynı etiketi doğrulayabilme olasılıklarının yüksek olmasıdır.

Örneğin bir görüntü için direksiyon açısının 15° olduğunu varsayalım. Eğitim sürecinde, bu görüntüde netliği azaltmak için bir miktar gürültü eklenmiş olsun. Model, gürültülü bir görüntüden de yine aynı 15° 'lik direksiyon açısını doğru bir şekilde tahmin edebilmelidir. Ancak, görüntüdeki aşırı bozulma durumunda, modelin aynı açıyı öğrenmesi, test aşamasında doğru tahminler yapmasını engelleyebilir. Bu nedenle, veri artırma sürecinde görüntülerin aşırı derecede bozulmamasına dikkat edilmeli ve modelin

gerçek dünya koşullarına daha iyi uyum sağlayabilmesi için dengeli bir veri kümesi oluşturulmalıdır.

Eğitim sürecinde, veri kümesinin kalitesini ve çeşitliliğini artırmak için aynı etiketleri koruyan üç veri artırma tekniği kullanılmıştır. Bu teknikler aynı zamanda, veriye aşırı uyum (overfitting) olasılığını azaltmak amacını da taşımaktadır (Krizhevsky et al., 2012). Bu çalışmada, parlaklık azaltma veya artırma, kontrast, rastgele gölge ekleme, çeviri, döndürme, kırpma gibi farklı veri artırma teknikleri detaylıca incelenmiştir. Çalışma kapsamında, bu teknikler arasından en uygun bulunan üç tanesi veri artırma yöntemi olarak belirlenmiştir.

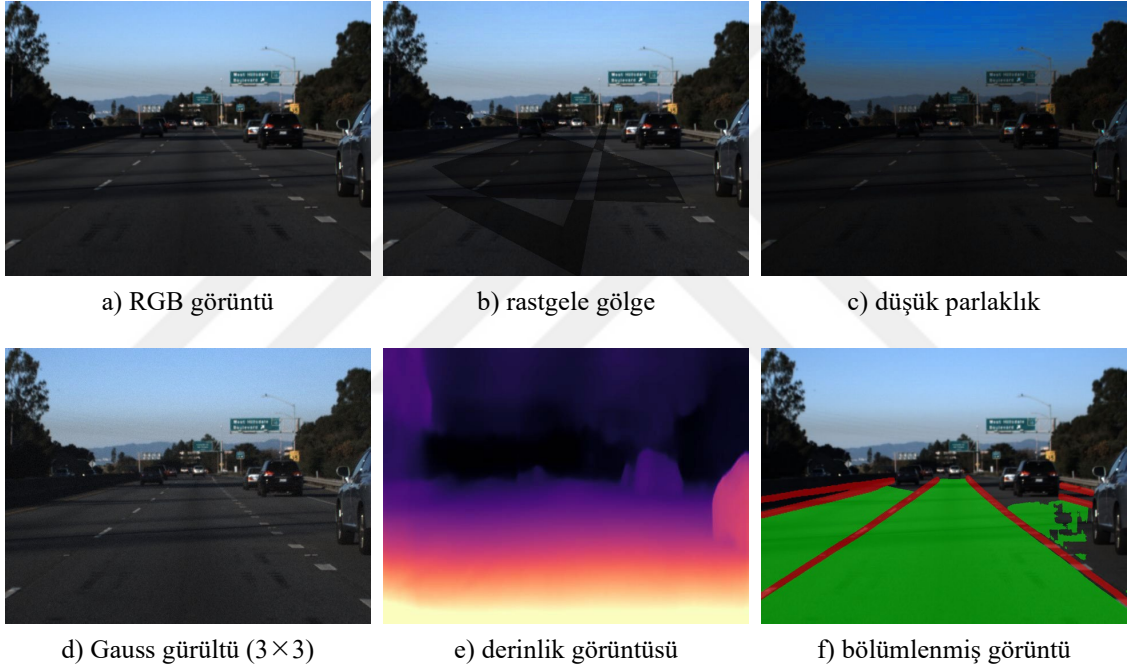
Her veri artırma tekniği, genel veri kümesinin tekrarlanmasını önlemek amacıyla eğitim verisinin farklı kesimlerine uygulanmış ve verinin çeşitliliğini rastgelelik ilkesinden faydalananarak artırmaya çalışmıştır.

Görüntülerdeki aydınlatma değişiklikleri, bilgisayarlı görü problemlerinde daima önemli bir etmen olmuştur. Gölgeler, bazen nesne yanlışlamasına veya mevcut nesnelerle karışıklığa yol açabilir. Özellikle otonom sürüş araştırmalarında, kameralarla yakalanan görüntülerde, parlaklık ve gölge özellikleri günün farklı saatlerine bağlı olarak değişebilir. Araç, bu durumlarda aydınlatma koşullarından bağımsız olarak doğru şekilde algılama yapabilmelidir. Sürüş sırasında çevrede bulunan ağaçlar, binalar, direkler ve diğer araçlar gibi etkenler günün saatine bağlı olarak farklı şekillerde gölgelerin oluşmasına neden olabilir. Bu gölgeler, yeterince eğitilmemiş bir model tarafından cisim olarak algılanabilir ve yanlış kararlara yol açabilir. Hava durumu, güneşin durumu (güneşli, gün batımı, gün doğumu veya öğle saatleri gibi), gölgelerin karakteristiğini etkileyebilir. Otonom sürüş için kullanılacak bir model, bu gölge değişimlerinden etkilenmeden sağlıklı sonuçlar üretebilmelidir.

Bu çalışmada, çeşitli aydınlatma senaryolarını taklit etmek ve modeli bu koşullara daha dayanıklı hale getirmek için rastgele gölge ekleme (Bozkaya et al., 2021) ve düşük parlaklık teknikleri veri artırma yöntemi olarak benimsenmiştir. Sürüş sahnesinde dinamik olarak değişen nesnelerin yanı sıra aydınlanmanın ve gölgelerin de sürekli değişmesi çevresel algılamayı güçleştirir. Gerek sürülebilir alanda olsun gerekse diğer cisimler üzerine düşen gölgeler olsun, ardışık görüntüler arasındaki farklı aydınlanma ve gölge yapıları modeli zorlayacak unsurlardır. Çevredeki yüksek yapıların, ağaçların, yoldaki yüksek araçların gün ışığını aniden kesmesi veya aniden gün ışığının tekrar sahnede görünmesi modelin her şartta aydınlanma şartlarından etkilenmeden başarılı bir

şekilde ortamı yorumlamasını gerektirir. Bu nedenle eğitim kümesindeki veriler üzerinde rastgele gölge ekleme ve düşük parlaklık yöntemleri uygulanarak veri artırımı yapılmıştır.

Ayrıca, kameraların hareket halindeki bir araca monte edilmesi durumunda, hareketin neden olduğu görüntü bulanıklığı ve sensör gürültüsü gibi faktörler tespit sürecini zorlaştırabilir. Modelin bu gibi durumlara karşı dayanıklılığını artırmak için üçüncü bir veri artırma tekniği olarak Gauss tipi gürültü ekleme yöntemi uygulanmıştır; bu da sensör gürültüsünü taklit etmeyi ve modelin sağlamlığını artırmayı amaçlamaktadır. Şekil 4.6’da, veri kümesinden alınmış bir görüntü ve bu görüntüye uygulanmış veri artırma yöntemleri ile görüntüden elde edilmiş derinlik görüntüsü ve anlamsal bölümlenmiş görüntü görülmektedir.



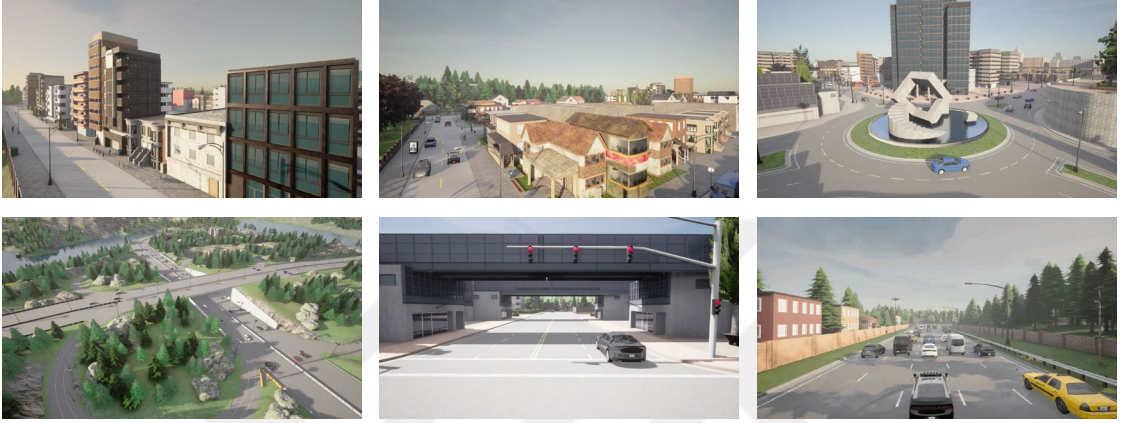
Şekil 4.6 Orijinal görüntü (a) üzerinde veri artırma yöntemlerinin uygulanmasıyla elde edilmiş rastgele gölgeli görüntü (b), düşük parlaklığa sahip görüntü (c), Gauss gürültü eklenmiş görüntü (d) ve orijinal görüntüden elde edilmiş derinlik görüntüsü (e) ile anlamsal bölümlenmiş görüntü (f)

4.2 CARLA Veri Kümesi ve Simülasyon Platformu

CARLA, otonom sürüş sistemlerinin geliştirilmesi ve test edilmesi için tasarlanmış bir açık kaynaklı simülasyon platformudur. Bu platform, gerçek dünya koşullarını sanal bir ortamda simüle ederek, yapay zekâ modellerinin ve sürüş stratejilerinin geliştirilmesine olanak tanır.

4.2.1 CARLA otonom sürüş simülâtörü

CARLA simülâtörü üzerindeki kontrol, Python ve C++'da işlenen ve proje ilerledikçe sürekli büyüyen bir API aracılığıyla sağlanmaktadır. CARLA, simülasyonu çalıştırmak için Unreal Engine'i (Unreal Engine, 2023) temel alır ve yolları ve kentsel ortamları tanımlamak için de OpenDRIVE standardını (OpenDRIVE, 2023) kullanır.



Şekil 4.7 CARLA simülâtöründe farklı kasabalara ait görüntüler. Üst sıradaki görüntüler soldan sağa *Town01*, *Town02*, *Town03*; alt sıradaki görüntüler soldan sağa *Town04*, *Town05* ve *Town06*'dan alınmıştır

CARLA simülâtörü, genişleyebilen bir istemci-sunucu mimarisiyle (client-server architecture) yapılandırılmıştır. Sunucu, simülasyonun tüm yönlerinden sorumludur; bu, sensör oluşturma, fiziksel hesaplamalar, dünya durumu ve aktör güncellemeleri gibi bir dizi görevi içerir. Özellikle makine öğrenimi gibi yoğun işlemlerle uğraşırken, gerçekçi sonuçlar elde etmek için sunucunun özel bir GPU (Graphics Processing Unit- Grafik işlem birimi) üzerinde çalıştırılması gerekir.

İstemci tarafı ise, sahnedeki aktörlerin mantığını kontrol eder ve dünya koşullarını ayarlar. Bu, yeni işlevler sağlamak için sürekli gelişen sunucu ve istemci arasında aracılık eden bir katman olan CARLA API'den (Python veya C++'da) yararlanılarak elde edilir.

- *Trafik yönetici:* CARLA'da öğrenme için kullanılanın yanı sıra araçların kontrolünü sağlayan yerleşik bir trafik kontrolörü vardır. Bu kontrolör, gerçekçi davranışlara sahip kentsel benzeri ortamları yeniden oluşturmak için bir yürütücü görevi görür. Trafik yönetici oto pilot modunda araçları kontrol eder.
- *Sensörler:* Araçlar, gerçek dünyada olduğu gibi simülasyon ortamında da çevreleri hakkında bilgi almak için sensörleri kullanır. CARLA'da

kamera, LiDAR, radar gibi çok sayıda farklı sensör vardır. Bunlar araçlara kolayca eklenebilir ve verileri kaydedilebilir.

- *Kaydedici:* Simülasyonu adım adım tekrar canlandırmak için kullanılır. Kayıtlı bütün araç hareketleri, sensör verileri ile birlikte yeniden canlandırılabilir.
- *Açık nitelikler:* CARLA’da hava koşulları ve ortamda bulunan diğer aktörler (araçlar, yayalar) kontrol edilebilir. Bu unsurlar kolaylıkla özelleştirilebilmekte ve basit yönergelerle oluşturulabilmektedir.

CARLA, gerçek dünya ortamını, şehirlerini ve otoyollarını ve bu sürüş alanlarını işgal eden araçları ve diğer nesneleri taklit eden oldukça gerçekçi bir ortam sunar. Bugün itibarıyla CARLA’da hazır olarak sunulan 12 farklı kasaba mevcuttur (Şekil 4.7 ve Şekil 4.8). “Town” olarak adlandırılan bu yerleşkelerin her biri farklı özelliklere sahiptir. Örneğin bazılarında gidiş-geliş yollar vardır, bazıları bölünmüş yollara sahiptir, bazılarında geniş kavşaklar mevcutken bazılarında sinyalizasyonlu kavşaklar vardır. Bazı haritalar küçük, bazı haritalar çok daha kapsamlı ve büyüktür. Çevre görüntüleri de her bir kasaba için farklı yapıdadır. Ayrıca CARLA, kullanıcılara kendi haritalarını oluşturma imkânı da sunmaktadır. Kullanıcılar burada yol renginden, binalara, sokak lambalarından ağaçlara kadar pek çok farklı özelliği kontrol edebilir.



Şekil 4.8 Soldan sağa *Town04*, *Town05* ve *Town06* kasabalarının havadan görünüşleri

CARLA’da araçlar oto pilot modunda çalışabilir. Bu modda, bir trafik yönetici araçları gerçekçi dünya koşullarına göre kontrol eder. Trafik yöneticinin amacı, simülasyonu gerçekçi kentsel trafik koşullarıyla yürütmektir. Kullanıcılar, örneğin belirli öğrenme koşullarını ayarlamak için bazı davranışları özelleştirebilir.

CARLA simülatörü, kullanıcılara hava ve aydınlanma şartlarını da tam olarak kontrol etme imkânı sunar. Havadaki bulutluluk, yağmur, sis, rüzgâr durumları çok

hafiften çok ağıra kadar ayarlanabilir. Zeminde bulunan ıslaklık miktarı dahi ayarlanabilmekte ve görsel sürüş için zorluklardan biri olan yüzey yansıması modellenebilmektedir. Benzer şekilde güneşin gökyüzündeki konumuna göre güneş ışınlarının geliş açısı seçilebilmektedir. Öğlen açık havada veri toplanabileceği gibi, yağmurlu gün batımında da veri toplanabilmektedir.



Şekil 4.9 CARLA sürüş simülatörü ile toplanmış veri kümesinden farklı hava (ıslak zemin, orta yağmurlu, bulutlu, açık) ve aydınlanma şartlarına (gündüz, gün batımı) ait görüntüler

Şekil 4.9’da CARLA sürüş simülatörü ile oluşturulan veri kümesinden alınmış farklı hava ve aydınlanma şartlarını gösteren kamera görüntüleri görülmektedir. Şekilde hem açık ve yağışsız havada alınmış görüntüler hem de yağışlı ve ıslak zemin şartlarında alınmış görüntüler görülmektedir. Görüldüğü yağışlı durumlarda yol yüzeyinde oluşmuş su birikintileri mevcuttur. Bu alanlar sürülebilir alan tespiti açısından modelin üç görü işleme kolu olan RGB, derinlik ve segmentasyon alt ağları için oldukça zorlu durumlardır. Bu alanların zeminle birliktymiş gibi değil de ayrı birer cisim gibi algılanma ihtimali de oldukça yüksektir. Ancak tez çalışmasında önerilen yöntem, test verisinde bu tür alanlar olmasına rağmen oldukça başarılı sonuçlar üretmiştir. Yine şekilde de görüleceği üzere sürüş sahnesinde bulunan farklı araçlar, yol çizgileri, trafik ışıkları, trafik levhaları, binalar ve ağaçlar, gerçek dünya verilerini oldukça başarılı bir şekilde temsil etmektedir.

4.2.2 CARLA veri kümesi

Derin öğrenme sistemlerinin tek bir veri kümesi üzerinde gösterdikleri başarı bazen güvenilir bir gösterge olmayabilir. Bir derin öğrenme modelinin, benzer ortamlarda toplanmış eğitim ve test veri kümeleriyle elde ettiği başarı, modelin aşırı uyum (overfitting) eğiliminde olduğunu işaret edebilir. Aşırı uyum, modelin eğitim verilerine aşırı derecede adapte olması ve genelleme yeteneğinin azalması durumunu ifade eder. Bu durumda, model eğitim verilerine çok iyi uyum sağlar ve benzer ortamlarda toplanmış test verileriyle başarılı sonuçlar üretebilir, ancak farklı ortamlarda performansı düşebilir.

Otonom sürüş örneği üzerinden düşünüldüğünde, örneğin bir şehrin aynı bölgesinden toplanmış eğitim ve test kümeleriyle başarılı sonuçlar elde edilirken, aynı şehrin farklı bölgelerinde toplanmış eğitim ve test verileriyle aynı başarı seviyesine ulaşılamayabilir.

Önerilen modelin genellenebilirliğini ve sağlamlığını değerlendirmek amacıyla CARLA sürüş simülatörü özellikle seçilmiştir. CARLA bünyesinde, farklı yapılara sahip kasabalar bulunmaktadır (Şekil 4.7). Bu veri kümesi oluşturulurken, eğitim, doğrulama ve test kümeleri tamamen farklı kasabalardan toplanmıştır. Eğitim verileri *Town01*, *Town02* ve *Town03* kasabalarında toplanmışken, eğitim sırasında doğrulama için kullanılan veri kümesi *Town04*'ten elde edilmiştir. Test verileri ise *Town05*'te toplanmıştır. Bu kasabaların her biri farklı özellikte yollara ve çevre görüntülerine sahiptir. Dolayısıyla eğitim, doğrulama ve test görüntülerinin yapıları birbirinden oldukça farklıdır.

Doğrulama kümesi, modelin eğitim verileri üzerindeki performansının dışında, henüz görmediği, ayrı bir veri kümesi üzerindeki performansını değerlendirmek için kullanılır. Genellikle eğitim için ayrılan verinin bir kısmı da doğrulama için ayrılarak kullanılır, doğrulama verileri eğitim esnasında kesinlikle modele girmez. Bu çalışmada, doğrulama verisi olarak ilk üç kasabadan toplanan eğitim verisinin bir kısmı değil, tamamen farklı bir kasabadan toplanan veriler doğrulama verisi olarak kullanılmıştır. Doğrulama süreci, modelin aşırı uyuma gitmesini önler ve modelin eğitim sonrası hiç görmediği bir test verisi üzerinde de başarılı sonuçlar elde edebilmesi için genelleme yeteneğini artırır.

CARLA ile oluşturulan veri kümesinde RGB kamera görüntüsü, derinlik görüntüsü ile birlikte araç hızı ve direksiyon açısı toplanmıştır. Kamera görüntüleri 90°

görüş alanına sahip bir kamera ile 640×480 çözünürlükte toplanmıştır. Bu veriler ortalama 20Hz ile toplanmıştır. Tüm kasabalarda, yaya sayısı ve araç sayısı 40 olarak ayarlanmıştır. Veri toplanan beş şehrin hepsinde farklı aydınlanma ve farklı hava koşullarında veri toplanmıştır. Bu aydınlanma-hava koşulları çiftleri, <gündüz-açık>, <gün batımı- ıslak zemin>, <gündüz- bulutlu, ıslak zemin>, <gündüz- orta yağmurlu> şeklindedir. Toplanan verinin ayrıntıları, Tablo 4.2’de ayrıntılı olarak sunulmuştur.

Tablo 4.2 CARLA sürüş simülatörü ile toplanan verinin ayrıntıları

		Gündüz			Gün batımı	Toplam
		Açık	Bulutlu ıslak zemin	Orta yağmurlu	Islak zemin	
Eğitim	<i>Town01</i>	9.000	-	-	1.080	10.080
	<i>Town02</i>	9.000	1.080	-	-	10.080
	<i>Town03</i>	9.000	-	1.080	-	10.080
Doğrulama	<i>Town04</i>	6.750	270	270	270	7.560
Test	<i>Town05</i>	3.750	150	150	150	4.200

CARLA sürüş simülatörü ile toplam 42.000 kare veri toplanmıştır. Eğitim verileri *Town01*, *Town02* ve *Town03*’ten; doğrulama verileri *Town04*’ten; test verileri ise *Town05*’ten toplanmıştır. Veri içindeki her bir zaman adımı için RGB görüntü, derinlik görüntüsü, araç hızı ve direksiyon açısı kaydedilmiştir. Toplanan 42.000 karenin, Udacity veri setinde olduğu gibi, %10’una karşılık gelen 4.200 kare, test verisi olarak *Town05*’te toplanmıştır. Dolayısıyla *Town05*, yalnızca test için kullanılmıştır. Test verisinin %90’ı olan 3750 kare <gündüz- açık> şartlarında toplanırken, kalan %10’u <gündüz- bulutlu, ıslak zemin>, <gündüz- orta yağmurlu> ve <gün batımı- ıslak zemin> şartlarında toplanmıştır. Toplam veriden geriye kalan 37.800 karenin %20’sine karşılık gelen 7.560 kare ise doğrulama verisini oluşturmaktadır. Doğrulama verisi de tamamen ayrı bir kasabada (*Town04*), farklı aydınlanma ve hava koşullarında toplanmıştır. Benzer şekilde doğrulama verisinin %90’ı olan 6.750 kare <gündüz- açık> şartlarında toplanırken, kalan %10’u <gündüz- bulutlu, ıslak zemin>, <gündüz- orta yağmurlu> ve <gün batımı- ıslak zemin> şartlarında toplanmıştır. Eğitim verisi olarak ise 30.240 veri, *Town01*, *Town02*, *Town03* kasabalarında toplanmıştır. Eğitim verisinin %90’ı olan 27.000 kare <gündüz- açık> şartlarında toplanırken, kalan %10’u farklı aydınlanma ve hava koşullarında toplanmıştır. Eğitim verisinin bu kalan kısmında 1.080 kare <gündüz- bulutlu, ıslak zemin> şartlarında, 1.080 kare <gündüz- orta yağmurlu> havada ve 1.080 kare <gün batımı- ıslak zemin> şartlarında toplanmıştır.

Geliştirilen model, ağa girdi olarak RGB görüntü ile bu RGB görüntüden elde edilmiş derinlik görüntüsü ve anlamsal bölümlenmiş görüntüyü kullanır. Daha önce de bahsedildiği gibi derinlik tahmini, Monodepth2 algoritması ile yapılmaktadır. CARLA sürüş simülatörü RGB görüntü ile birlikte derinlik görüntüsü de verebilmektedir. Bu yüzden ham görüntülere ilave olarak CARLA'nın sağladığı derinlik görüntüleri de toplanmıştır (Şekil 4.10). Önerilen model, hem Monodepth2 algoritmasının ürettiği derinlik görüntüleri ile hem de CARLA'nın sağladığı derinlik görüntüleri ile test edilmiştir. Elde edilen sonuçlar Bölüm 5.3.1.2 ve Bölüm 5.3.2.2'de sunulmuştur.



Şekil 4.10 CARLA veri kümesinden alınan bir RGB görüntüye ait farklı derinlik görüntüleri

Udacity veri kümesi, kesintisiz bir rota boyunca toplanmıştır. Bu durum, eğitim kümesinde benzer görüntülerin bulunma olasılığı nedeniyle modelin genelleme yeteneğini geliştirmek adına veri artırma tekniklerinin kullanılmasını gerektirmiştir. Veri artırma teknikleri, sınırlı veri durumlarında, eğitime katılan veri sayısını artırmanın yanı sıra veri çeşitliliğini de sağlayabilir.

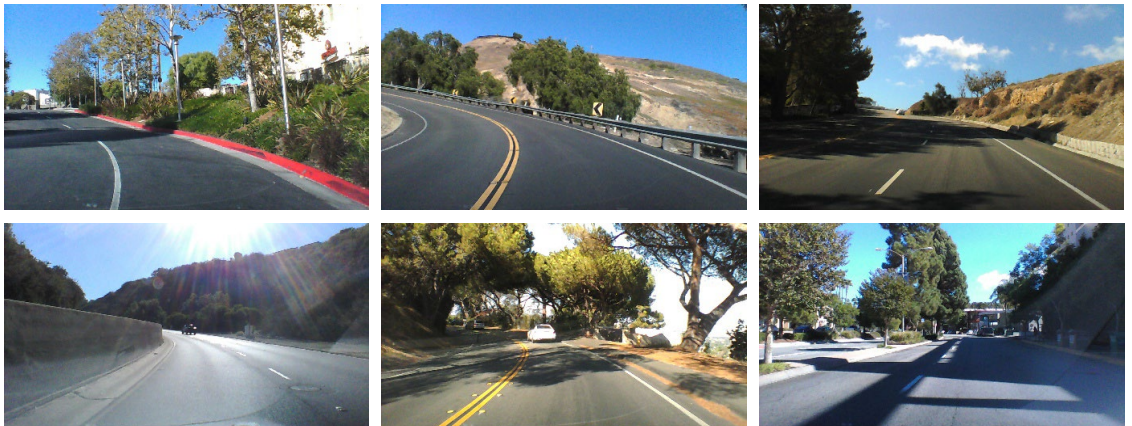
CARLA ile yapılan çalışmada ise veri artırma teknikleri kullanılmamıştır. Bunun iki temel nedeni vardır: Birincisi, CARLA ortamında veri miktarı tamamen geliştiricinin kontrolündedir ve eğer veri eksikliği yaşıyorsa, simülatörde daha fazla veri toplanarak bu sorun giderilebilir. İkincisi ise, veri çeşitliliğini sağlamak amacıyla CARLA eğitim seti üç farklı kasabada toplanmış, doğrulama ve test verileri de aynı şekilde farklı kasabalardan elde edilmiştir. Ek olarak, aynı kasaba içinde toplanan verilerde çeşitlilik sağlanması için farklı aydınlatma ve hava koşullarında da veri toplanmıştır. Bu nedenle, gerçek veri toplama yöntemiyle doğal çeşitlilikte veri artırma tercih edilmiş ve mevcut veri, yazılım aracılığıyla değil, farklı türlerde ve gerçek ortamlarda toplanmış verilerle genişletilmiştir.

4.2.2.1 Veri ön-işleme

Udacity veri kümesinde olduğu gibi CARLA veri kümesinde de kullanılan bütün giriş görüntülerinin piksel değerleri $[0,255]$ aralığından $[0,1]$ aralığına normalize edilmiştir. Veri kümesinde toplanan görüntülerin çözünürlüğü 640×480 'dir. Udacity veri kümesinde olduğu gibi, bu değerler, yol çizgileri, diğer araçlar ve yol kenarları gibi sürüşle ilgili özelliklere öncelik vermek amacıyla 640×380 olarak değiştirilmiş ve gökyüzünü içeren üst kısım çıkarılmıştır. Yine Udacity veri kümesi ile yapılan çalışmada olduğu gibi, kullanılan EfficientNetV2 modelleri ImageNet ön eğitilmiş olduğu için giriş görüntülerinin boyutları daha sonra 240×240 boyutuna yeniden boyutlandırılmıştır.

4.3 Sully Chen Veri Kümesi

Sully Chen veri kümesi, direksiyon açısı tahmini üzerine odaklanan bir veri kümesidir. Bu veri kümesi, çeşitli sürüş senaryolarını içeren kamera görüntüleri ve bunlarla ilişkili olarak sadece direksiyon açısı bilgisini içerir. Udacity ve CARLA veri kümelerinden farklı olarak, Sully Chen veri kümesi özellikle direksiyon açısı tahmini üzerine yoğunlaşmış ve hız verisi içermemektedir. Sully Chen tarafından oluşturulmuş iki farklı veri kümesi vardır. Bunlardan ilki 45,500 görüntü ve direksiyon açısı içeren veri kümesidir. İkinci veri kümesi ise 63,000 görüntü içermektedir. Bu tez çalışması kapsamında 45,500 görüntü içeren birinci veri kümesi kullanılmıştır.



Şekil 4.11 Sully Chen veri kümesinden alınmış farklı ışıklandırma, farklı yol yapıları gibi değişik sürüş koşullarını içeren bazı görüntüler

Bu veri kümesi de Udacity veri kümesine benzer şekilde hem şehir içi hem de şehir dışı denebilecek farklı sürüş koşulları ve değişken trafik yoğunlukları gibi senaryoları barındırmaktadır. Sully Chen veri kümesi gündüz ve güneş ışığının en verimli olduğu saatlerde toplanmıştır. Bu yüzden görüntüler içinde, Udacity veri kümesinde olduğu gibi çok ışıklı veya koyu gölgeli alanlar barındırmaktadır. Şekil 4.11’de Sully Chen veri kümesinden alınmış farklı ışıklandırma ve farklı yol yapıları gibi değişik sürüş koşullarını içeren bazı görüntüler görülmektedir.

Bu veri kümesindeki görüntüler 455×256 boyutlarında toplanmıştır. Daha önceki veri kümelerinde olduğu gibi sürüşle ilgili olmayan gökyüzü kısmı tüm görüntülerde kırılmış ve görüntü boyutu 380×256 ’ya düşürülmüştür. Yine daha önceki veri kümelerinin işlenmesi sırasında uygulanan prosedür burada da uygulanmış ve ESA ağı ImageNet ön eğitilmiş olduğu için ve ImageNet’in eğitim görüntülerinin boyutları 224×224 olduğu için görüntüler 240×240 olarak yeniden boyutlandırılmıştır.

Sully Chen veri kümesinin de %10’u test verisi olarak ayrılmıştır. Dolayısıyla 4,550 adet test verisi mevcuttur. Geriye kalan 40,950 görüntünün %20’si olan 8,190 görüntü doğrulama verisi olarak ayrılmış kalan 32,760 görüntü ise eğitim verisi olarak kullanılmıştır.

Sully Chen veri kümesinde de ağa giren bütün giriş görüntülerinin piksel değerleri $[0,255]$ aralığından $[0,1]$ aralığına normalize edilmiştir. Direksiyon açısı değerleri ise, literatürde bu veri kümesini kullanan ve ilerde karşılaştırmalı sonuçlar bölümünde de bahsedilecek olan çalışmalarda radyan olarak kullanıldığı için radyan cinsine çevrilmiştir. Sully Chen veri kümesi için herhangi bir veri artırma yöntemi uygulanmadan önerilen model eğitilmiş ve teste tabi tutulmuştur.

5. DENEYSEL SONUÇLAR

Modelin mimarisi ve performansı çok sayıda deneme ve eksiltme neticesinde sistematik olarak iyileştirilmiştir. Uygun özellikte alt ağlar ve modüller belirlenirken aynı zamanda ileri ön-işleme ve veri artırma teknikleri de uygulanmıştır. Modelin içinde bulunan alt ağların ve modüllerin parametreleri de yine bu denemeler neticesinde elde edilmiştir. Bu süreç modelin mevcut ve son haline ulaşmasını sağlayarak amaçlanan görevlerin etkin bir şekilde yerine getirilmesini sağlamıştır. Takip eden bölümlerde sırasıyla, modelin eğitimi sırasında kullanılan alt yapının özellikleri ve seçilen parametreler; çok geniş bir tabanda Udacity veri kümesi üzerinde yapılmış ayrıntılı bir eksiltme çalışmasının sonuçları ve bu sonuçlara ait yorumlar; Udacity, CARLA ve Sully Chen veri kümeleri ile elde edilen deneysel sonuçlar; bu sonuçların literatürdeki diğer çalışmalarla kıyaslanması ve modelin çalışma süresi analizi ayrıntılı bir şekilde anlatılacaktır.

5.1 Eğitim Ayrıntıları

Geliştirilen model Udacity, CARLA ve Sully Chen veri kümeleri üzerinde eğitilip test edilmiştir. Daha önce Bölüm 4.1’de bahsedilen Udacity veri kümesi, modelin amaçladığı görevleri gerçekleştirmek için belirlenmiş başlıca veri kaynağıdır. Ancak elde edilen sonuçları pekiştirmek ve modelin başarısının veri kümesine özgü olmadığını doğrulamak adına CARLA ve Sully Chen veri kümelerinde de eğitim ve test gerçekleştirilmiştir.

Bu çalışma sürecinde, Udacity ve CARLA veri kümelerinde tek bir kameradan elde edilen sürüş sahnesi görüntüleri ile birlikte eş zamanlı olarak kaydedilen direksiyon açısı ve hız bilgileri eğitim verisi olarak kullanılmıştır. Ancak Sully Chen veri kümesi, kamera görüntüleri ile birlikte yalnızca direksiyon açısı içermekte olup hız verisini barındırmamaktadır. Önerilen ağ yapısı, uzaysal özellikleri üç adet EfficientNetV2 tabanlı ESA dalıyla kodlarken, zamansal dinamikleri yakalamak için ise dördüncü dal olarak bir LSTM bloğu kullanır. Bu dört dalın birleşimi hem direksiyon açısı hem de hız için birleşik bir tahmin üretmektedir (Şekil 3.1).

Eğitim esnasında, ESA kolları, modelin görsel girdi ile kontrol eylemleri arasındaki ilişkiyi öğrenmesini sağlayan ilgili direksiyon açısı ve hız bilgisiyle etiketlenmiş monoküler görüntülerle beslenir. LSTM bloğu da geçmişe dönük olarak

daima son on direksiyon açısı ve hız değerini giriş olarak alır. Modelde kullanılan her bir ESA, ImageNet veri kümesi üzerinde ön-eğitilmiştir. Tüm deneyler, bir RTX 3080 Ti GPU üzerinde Keras (Chollet, 2015) ve Tensorflow (Abadi et al., 2016) kullanılarak gerçekleştirilmiştir.

Tensorflow, Google Brain tarafından geliştirilmiş Python tabanlı açık kaynaklı bir makine öğrenimi ve derin öğrenme kütüphanesidir. Tensorflow, araştırmacıların karmaşık matematiksel işlemlerle boğulup zaman kaybetmesini önlemek amacıyla geliştirilmiş pek çok fonksiyon barındırır. Sadece CPU üzerinde değil GPU ve TPU üzerinde de paralel işlem yaparak çalışabilir. Tensorflow ile geliştirilmiş programlar kolaylıkla C++ ve Javascript gibi dillere dönüştürülebilir. Tensorflow yalnızca basit bir kütüphane değildir. Geniş bir bileşenler ekosistemine ev sahipliği yapan geniş bir platformdur. Bu bileşenlerin birçoğu Google tarafından geliştirilmiştir. Keras'a göre daha düşük seviyeli bir kütüphanedir.

Keras, Tensorflow'a kıyasla daha yüksek seviyeli işlemler sağlar. Keras, yapay zekâ çalışmaları için geliştirilmiş Python arayüzü sağlayan ve açık kaynak kütüphaneye sahip olan bir Uygulama Programlama Arabirimi (API)'dir. Keras, kullanıcıyı daha az yormak adına çok geniş bir kütüphaneye sahiptir. Tensorflow üzerine inşa edilmiştir. Yapay zekada gerekli pek çok araç, içinde mevcuttur. Başlangıçta hızlı deneylerini gerçekleştirebilmek adına geliştirilmiştir. Makineler için değil, insanlar için geliştirilmiş bir API'dir. Tutarlı ve basit bir çalışma yapısına sahip olan Keras, yaygın kullanım durumları için gerekli eylem sayısını minimuma indirir. Ayrıca Keras kullanıcı hatası gibi durumlarda net ve eyleme geçirilebilir geri bildirim sağlar. Bu da başlangıç seviyesinden uzman seviyesindeki kullanıcılara kadar tercih sebebi olmasını sağlamıştır. Keras 2015 yılında ilk çıktığında, başlangıçta Theano tensor işleme kütüphanesi üzerine inşa edilmiştir. Ancak daha sonra kendisinden sekiz ay sonra piyasaya çıkan Tensorflow yapısını kullanmaya başlamıştır. Bugün Tensorflow ve Keras simbiyotik bir ilişkiye sahiptir ve genellikle kullanıcılar tarafından iç içe kullanılırlar (Chollet, 2017).

Tensorflow ve Keras aracılığıyla geliştirilen bu model, Stokastik Gradyan İniş (SGD) yöntemi kullanılarak Nadam optimize edici (Dozat, 2016) ile 30 dönem boyunca eğitilmiştir. Nadam optimize edici için Tensorflow'un varsayılan parametreleri tercih edilmiştir: Birinci moment β_1 için azalma oranı 0,9 olarak ayarlanmış, ikinci moment β_2 için azalma oranı 0,999 olarak ayarlanmış ve sayısal kararlılık sabiti ϵ ise $1e-07$ olarak belirlenmiştir. Nadam, temelde Kingma ve Ba tarafından geliştirilen Adam yöntemine (Kingma and Ba, 2015) dayanır. Adam, momentum ve uyarlanabilir öğrenme oranı olmak

üzere iki önemli bileşeni içerirken, Nadam bu bileşenlerdeki geliştirmelerle Adam'ın ötesine geçmeyi hedefler. Ancak, düzenli momentumun kavramsal ve ampirik olarak Nesterov'un hızlandırılmış gradyanı (NAG) adı verilen benzer bir algoritmanın gerisinde olduğu gösterilebilir. Nadam yöntemi, bu temel bileşenlerdeki geliştirmelerle, özellikle de Adam'a kıyasla daha hızlı ve güvenilir bir şekilde global minimuma yaklaşma yeteneğiyle dikkat çeker.

Bununla birlikte, Nadam'ın özellikle büyük veri setleri veya karmaşık ağ yapılarıyla kullanıldığında, genellikle daha dengeli ve güvenilir bir performans sunma eğiliminde olduğu gözlemlenmiştir. Bu, optimize edici algoritmalar arasında Nadam'ın giderek daha popüler hale gelmesine ve tercih edilmesine neden olan önemli bir faktördür.

Eğitim sürecinde modelin verimli bir şekilde yakınsamasını sağlamak için öğrenme oranını optimize etmek kritik bir adımdır. Bu çalışmada, öğrenme performansını artırmak amacıyla bir öğrenme oranı azalma tekniği kullanılmıştır. Bu mekanizma, eğer doğrulama kaybı (validation loss) iki ardışık dönem boyunca belirli bir eşiğin üzerinde iyileşmezse öğrenme oranını 0,2 faktöründe azaltarak süreci düzenler. Başlangıç öğrenme oranı, eğitimin ilk aşamasında 0,001 olarak belirlenmiştir.

Bu azalma stratejisi, modelin eğitimindeki performansı dengelemek ve aşırı uyumu engellemek adına oldukça kritiktir. Doğrulama kaybında belirgin bir düzelme olmadığında öğrenme oranının düşürülmesi, modelin daha dengeli ve istikrarlı bir şekilde öğrenmesini sağlar. Aynı zamanda, başlangıçta düşük bir öğrenme oranı belirleyerek modelin daha istikrarlı bir şekilde global minimuma doğru ilerlemesine yardımcı olur. Bu strateji, modelin daha dikkatli bir şekilde öğrenmesini sağlayarak genellikle daha başarılı sonuçlar elde etme eğiliminde olur.

Derin öğrenme modellerinin eğitimi, elde edilen sonuçların analizi ve nihayetinde iyi bir öğrenme performansı sergileyen başarılı bir modelin geliştirilmesi, temel olarak kullanılan kayıp fonksiyonları vasıtasıyla gerçekleşir. Eğitim süreci boyunca model, tahminler üretir ve bu üretilen tahminlerin başarı seviyesini değerlendirmek için ölçümler gereklidir. Bu ölçümler, beklenen değerlerle karşılaştırılarak modelin performansını gösterir. Kayıp fonksiyonu, gerçek değerlerle tahmin edilen değerler arasındaki farkı ölçerek modelin başarısını temsil eden bir skaler değer üretir. Optimizasyon süreci, bu kayıp fonksiyonunu en aza indirmek üzere modelin ağırlıklarını ve önyargılarını iteratif olarak ayarlar.

Sinir ağlarındaki ağırlıklar, farklı katmanlardaki nöronlar arasındaki bağlantıların güçlerini temsil eder. Eğitim süreci boyunca, bu ağırlıklar geri yayılım yoluyla güncellenir. Yani, her bir ağırlığa göre kayıp fonksiyonunun gradyanları hesaplanır. Bu gradyanlar, her bir ağırlığın kaybını azaltmak için yapılması gereken değişikliklerin yönünü ve büyüklüğünü belirtir. Bu sayede, modelin güncellenmiş ağırlıkları, kayıp fonksiyonunu minimize etmek ve daha doğru tahminler yapmak için optimize edilmiş hale gelir.

Literatürde geliştirilmiş çok farklı yapılarda kayıp fonksiyonları mevcuttur. Bu fonksiyonlar, kullanılan verinin yapısına göre seçilirler. En çok kullanılan kayıp fonksiyonlarından biri Ortalama Karesel Hata (MSE) fonksiyonudur. MSE, tahmin edilen değerlerle gerçek değerler arasındaki farkı aldıktan sonra karesini alır. Kare alma işlemi, negatif ve pozitif fark değerlerinin eşit ağırlıkta değerlendirilmesini sağlar. Büyük farklar, kare alındığı için daha da büyür ve MSE'ye daha fazla katkıda bulunur. Bu durum, modelin bu tür büyük hataları azaltmaya veya minimize etmeye çalışmasını sağlar. MSE'nin optimize edilmesi veya türevinin alınması, matematiksel olarak kolaydır. Karelerin alınması, hesaplama ve analitik işlemleri basitleştirir, bu da optimizasyon algoritmalarıyla daha uyumlu hale getirir. Ancak kare alma işlemi bazı durumlarda dezavantaja dönüşebilir. Örneğin kare alma işlemi aykırı değerleri daha da vurgular. Bu durum, aykırı değerlerin modelin performansını daha fazla etkilemesine neden olabilir. Karesel hata fonksiyonuna ait denklem aşağıdaki gibidir:

$$\mathcal{L} = \frac{1}{n} \sum_{i=1}^n (s_i - \tilde{s}_i)^2 \quad (5.1)$$

Denklem (5.1)'de $\mathcal{L}$, karesel hatayı; n , örnek sayısını; s_i , i numaralı örnek için gerçek değeri; $\tilde{s}_i$ ise i . numaralı örnek için tahmin değerini ifade etmektedir.

Bir diğer kayıp fonksiyonu ise ortalama mutlak hata (MAE) fonksiyonudur. Gerçek değerler ile tahmin edilen değerler arasındaki farkların mutlak değerlerinin ortalamasının alınmasıyla elde edilir. Bu durumda, farkların pozitifliği veya negatifliği dikkate alınmaz, sadece farkın büyüklüğü önemlidir. MAE, aykırı değerlerin etkisini daha dengeli bir şekilde ele alır. Çünkü bu ölçüm, farkların mutlak değerlerini alır, bu da büyük farkların diğer farklarla aynı şekilde önemsenmesine yol açar. MAE, aykırı değerleri daha az vurgular, bu da modelin bu tür büyük hataları düzeltmemesine neden olabilir.

$$\mathcal{L} = \frac{1}{n} \sum_{i=1}^n |s_i - \tilde{s}_i| \quad (5.2)$$

MAE’de de MSE’ye benzer şekilde, $\mathcal{L}$, karesel hatayı; n , örnek sayısını; s_i , i numaralı örnek için gerçek değeri; $\tilde{s}_i$ ise i numaralı örnek için tahmin değerini ifade etmektedir (Denklem (5.2)).

Makine öğrenmesinde sınıflandırma ve regresyon için kullanılan çok sayıda kayıp fonksiyonu mevcuttur. Bunlardan regresyon için kullanılanlardan bazıları, MSE, MAE ve Huber kaybıdır. Sınıflandırma için kullanılanlar ise, ikili çapraz entropi kaybı, kategorik çapraz entropi kaybı, menteşe kaybı ve logaritmik kayıptır.

Bu çalışmada çeşitli kayıp fonksiyonları denenmiştir. Ancak, ortalama karesel hata (MSE), hem direksiyon açısı hem de hız için kayıp fonksiyonu ($\mathcal{L}$) olarak benimsenmiştir. Toplam kayıp $\mathcal{L}_{toplaml}$, direksiyon açısı kaybı $\mathcal{L}_{direksiyon}$ ve hız kaybı $\mathcal{L}_{hız}$ ’ın ağırlıklı toplamı olarak hesaplanmıştır:

$$\mathcal{L}_{toplaml} = \lambda_{dr} \mathcal{L}_{direksiyon} + \lambda_{hz} \mathcal{L}_{hız} \quad (5.3)$$

Denklem (5.3)’te λ_{dr} ve λ_{hz} , sırasıyla direksiyon açısı ve hız için kayıp ağırlıklarıdır (Tablo 5.1).

Sırasıyla $\mathcal{L}_{direksiyon}$ ve $\mathcal{L}_{hız}$ olarak gösterilen direksiyon açısı ile hız tahminlerine yönelik kayıp fonksiyonları şu şekilde tanımlanır:

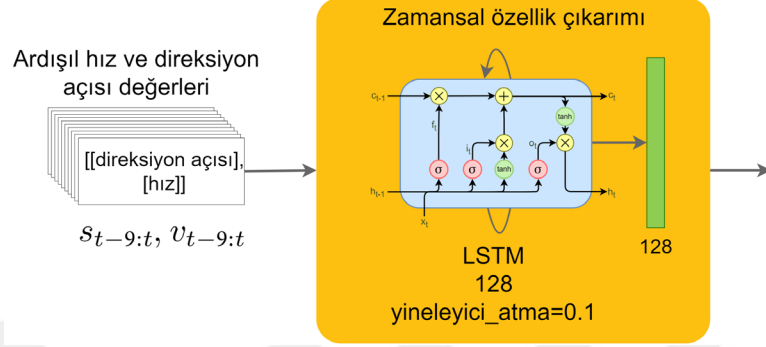
$$\mathcal{L}_{direksiyon} = \frac{1}{n} \sum_{i=1}^n (s_i - \tilde{s}_i)^2 \quad (5.4)$$

$$\mathcal{L}_{hız} = \frac{1}{n} \sum_{i=1}^n (v_i - \tilde{v}_i)^2 \quad (5.5)$$

Denklem (5.4) ve (5.5)’te, n , toplam örnek sayısını; s_i ve v_i , i . örneğin gerçek direksiyon açısı ve hızını temsil ederken; $\tilde{s}_i$ ve $\tilde{v}_i$, karşılık gelen tahmin değerlerini ifade etmektedir.

Modelin dördüncü dalında, sıralı hız ve direksiyon verilerini işlemek için 128 hücreli bir LSTM bloğu kullanılmıştır (Şekil 5.1). LSTM birimi, zamansal dokuları ve

bağımlılıkları çıkarmak için son on direksiyon açısını ve hız değerini dikkate alır. Zamansal özelliklerin entegrasyonu, tahmin doğruluğunu ve bağlam farkındalığını artırır. Burada, LSTM içinde tekrarlayan durumların doğrusal dönüşümü için düşecek birimlerin oranı (recurrent_dropout) %10 olarak belirlenmiştir.



Şekil 5.1 Direksiyon açısı ve hızdaki zamansal bağımlılıkları çıkartmak için kullanılan LSTM dalı

5.2 Eksiltme Çalışması

Derin öğrenme modelinin yapısını detaylı bir şekilde analiz etmek, model bileşenlerinin görsel otonom sürüş sonuçlarına olan katkısını görmek ve bu bileşenlere ait en uygun parametreleri belirlemek amacıyla Udacity veri kümesi ile detaylı bir eksiltme çalışması gerçekleştirilmiştir. Bu eksiltme çalışmasının, gelecekteki çalışmalar için değerli bir referans olacağı düşünülmektedir. Tablo 5.1’de sunulan her bir sonuç, aynı testin 5 kez gerçekleştirilmesi ile elde edilmiş sonuçların ortalamasıdır.

Test 1 ile *Test 7* arasındaki testlerde, farklı ESA alt ağları kullanılarak ve farklı Özellik Kümesi Uzunlukları (ÖKU) seçilerek bireysel alt modellerin ana modele katkıları incelenmiştir. ÖKU, her bir alt ağ tarafından üretilen üst düzey özellik vektörüdür. Her bir alt ağ olmadan yapılan analizler, en kötü sonuçların RGB modülünün olmadığı durumda ortaya çıktığını ve hem hızı hem de direksiyon açısını olumsuz etkilediğini ortaya koymuştur. Yalnızca derinlik bileşeninin veya yalnızca semantik segmentasyon bileşeninin modelden çıkarılması, direksiyon açısı üzerinde olumsuz bir etkiye sahip olsa da hız üzerindeki etkisi nispeten küçük kalmaktadır.

Tablo 5.1 Eksiltme çalışması: Modellerin ve ayar değişkenlerinin model performansına katkıları

Test No	Özellik kümesi uzunluğu				Yoğun katman	Alt ağlar			Modal ağırlıklar				Kayıp	RMSE	
	RGB	Der.	Seg.	LSTM		RGB	Der.	Seg.	RGB	Der.	Seg.	LSTM		Dir. açısı	Hız
1	128	-	-	128	64	EV2-B0	-	-	0,66	-	-	0,33	MSE	0,0338	0,0569
2	128	128	-	128	64	EV2-B0	EV2-B0	-	0,33	0,33	-	0,33	MSE	0,0361	0,0561
3	128	-	128	128	64	EV2-B0	-	EV2-B0	0,33	-	0,33	0,33	MSE	0,0324	0,0577
4	-	128	128	128	64	-	EV2-B0	EV2-B0	-	0,33	0,33	0,33	MSE	0,0484	0,0612
5	128	128	128	128	64	EV2-B0	EV2-B0	EV2-B0	0,22	0,22	0,22	0,33	MSE	0,0282	0,0577
6	-	-	-	128	64	-	-	-	-	-	-	1,0	MSE	0,0265	0,567
7	200	100	100	128	64	EV2-B0	EV2-B0	EV2-B0	0,22	0,22	0,22	0,33	MSE	0,0164	0,0370
8	100	200	100	128	64	EV2-B0	EV2-B0	EV2-B0	0,22	0,22	0,22	0,33	MSE	0,1909	0,0529
9	100	100	200	128	64	EV2-B0	EV2-B0	EV2-B0	0,22	0,22	0,22	0,33	MSE	0,0483	0,0590
10	200	100	100	256	64	EV2-B0	EV2-B0	EV2-B0	0,22	0,22	0,22	0,33	MSE	0,0324	0,0586
11	200	100	100	64	64	EV2-B0	EV2-B0	EV2-B0	0,22	0,22	0,22	0,33	MSE	0,1927	0,0685
12	200	100	100	128	128	EV2-B0	EV2-B0	EV2-B0	0,22	0,22	0,22	0,33	MSE	0,0287	0,0493
13	200	100	100	128	32	EV2-B0	EV2-B0	EV2-B0	0,22	0,22	0,22	0,33	MSE	0,0547	0,0580
14	200	100	100	128	64	EV2-B0	EV2-B0	EV2-B0	0,22	0,22	0,22	0,33	MSE	0,0271	0,0704
15	200	100	100	128	64	EV2-B0	EV2-B0	EV2-B0	0,3	0,15	0,15	0,4	MSE	0,0306	0,0557
16	200	100	100	128	64	EV2-B0	EV2-B0	EV2-B0	0,3	0,15	0,15	0,4	$1,1 \times St^*$	0,0201	<u>0,0390</u>
17	200	100	100	128	64	EV2-B0	EV2-B0	EV2-B0	0,3	0,15	0,15	0,4	$1,1 \times St^*$	0,1397	0,0738
18	200	100	100	128	64	EV2-B0	EV2-B0	EV2-B0	0,22	0,22	0,22	0,33	$1,1 \times St^*$	0,0368	0,0932
19	200	100	100	128	64	EV2-B0	EV2-B0	EV2-B0	0,22	0,22	0,22	0,33	$1,1 \times Sp^*$	<u>0,0171</u>	0,0415
20	200	100	100	128	64	EV2-B0	EV2-B0	EV2-B0	0,22	0,22	0,22	0,33	Özel**	0,0178	0,0439

EV2-B0, dondurulmuş katmanlara sahip EfficientNetV2-B0 anlamına gelir. Kırmızı renk, modelin yalnızca son 5 katmanının eğitilebilir olduğunu gösterir.

* Kayıp sütunundaki ' $1,1 \times St^*$ ' gibi girişler, toplam kaybın ağırlıklı bir toplam olarak hesaplandığını gösterir: $1,1 \times St + 1,0 \times Sp$ (Sp: Hız bileşeni, yani $\mathcal{L}_{hız}$, St: Direksiyon açısı bileşeni, yani $\mathcal{L}_{direksiyon}$, baş tarafta yer alan katsayılar da λ_{dr} ve λ_{hz} 'yi temsil etmektedir). Yazılmayan bileşen 1 ile çarpılmıştır.

** Özel kayıp, (Wan et al., 2021)'de kullanılan bir kayıp fonksiyonudur.

En iyi sonuçlar koyu yazı ile, en iyi ikinci sonuçlar ise altı çizili olarak gösterilmiştir

Test 1'deki gibi derinlik tahmininden faydalanmadan ve semantik segmentasyonu sisteme dahil etmeden, literatürdeki birçok çalışma gibi yalnızca RGB modülü ile elde edilen sonuçlar da yukarıda bahsedilen sonuçlara yakın olacak şekilde daha yüksek hata değerine sahiptir (Buna rağmen elde edilen hata değeri, literatürdeki en başarılı sonucun ardından ikinci sırada yer almaktadır). Ancak her üç alt ağ birlikte ve eşit ÖKU ile kullanıldığında daha başarılı sonuçlar elde edilmiştir. Sadece LSTM kullanılarak elde edilen sonuçlar da (*Test 6*) şaşırtıcı şekilde tüm alt ağların eşit ÖSÜ kullanılarak elde edildiği sonuçlara yakın çıkmıştır. RGB alt ağının ÖKU değeri, 7. testte diğerlerine kıyasla iki katına çıkarılarak 200 olarak tanımlanmıştır. ÖKU oranlarının (2-1-1) olarak alındığı, RGB, derinlik ve segmentasyon modülleri ile LSTM modülünün birlikte kullanıldığı *Test 7*, hem direksiyon açısı hem de hız için en başarılı sonuçları vermektedir. Her bir alt ağın gerekliliği ve sisteme ayrı ayrı değil, birlikte sundukları katkı burada açıkça görülmektedir. Ayrıca bu sonuçlar, ağların sahip olduğu ÖKU değerinin de sonucu doğrudan etkilediğini göstermiştir.

Test 8 ve *Test 9*, derinlik ve segmentasyon modalitelerinin etkisini, ilgili ÖKU değerlerini diğerlerine kıyasla iki katına çıkararak araştırmaktadır. Önemli bir iyileşme elde edilmemesine rağmen, derinlik modülünde yüksek ÖKU değeri seçilmesi ile birlikte direksiyon açısı hatasında önemli bir artış gözlenmiştir (*Test 8*). Yine benzer şekilde derinlik veya anlamsal görüntülere ait ÖKU değerlerinin artırılması hız tahminini de olumsuz etkilemiştir.

Test 10-11, LSTM'in yüksek ve düşük etkiye sahip olduğu durumların; *Test 12-13* ise yoğun katmanların yüksek ve düşük etkiye sahip durumların birleştirme üzerindeki etkisini görmek için yürütülmüştür. Özellikle, sırasıyla LSTM ve yoğun katmanın azaltılmış katkılarını içeren *Test 11* ve *Test 13*, her iki hatada da kayda değer bir artış göstermiştir.

Test 14'te, RGB alt ağının son 5 katmanının eğitilebilir olarak seçilmesi hızı önemli ölçüde olumsuz etkilemiştir. Benzer bir testte, yine RGB alt ağının son 5 katmanının eğitilebilir olarak seçildiği ve ağ ağırlıklarının değiştirildiği durumda, hız hatasında bir önceki teste göre önemli bir düşüş görülürken, direksiyon açısı hatasında hafif bir artış meydana gelmiştir (*Test 15*). *Test 16*'da kullanılan kayıp denklemindeki direksiyon açısının ağırlığının artırılması, eşit ağırlıklı toplam kaybın kullanıldığı bir önceki iterasyona (*Test 15*) kıyasla her iki hatada da performansı artırmıştır. *Test 16*'da kullanılan parametreler ikinci en düşük hız hatasını vermiş ve kayıp denklemi değişikliklerinin faydalı olabileceğini göstermiştir. Tüm katmanların dondurulduğu (en

iyi sonuçların elde edildiği teste benzer şekilde) ancak kayıp denklemindeki direksiyon açısının ağırlığının artırıldığı (burada alt ağ ağırlıkları da farklı) *Test 17*'de hem hız ve hem de direksiyon açısı için RMSE değerinin arttığı gözlemlenmiştir. RGB alt ağının ve kayıp denklemindeki direksiyon açısının ağırlığının ikisinin birden artırılmasının, direksiyon açısı tahminini çok olumsuz etkilediği görülmüştür. Daha sonra kayıp denklemindeki direksiyon açısının ağırlığının korunduğu ancak alt ağ ağırlıklarının eski duruma geri döndüğü *Test 18*'de direksiyon açısı hatasının ciddi oranda azaldığı ancak yine de en başarılı testten daha yüksek çıktığı, hız hatasının ise *Test 17*'ye göre arttığı gözlemlenmiştir. Hız kaybının toplam kayıp içinde ağırlıklandırılması hız hatasını önemli ölçüde azaltırken, direksiyon açısı hatasında ikinci en iyi sonucu vermiştir (*Test 19*). *Test 19*'da da alt ağların ağırlıkları eşit alınmıştır. Son olarak, MSE'ye dayalı hız kaybıyla birlikte direksiyon açısı için özel bir kayıp denkleminin (Wan et al., 2021) kullanılması, ikinci en düşük direksiyon hatasına yakın bir hata sağlasa da, Udacity veri kümesinde hem hız hem de direksiyon açısı için en iyi performansı gösteren *Test 7*'nin sonuçlarını geçememiştir (*Test 20*).

Burada çok daha fazla parametre değişikliği ve ince ayar çalışması yapılarak daha iyi sonuçlar elde etme yoluna gidilebilir. Ancak her test sonucu için aynı eğitim ve test sürecinin beş defa yapıldığı göz önünde bulundurulursa toplam test sayısının çok yüksek rakamlara çıkacağı açıktır.

5.3 Sonuçlar ve Analiz

Bu tez çalışması ile elde edilen sonuçlar, önerilen modelin direksiyon açıları ve hız tahminleri konusunda farklı veri kümelerinde ve farklı sürüş koşullarında güçlü performans sergilediğini göstermektedir. Udacity veri kümesi, çeşitli gün ışığı zamanlarında ve değişik arazi koşullarında toplanmış geniş bir görüntü yelpazesini içermektedir. Bu veri kümesi, farklı trafik yoğunluklarına, çeşitli yol tiplerine ve değişen aydınlatma koşullarına dayalı olarak oluşan karmaşık yol senaryolarını içermektedir. Bu durum, bazen aşırı parlaklık veya gölgelenme gibi zorlu görsel engellerin de ortaya çıkmasına neden olmaktadır. Ayrıca, veri kümesi değişken trafiğe sahip sahneleri ve farklı yol türlerindeki çeşitli dönüşleri de içermektedir. Önerilen model, bu zorlu koşullara rağmen literatürdeki diğer yöntemlerden daha üstün sonuçlar elde ederek dikkate değer bir performans sergilemektedir.

CARLA sürüş simülatörü ile oluşturulan CARLA veri kümesi de çeşitli aydınlanma zamanlarında ve çeşitli hava koşullarında toplanmıştır. Bu yüzden veri çeşitliliği açısından oldukça zengin bir yapıya sahiptir. Bu çeşitlilik, elde edilen sonuçların genellenebilirliğini artırarak modelin eğitim sürecindeki performansını geliştirmiştir. Farklı aydınlanma zamanları ve hava koşullarını içeren bu geniş veri yelpazesi, modelin çeşitli senaryolardaki performansını değerlendirmesine olanak tanımıştır. Bu zengin veri kümesi, modelin değişken koşullara karşı daha sağlam ve esnek bir performans sergilemesine katkıda bulunmuştur. Bu zenginlik aynı zamanda, modelin eğitim sürecinde karşılaşılabileceği gerçek dünya senaryolarını daha iyi kavramasına ve genelleme yeteneğini güçlendirmesine yardımcı olmuştur.

Ayrıca, eğitim, doğrulama ve test kümeleri farklı kasabalarda toplandığı için eğitim verisi ile test edilen kasabanın yol yapısı, trafik koşulları ve çevresel unsurlar ve binaları oldukça farklılık göstermektedir. Bu durum, CARLA ile elde edilen sonuçların önemini artırmaktadır. Mevcut literatürde birçok derin öğrenme uygulaması, aynı ortamda toplanan verinin eğitim, doğrulama ve test olarak ayrılması esasına dayanır. Bu nedenle, modelin test sırasında karşılaşacağı veri türü, eğitim esnasında gösterilen veriyle benzer özellikler içerse bile, elde edilen sonuçların güvenilirliği sorgulanabilir. Benzer özelliklere sahip veri ile eğitilen modeller, test aşamasında da benzer veriyle başarılı sonuçlar elde edebilirler.

Bu nedenle CARLA ile yapılan çalışmada farklı kasabalarda eğitilen modelin, hiç eğitim verisi toplanmamış bir kasabada test edildikten sonra literatürdeki en başarılı tahmin sonuçlarını üretmesi, modelin genelleme yeteneği ve esnekliği açısından oldukça önemlidir.

Önerilen model, Sully Chen veri kümesi üzerinde de test edilmiştir. Sully Chen veri kümesi, ön kameradan toplanan görüntülerle birlikte yalnızca direksiyon açısı bilgisini içermekte olup hız verisi içermemektedir. Bu veri kümesi üzerinde elde edilen sonuçlar, literatürdeki diğer Sully Chen uygulamalarıyla karşılaştırılmış ve en başarılı sonuçların bu çalışmada elde edildiği tespit edilmiştir.

Modelin Sully Chen veri kümesi üzerindeki performansı, yalnızca direksiyon açısına dayalı tahminlerdeki doğruluğu göstermesi açısından önemlidir. Bu veri kümesi, özellikle hız bilgisinin olmaması sebebiyle modelin sadece yönlendirme açısından nasıl bir performans gösterdiğini değerlendirmek için önemli bir fırsat sunmaktadır. Bu değerlendirme, modelin yalnızca görsel veriye dayalı olarak ne kadar doğru ve güvenilir bir şekilde direksiyon açısı tahmin edebildiğini belirlemek açısından kritiktir.

Bu çalışmada önerilen model, aracın anlık direksiyon açısı ve hızını yalnızca bir monoküler kamera görüntüsü kullanarak tahmin etmektedir. Uçtan uca (end-to-end) olarak adlandırılan bu yöntemde, algılama ile üretilen kontrol çıktıları arasında tek bir kapalı kutu sistem vardır. Geleneksel sistemlerde, genellikle bir dizi aşamalı işlem bulunur. Algılamadan sonra, yerel işlem birimleri, sensör verileri üzerinde işlem yapmak ve ardından bir dizi işlem sonucu üretilen çıktılar kontrol edilir. Bu aşamaların her biri genellikle ayrı ayrı ayarlanır ve optimize edilir. Buna karşılık, uçtan uca sistemlerde, bu aşamalı süreçler tek bir entegre yapı içinde birleştirilir. Bu sistemler, giriş verilerini doğrudan çıktıya dönüştüren karmaşık bir model veya ağ tarafından yönetilir. Bu sayede, daha az insan müdahalesi gerektiren ve genellikle daha az el ile ayar gerektiren bir yapı oluşur. Bu durum, genellikle daha hızlı ve daha tutarlı sonuçlar elde etme potansiyeli doğurur.

5.3.1 Direksiyon açısı tahmini

Direksiyon açısı tahmini, önerilen modelin iki ana çıktısından biridir. Model, çeşitli sürüş koşullarında ve farklı yol senaryolarında görüntü üzerindeki özellikleri analiz ederek aracın direksiyon açısını doğru bir şekilde tahmin etmek için eğitilmiştir. Bu çalışmada, direksiyon açısı tahmini monoküler kamera görüntüsü üzerinden gerçekleştirilmektedir. Bu tahminler, uçtan uca sistemin bir parçası olarak gerçekleştirilir ve farklı alt ağlar içeren bir ağ yapısı kullanılarak elde edilen sonuçlar, modelin genel performansının göstergesidir.

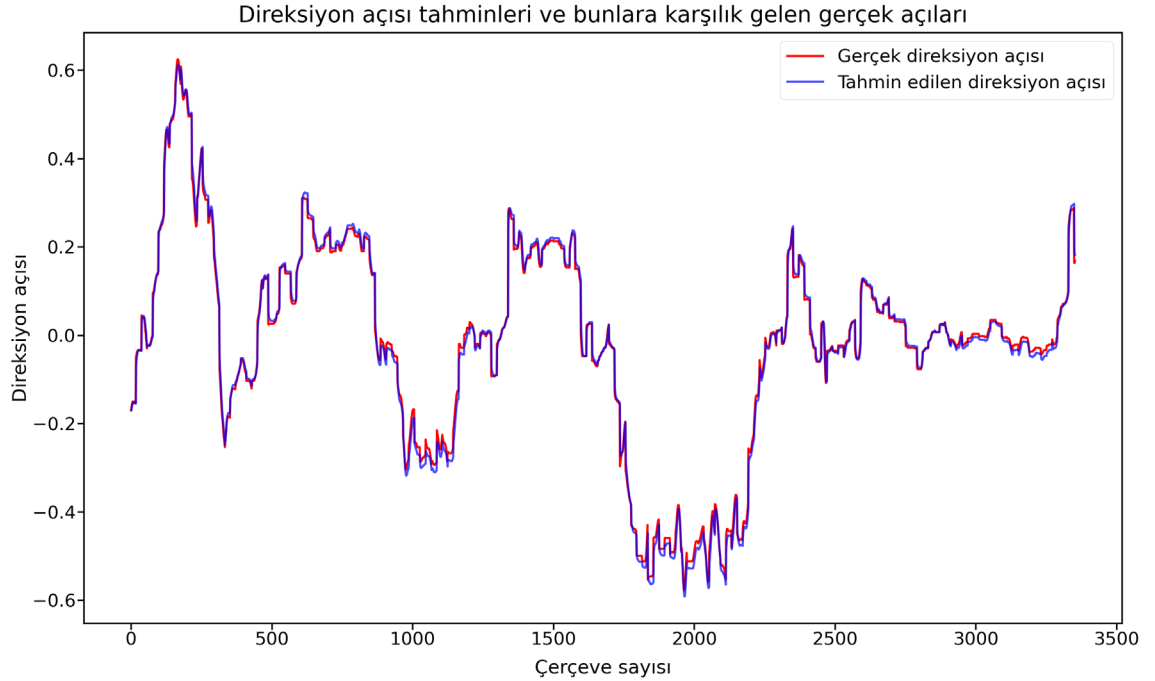
Sürüş anında aracın (kameranın) ön görüş alanında her zaman kusursuz bir sürüş sahnesi bulunmaz. Bazen aydınlatma, çevredeki diğer araçlar ve yolun yapısı (düz yol veya keskin bir şekilde dönen) gibi nedenlerden dolayı algılama zorlaşır. Bu gibi durumlarda yalnızca tek bir girdiden bilgi sağlamaya çalışmak yerine, sistemin algısını artıracak farklı yapıda bilgilere de ulaşmak sistemin daha sağlıklı sonuç üretmesini sağlayacaktır. Genellikle bu iyileştirme çalışmaları literatürde farklı tipte sensörler eklenerek yapılmaya çalışılmıştır. Ancak bu çalışmada, tek bir sensör kullanarak farklı yapıda giriş verileri oluşturmaya çalışılmıştır. RGB kamera görüntüsüne ilaveten, bu görüntüden elde edilen derinlik görüntüsü ve semantik segmentasyon görüntüsü elde edilmiştir. Böylece sisteme sanki üç sensörden bilgi geliyormuş gibi üç farklı giriş oluşturulmuştur. Bunun amacı ise sistemin, yukarıda bahsedilen zorlu sürüş şartlarında

bir girişten sağlayamadığı bilgiyi diğer girişlerden sağlamaya çalışmasıdır. Böylece araç kesintisiz olarak başarılı şekilde direksiyon açısı üretebilecektir.

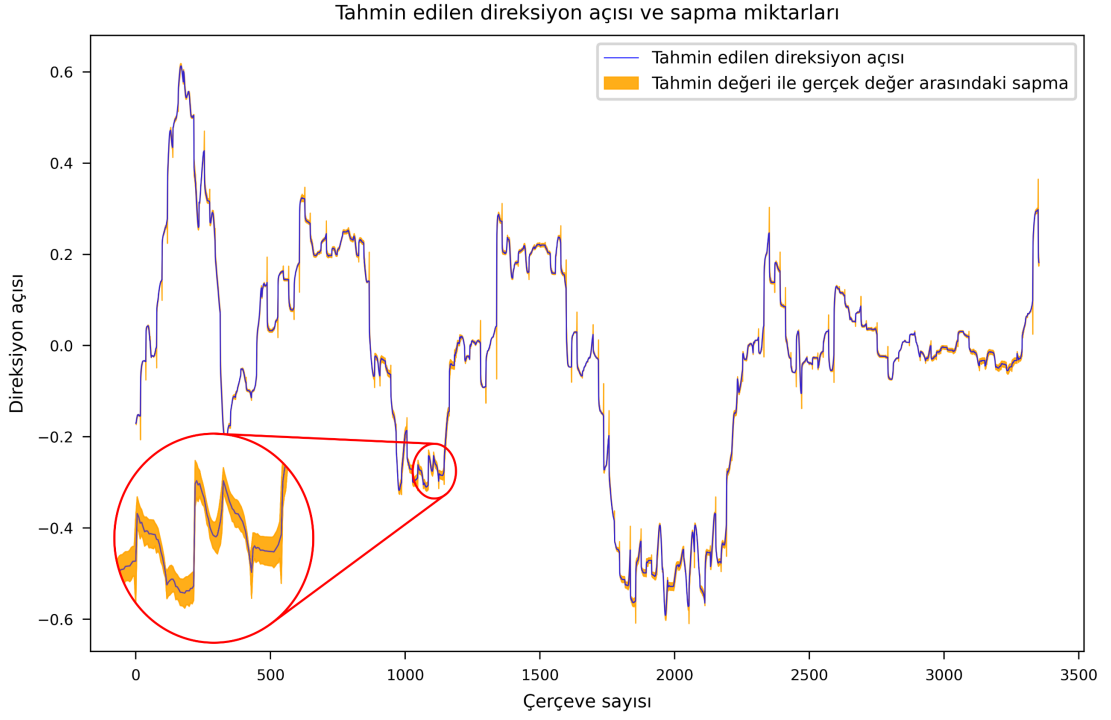
5.3.1.1 Udacity veri kümesi ile elde edilen direksiyon açısı tahminleri

Udacity veri kümesi ile elde edilen direksiyon açısı tahmin sonuçları Şekil 5.2’de görsel olarak sunulmaktadır. Mavi eğri model tarafından tahmin edilen direksiyon açılarını temsil ederken, kırmızı eğri gerçek veriyi göstermektedir. Bu grafiksel sunum, modelin sahip olduğu doğruluk ve güvenilirliği göstermektedir. Model, direksiyon açısını öngörmek için kullanılan verilere dayanarak, aracın dönüşlerini ve yön değişikliklerini oldukça doğru bir şekilde tahmin edebilmektedir. Özellikle, tek bir kamera kullanılarak elde edilen verilere dayanarak bu derecede kesin sonuçlar elde edilmesi, modelin gerçek dünya koşullarında da etkili bir şekilde çalışabileceğini ve direksiyon davranışlarını tahmin etmede güçlü bir yetkinliğe sahip olduğunu göstermektedir.

Bu sonuçlar, özellikle otonom sürüş sistemleri gibi alanlarda, araçların çevresel koşullara uyum sağlaması ve güvenli bir şekilde hareket etmesi açısından büyük önem taşımaktadır. Modelin başarılı tahmin yeteneği, sürücüsüz araçlar ve ileri düzey sürüş destek sistemlerinin geliştirilmesinde önemli bir adım olarak değerlendirilebilir.



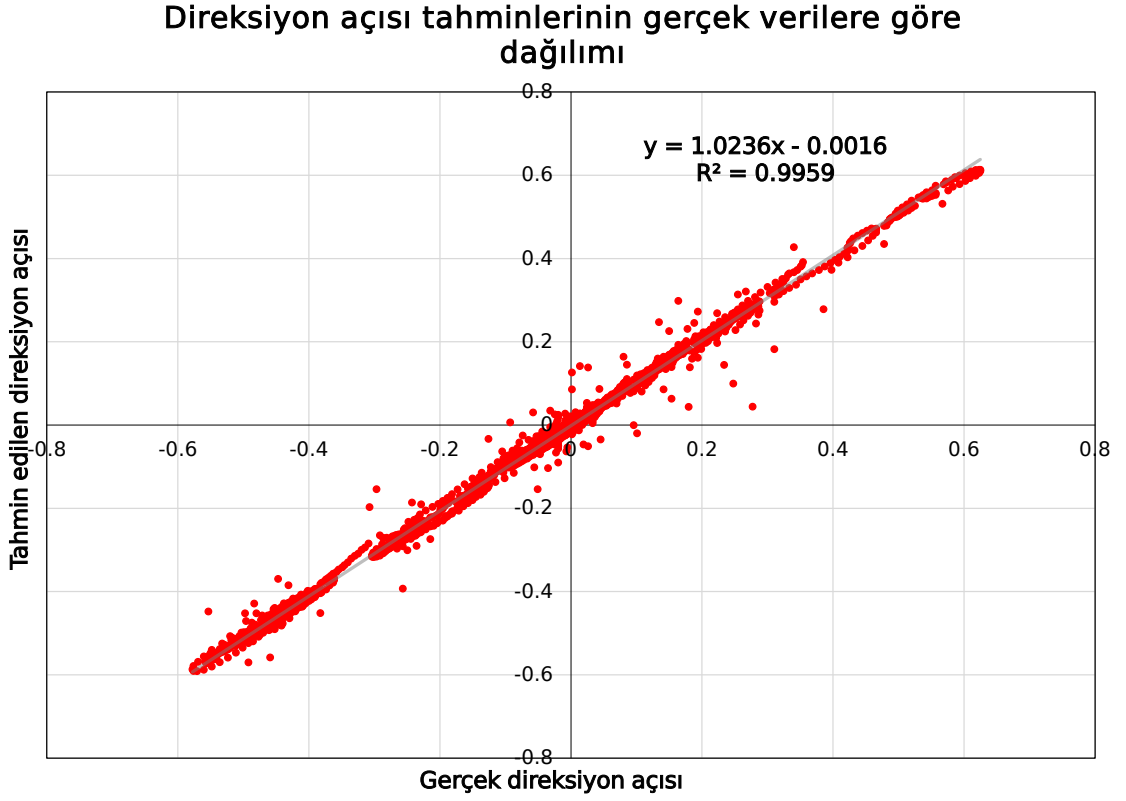
Şekil 5.2 Udacity veri kümesi üzerinde direksiyon açısı tahminlerinin sonuçları ve bunlara karşılık gelen gerçek açıları. Direksiyon açısı tahminleri mavi renkle gösterilirken, gerçek direksiyon açısı değerleri kırmızı renkle gösterilmiştir



Şekil 5.3 Udacity veri kümesi üzerinde tahmin edilen direksiyon açıları ve her tahmin sırasında oluşan sapma miktarları. Direksiyon açısı tahminleri mavi çizgi ile gösterilirken, sapma miktarları sarı renkli bant ile gösterilmiştir

Şekil 5.3, her bir görüntü için yapılan direksiyon açısı tahminlerinin gerçek değerlerden sapmasını göstermektedir. Grafikte, x-ekseni her bir görüntüye karşılık gelirken, y-ekseninde ise direksiyon açısı bulunmaktadır. Her bir veri noktası, bir görüntünün modele sunulmasıyla elde edilen tahmini ve bu tahminin gerçek değerden ne kadar farklı olduğunu ifade etmektedir. Burada tahmin edilen direksiyon açıları mavi ile gösterilirken, sapmalar sarı alanla gösterilmiştir. Bu sarı alanın genişliği sapmanın miktarını göstermektedir. Daha iyi anlaşılması açısından bir kısımdaki sapma, grafik üzerinde ayrıntılı olarak gösterilmiştir. Burada bazı yerlerde sapma miktarı fazla iken bazı yerlerde sapmanın düştüğü görülmektedir. Yine grafikte, direksiyon açısının zıt yöne değişmeye başladığı noktalarda, yani aracın istikametini değiştirmeye başladığı yerlerde, direksiyon açısının çok kısa süreli olarak fazla miktarda saptığı ancak hemen bu durumu toparlayarak daha düşük sapmalarla bu dönüşü başarılı bir şekilde yaptığı görülmektedir.

Bu grafik, modelin genel performansını incelemek ve belirli görüntülerdeki tahmin hatalarını görsel olarak analiz etmek için kullanılmıştır. Modelin her bir görüntü için ne kadar doğru tahmin yaptığı veya hangi durumlarda hatalı olduğu, bu grafik yardımıyla detaylı bir şekilde incelenmiştir.



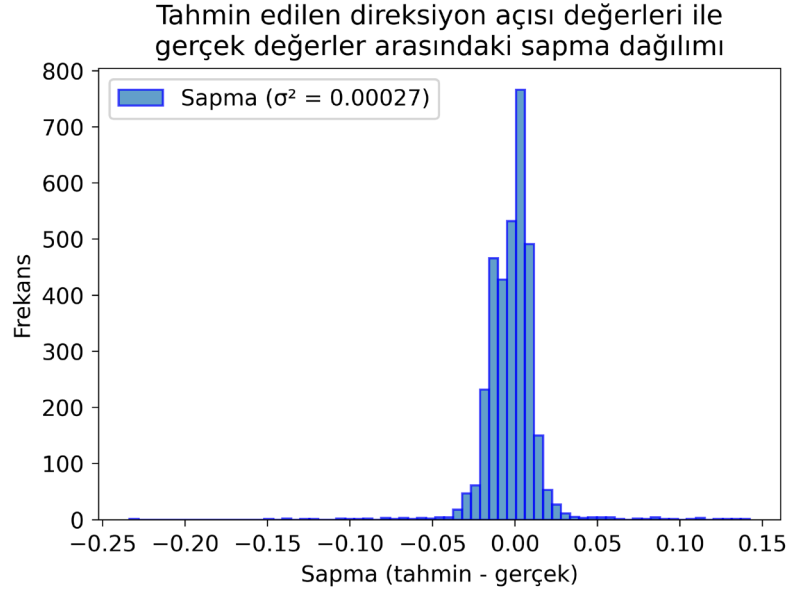
Şekil 5.4 Direksiyon açısı tahminlerinin gerçek verilere göre dağılımı

Şekil 5.4'te ise sapma değerlerinin dağılımı gösterilmiştir. Sapma değerlerinin dağılımı, modelin gücünü ve zayıflıklarını anlamak, eğitim sürecini geliştirmek için çalışma sırasında da kullanılmıştır. Elde edilen regresyon denklemi ve ortaya çıkan R^2 değeri, modelin gücünü ve elde edilen sonuçların sağlamlığını vurgulamaktadır. Şekilde ayrıca her iki yöne dönüşte de az miktarda oluşan aykırı değerler görülmektedir.

Sapma miktarının frekans bazlı dağılımı ise Şekil 5.5'te sunulmuştur. Grafikte, x-ekseni tahmin edilen ve gerçek değerler arasındaki farkı, y-ekseni ise bu farkın frekansını ifade etmektedir. Ayrıca grafik, $\sigma^2=0,00027$ varyans değeri ile sapmaların dağılımı hakkında bilgi vermektedir.

Grafiğe bakıldığında sapmaların büyük çoğunluğunun $(-0,05-0,05)$ aralığında küçük değerlerde yoğunlaştığı görülmektedir. Bu durum, tahminler ile gerçek değerler arasındaki farkların genellikle küçük olduğunu göstermektedir. Ayrıca, oldukça düşük σ değeri, sapmaların ne kadar sınırlı olduğunu göstermekte ve tahminlerin genellikle gerçek verilere yakın olduğunu vurgulamaktadır. Sapma dağılım grafiğinde, negatif ve pozitif sapmaların miktarı benzerlik göstermektedir. Negatif tarafta daha fazla ya da pozitif tarafta daha fazla sapmalar gözlemlenmemiştir; bunun yerine, her iki taraftaki sapmaların

sayısal açıdan benzer olduğu görülmüştür. Bu durum sapmaların dengeli bir şekilde dağıldığını, negatif ya da pozitif tarafa meyilli tahminler yapılmadığını göstermektedir.



Şekil 5.5 Frekans bazında direksiyon açısı sapma miktarının dağılımı

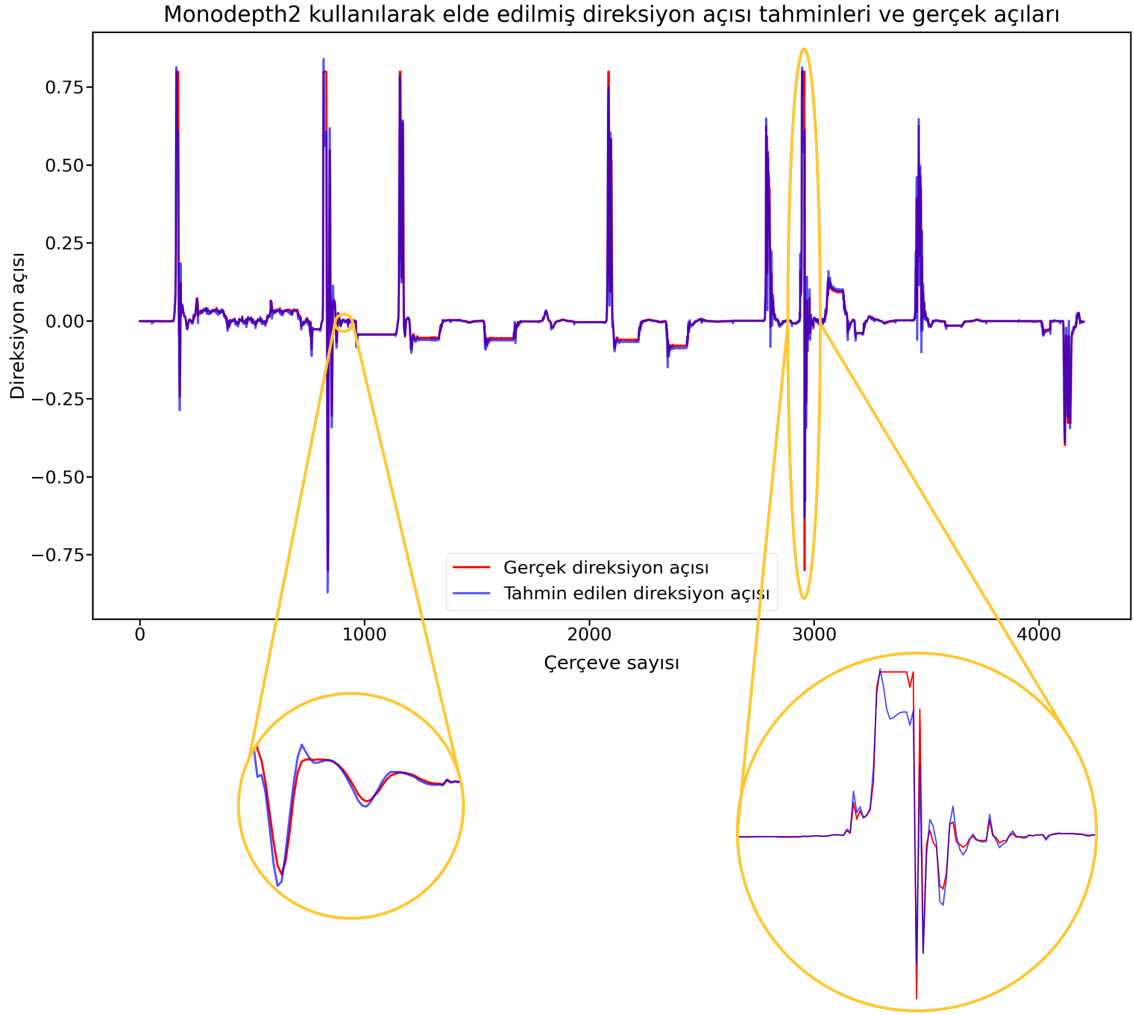
Bu grafik, derin öğrenme modelinin tahminlerinin yüksek doğrulukta olduğunu ve gerçek verilere uyum sağladığını göstermektedir.

5.3.1.2 CARLA veri kümesi ile elde edilen direksiyon açısı tahinleri

Bölüm 4.2.2’de bahsedildiği gibi CARLA veri kümesi oluşturulurken eğitim verisi *Town01*, *Town02* ve *Town03* kasabalarından toplanmışken, doğrulama verisi *Town04*’ten, test verisi ise *Town05*’ten toplanmıştır. Eğitim ve test verileri birbirinden çok farklı ortamlarda toplanmış olmasına rağmen, modelin direksiyon açısı tahminleri oldukça başarılı olmuştur.

CARLA sürüş simülatörünün avantajlarından biri de sensör bakımından oldukça zengin oluşudur. CARLA, kamera görüntüsüne ilaveten derinlik görüntüsü de verebildiğinden RGB görüntülerin yanında derinlik görüntüleri de toplanmıştır. Böylece hem CARLA’nın ürettiği derinlik görüntüleri kullanılarak, hem de önerilen modelin kendi derinlik tahmin algoritması olan Monodepth2 kullanılarak elde edilen derinlik görüntüleri ile direksiyon açısı tahmini yapılmıştır.

CARLA veri kümesi üzerinde Monodepth2 kullanılarak elde edilen direksiyon açısı tahmin sonuçları Şekil 5.6’da görsel olarak sunulmaktadır.

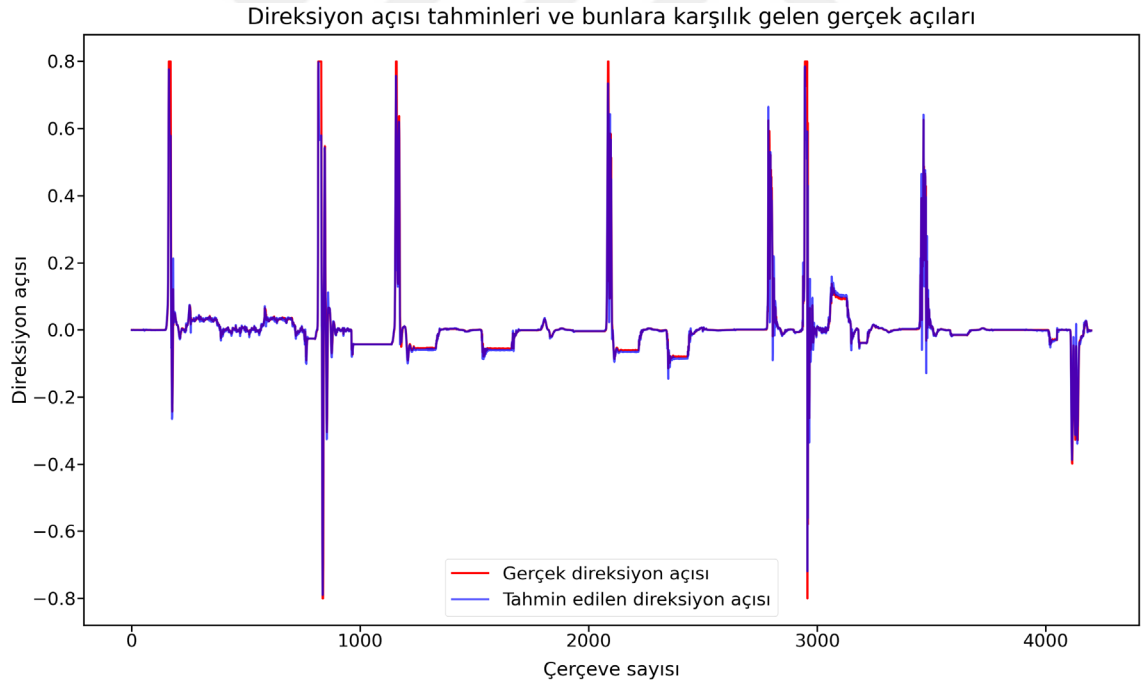


Şekil 5.6 CARLA veri kümesinde Monodepth2 kullanılarak elde edilmiş direksiyon açısı tahminleri

CARLA’da *Town05* kasabasında yapılan testler, önerilen modelin son derece başarılı olduğunu göstermiştir. Bu testlerde, derinlik algoritması olarak Monodepth2 kullanılan modelin direksiyon açısı tahminleri, gerçek değerlerle oldukça yakın bir uyum içindedir. Grafikselsel olarak incelendiğinde, tahmin edilen direksiyon açıları ile gerçek değerler arasında neredeyse tam bir örtüşme görülmektedir. Bu sonuçlar, modelin *Town05*’te gerçekleştirilen testlerde büyük bir doğrulukla çalıştığını ortaya koymaktadır. Ayrıca, bu başarının sayısal ifadesi olan RMSE değeri oldukça düşük olup, modelin tahminlerinin gerçek değerlere çok yakın olduğunu desteklemektedir (Tablo 5.3). CARLA veri kümesi ile elde edilen sonuçlar, literatürde direksiyon açısı sunan çalışmalar arasında en başarılı sonuçlar olmuştur. *Town05*’teki testler, önerilen modelin CARLA

veri seti üzerinde üstün bir performans sergilediğini ve direksiyon açısı tahmininde yüksek doğruluk sağladığını açıkça göstermektedir. Udacity veri kümesinde sonra, eğitim kümesi ve test kümesi tamamen ayrı kasabalarda toplandığı için farklı yapıda olan CARLA veri kümesi ile de başarılı sonuçlar elde edilmiş olması ayrıca önemlidir.

Geliştirilen model, derinlik tahmini için kullanılan Monodepth2 yöntemi yerine bir de CARLA'nın ürettiği derinlik görüntüleri kullanılarak test edilmiştir. Burada derinlik görüntüsü kaynağının değişmesi dışında bütün parametreler önceki testle aynı şekilde kullanılmıştır. Şekil 5.7'de görüldüğü gibi elde edilen sonuçlar, Monodepth2 kullanılarak elde edilen sonuçlara oldukça yakındır. Burada da tahmin edilen direksiyon açıları ile gerçek değerler arasında neredeyse tam bir örtüşme görülmektedir. Bu sonuçlar, modelde kullanılan monoküler derinlik tahmin yönteminin (Monodepth2), CARLA'nın ürettiği derinlik görüntüleri kadar kullanışlı derinlik tahmini ürettiğini ortaya koymaktadır. Her iki yöntemle elde edilen derinlik görüntüleri Şekil 4.10'da görülmektedir.



Şekil 5.7 Monodepth2 derinlik tahmin yöntemi yerine CARLA sürüş simülatörünün sağladığı derinlik görüntüleri kullanılarak elde edilmiş direksiyon açısı tahminleri

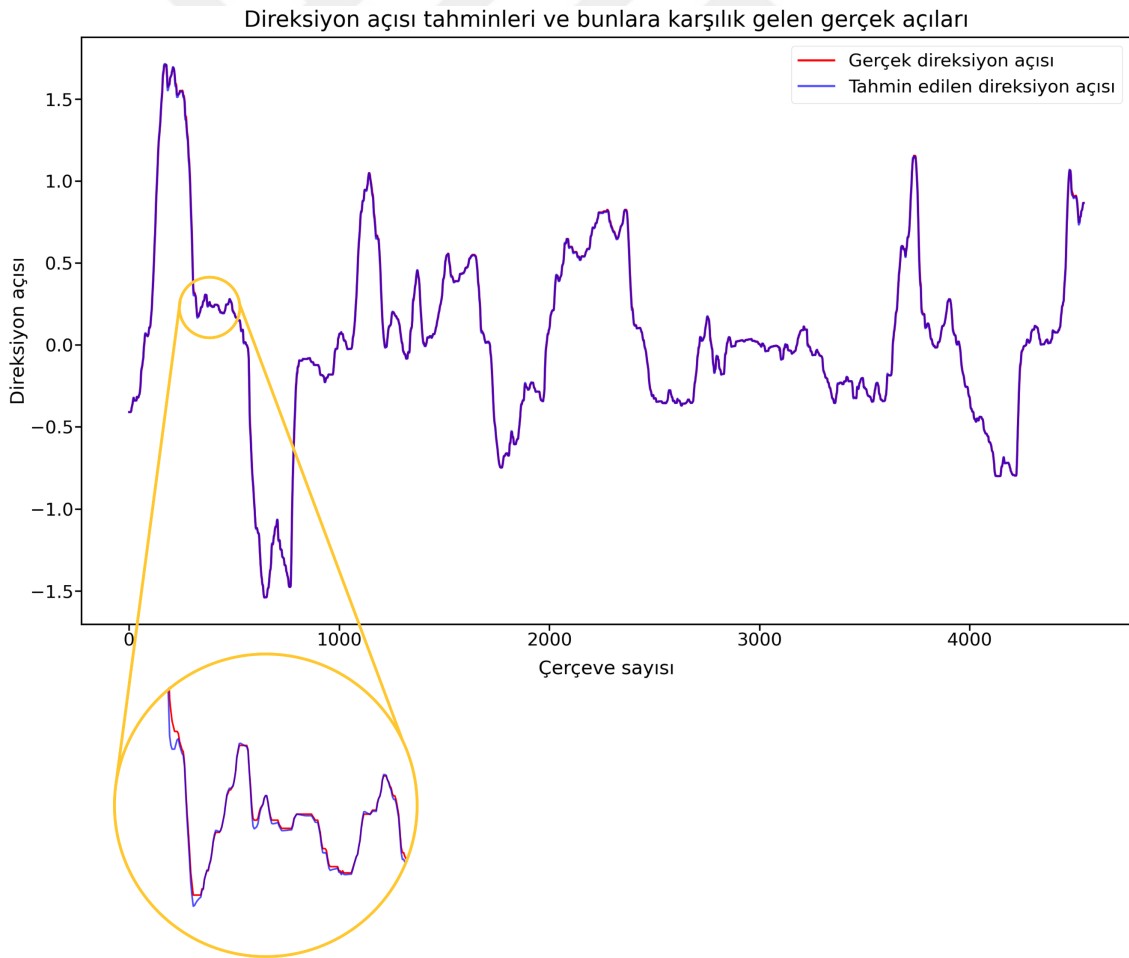
5.3.1.3 Sully Chen veri kümesi ile elde edilen direksiyon açısı tahinleri

Sully Chen veri kümesi, direksiyon açısı tahmini üzerine odaklanmış bir veri kümesidir. Udacity ve CARLA veri kümeleri ile elde edilen sonuçları pekiştirmek ve

farklı bir veri kümesi ile daha modelin direksiyon açısı tahminindeki başarısını doğrulamak amacıyla Sully Chen veri kümesi kullanılmıştır.

Bu veri kümesi de Udacity veri kümesinde olduğu gibi bilgisayarlı görü açısından zorlu olan güçlü ışıklar veya koyu gölgeler içerir. Sürüş verisinin büyük bir çoğunluğu şehir dışı olarak tanımlanabilecek yollarda, küçük bir bölümü ise şehir içinde kaydedilmiştir. Görüntülerde devamlı olarak aktif bir trafik mevcuttur ve yolların büyük bir kısmı gidiş-geliş şeklindedir.

Sully Chen veri kümesi ile elde edilen direksiyon açısı tahminleri ve gerçek değerler ile karşılaştırılması Şekil 5.8’de görülmektedir. Şekilde görüldüğü gibi modelin Sully Chen test verisi üzerindeki tahminleri çok başarılıdır. Tahmin edilen direksiyon açısı değerleri gerçek direksiyon açıları ile bire bir örtüşmektedir. Karşılaştırmalı sonuçlar bölümünde de anlatılacağı gibi önerilen model ile, literatürdeki en düşük hatalı sonuçlar elde edilmiştir.



Şekil 5.8 Sully Chen veri kümesinde elde edilmiş direksiyon açısı tahminleri

5.3.2 Hız tahmini

Önerilen çoklu görev modeli, zorlu sürüş koşullarında ve farklı yol senaryolarında kamera görüntüsü üzerinden direksiyon açısı tahminine benzer şekilde, aracın hızını da başarılı bir şekilde tahmin etmektedir. Bu çalışmada, araç hızı tahmini monoküler kamera görüntüsünden elde edilen özellikler ve zamansal veri yardımıyla elde edilmiştir. Bu tahminler, uçtan uca sistemin bir parçası olarak direksiyon açısı ile birlikte çoklu görev modeli kapsamında eş zamanlı olarak tahmin edilmektedir.

Araç hızını belirleyen en önemli faktörler yolun düzlüğü veya kavisli oluşu ile aracın ilerlediği trafik koşullarıdır. Bunun yanı sıra aracın önceki seyir hızlarına dair geçmiş verileri, tahmini hız hakkında önemli bir bilgi sunar. Gerçek dünyada senaryolarında, düz yollarda daha yüksek hızlara ulaşılabilirken, kavisli yollarda hız, kavis derecesine bağlı olarak daha düşük olabilir.

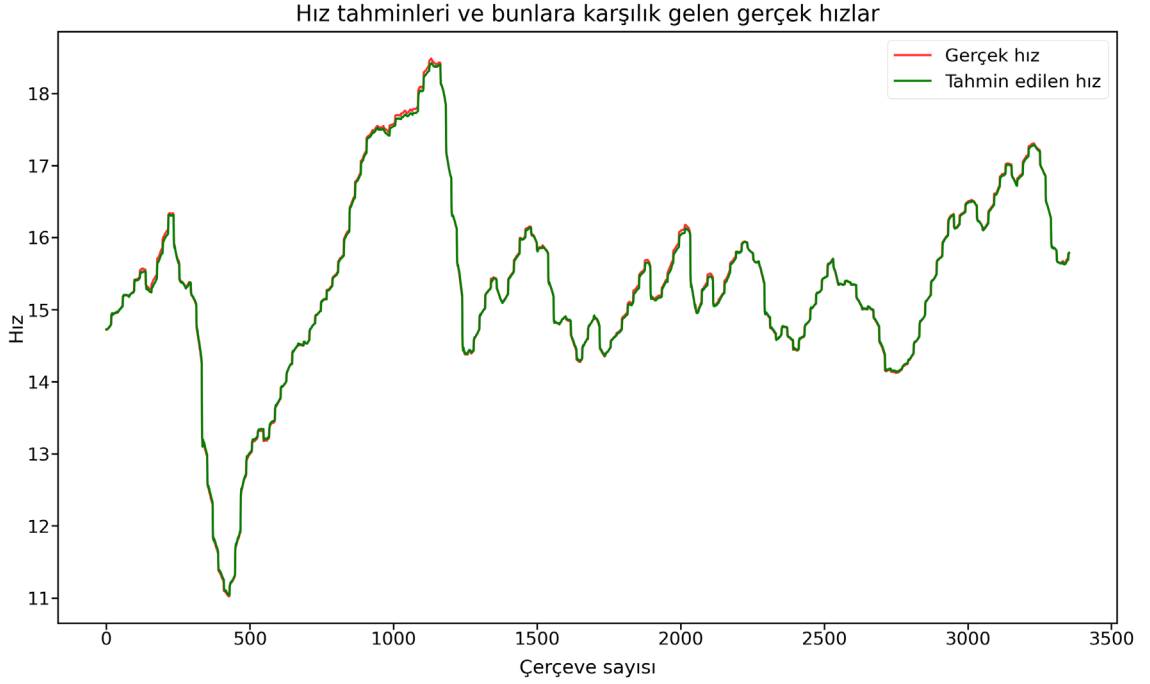
Bir aracın, önündeki araçla aynı anda bir virajı dönme durumu, hızın daha karmaşık bir biçimde yönetilmesini gerektirir. İnsan sürücülerin bu tür durumlarla başa çıkabilmesi, benzer senaryoları defalarca yaşamış ve tecrübe etmiş olmalarına bağlıdır. Derin öğrenme modeli de benzer bir yaklaşımı benimser; eğitim sırasında bu tür senaryoları içeren çeşitli verilerle beslenirler. Model, farklı yol koşulları altında elde edilen verilerle eğitilir ve benzer durumlarla başa çıkma yeteneğini geliştirir. Bu nedenle, modele sunulan eğitim verisinin çeşitliliği son derece önemlidir.

Bu tez çalışması kapsamında kullanılan veri kümelerinden Udacity ve CARLA veri kümeleri, görüntülerle birlikte hız verisi sağlarken Sully Chen veri kümesinde hız verisi bulunmamaktadır. Bu yüzden aşağıdaki bölümlerde, Udacity ve CARLA veri kümelerinde elde edilmiş hız tahminleri sunulacaktır.

5.3.2.1 Udacity veri kümesi ile elde edilen hız tahminleri

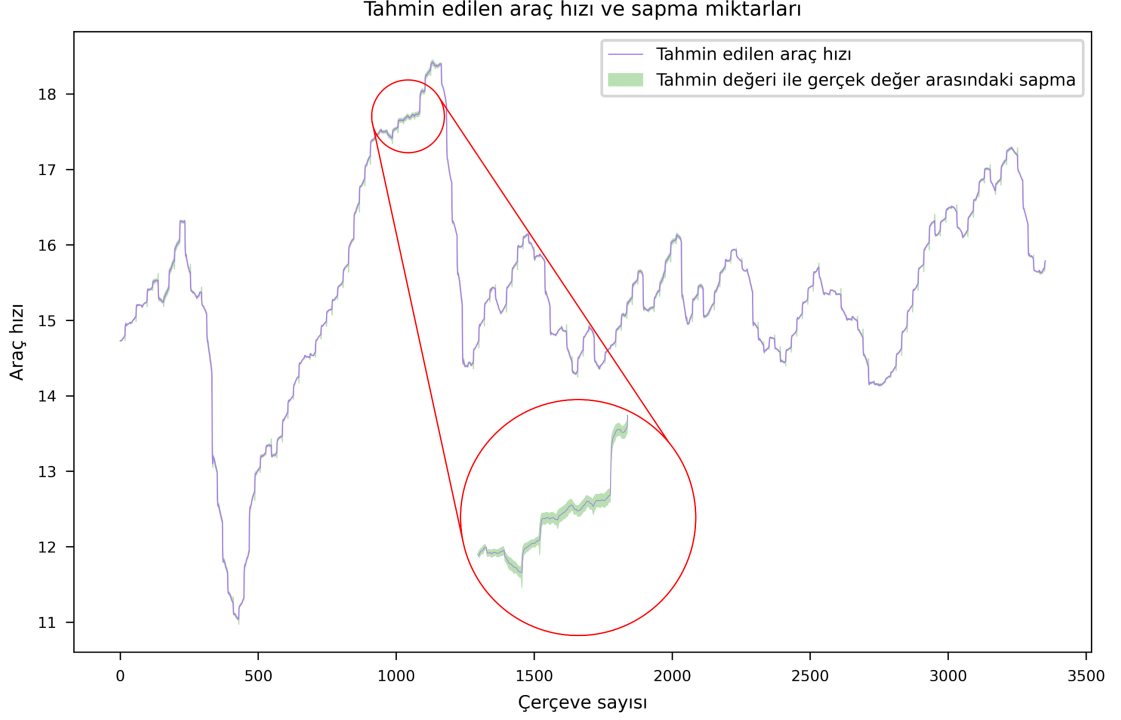
Udacity veri kümesi üzerinde yapılmış hız tahmin sonuçları, Şekil 5.9'da görsel olarak sunulmaktadır. Yeşil eğri model tarafından tahmin edilen araç hızlarını temsil ederken, kırmızı eğri gerçek hız değerlerini göstermektedir. Bu grafiksel sunum, modelin sahip olduğu doğruluk ve güvenilirliği göstermektedir. Model, araç hızını öngörmek için kullanılan verilere dayanarak, aracın oldukça başarılı bir şekilde tahmin edebilmektedir. Şekilde de görüldüğü gibi, yaklaşık 3500 görüntü boyunca araç hızı sabit kalmamış daima değişken olmuştur. Hızın istikrarlı bir şekilde değiştiği yerler dışında ani değişiklikler

yaptığı yerlerde (artarken birden azalmaya başlaması veya tam tersi) dahi modelin tahminleri oldukça başarılıdır. Tek bir monoküler kamera kullanılarak elde edilen verilere dayanarak bu derecede başarılı sonuçlar elde edilmesi, modelin gerçek dünya koşullarında da etkili bir şekilde çalışabileceğini ve hız davranışlarını tahmin etmede güçlü bir yetkinliğe sahip olduğunu göstermektedir.



Şekil 5.9 Udacity veri kümesi üzerinde elde edilmiş hız tahminleri ve bunlara karşılık gelen gerçek hız değerleri. Hız tahminleri yeşil renkle, gerçek hız değerleri ise kırmızı renkle gösterilmiştir

Şekil 5.10 ise, her bir görüntü için yapılan araç hızı tahminlerinin gerçek değerlerden sapmasını göstermektedir. Grafikte, x-ekseni her bir görüntüye karşılık gelirken, y-ekseninde ise hız bulunmaktadır. Her bir veri noktası, bir görüntünün modele sunulmasıyla elde edilen tahmini ve bu tahminin gerçek değerden ne kadar farklı olduğunu ifade etmektedir. Burada tahmin edilen hız mavi ile gösterilirken, sapmalar yeşil alanla gösterilmiştir. Bu yeşil alanın genişliği sapmanın miktarını göstermektedir. Daha iyi anlaşılması açısından bir kısımdaki sapma, grafik üzerinde ayrıntılı olarak gösterilmiştir. Burada da görüldüğü gibi değişken hız durumlarına rağmen sapma daima düşük kalmaktadır.



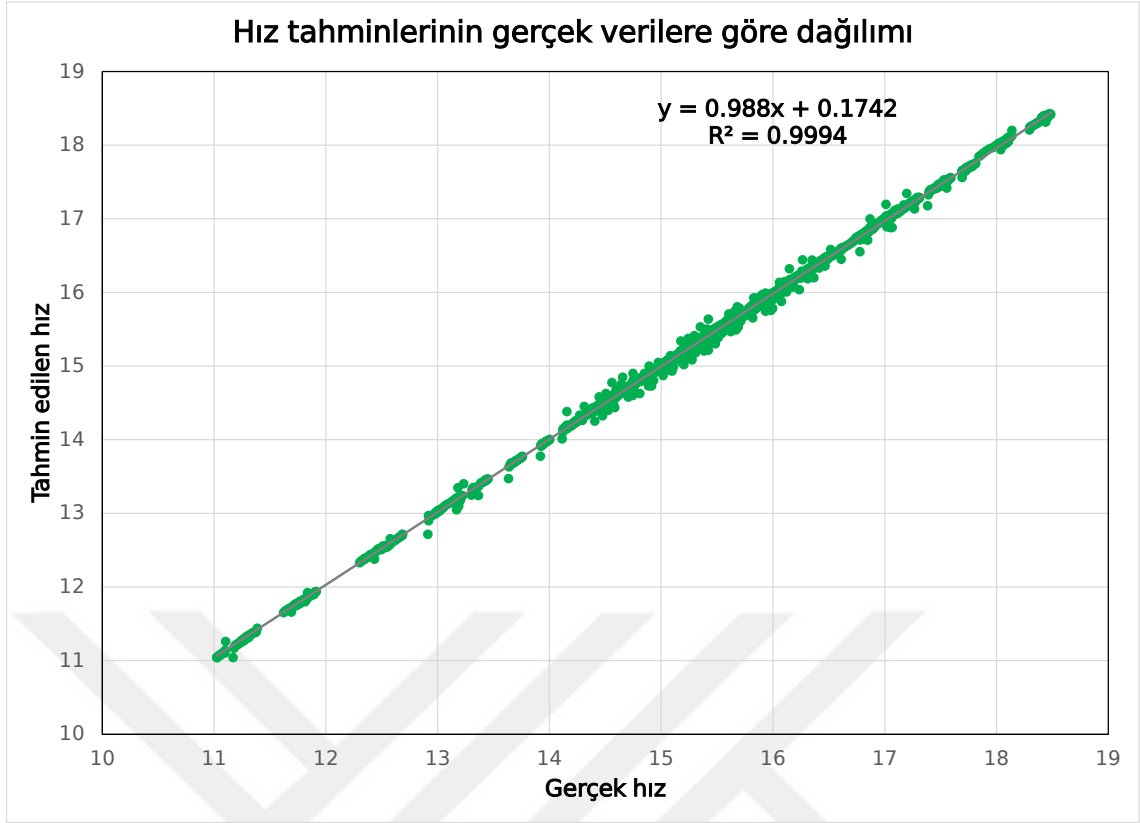
Şekil 5.10 Tahmin edilen araç hızı sonuçları ve her tahmin sırasında oluşan sapma miktarları. Araç hızı tahminleri mavi renkle gösterilirken, sapma miktarları yeşil renkle gösterilmiştir

Bu grafik, modelin hız performansı konusundaki genel performansını incelemek ve belirli görüntülerdeki tahmin hatalarını görsel olarak analiz etmek için kullanılmıştır. Modelin her bir görüntü için ve her türlü hız değişikliği durumunda dahi ne kadar doğru tahmin yaptığı grafikte açıkça görülmektedir.

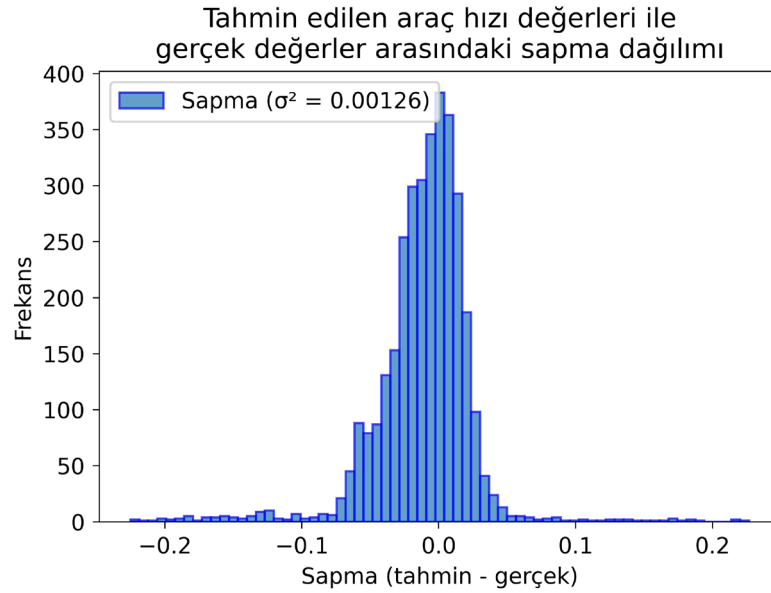
Şekil 5.11’de ise hızla ilgili sapma değerlerinin dağılımı gösterilmiştir. Araç hızına ait sapma değerlerinin dağılımı da modelin gücünü ve zayıflıklarını anlamak, eğitim sürecini geliştirmek için eğitimler sırasında kullanılmıştır. Elde edilen regresyon denklemi ve ortaya çıkan R^2 değeri, modelin gücünü ve elde edilen sonuçların sağlamlığını vurgulamaktadır.

Hızla ilgili yapılan tahminlerde aykırı değerlerin direksiyon açısına göre daha az olduğu görülmektedir. Bu nedenle elde edilen R^2 değeri, direksiyon açısında elde edilene göre daha yüksektir.

Hız tahminindeki sapma miktarının frekans bazlı dağılımı Şekil 5.12’de sunulmuştur. Grafikte, x-ekseni tahmin edilen ve gerçek değerler arasındaki farkı, y-ekseni ise bu farkın frekansını ifade etmektedir. Ayrıca grafik, $\sigma^2=0,00126$ varyans değeri ile sapmaların dağılımı hakkında bilgi vermektedir.



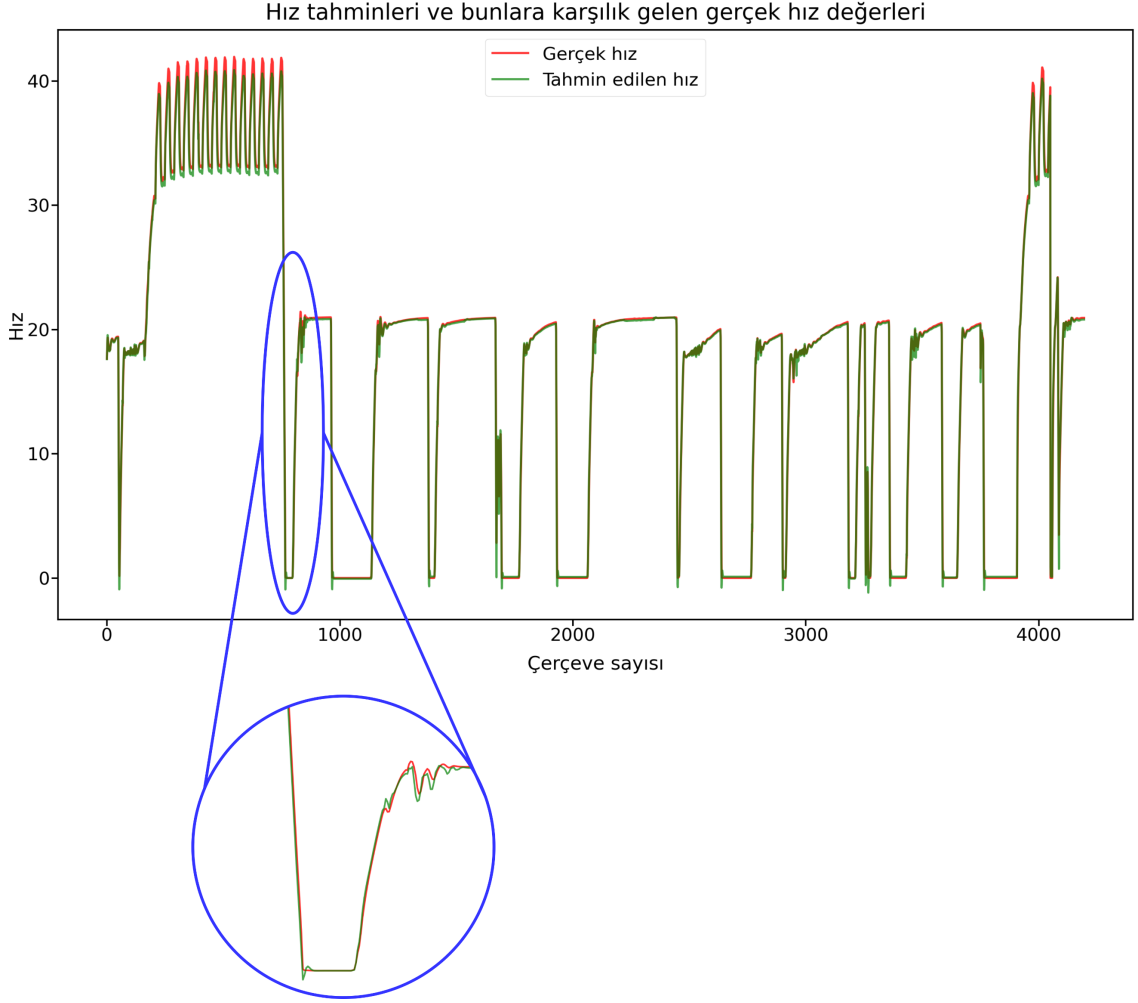
Şekil 5.11 Hız tahminlerinin gerçek verilere göre dağılımı



Şekil 5.12 Frekans bazında araç hızı sapma miktarının dağılımı

Grafiğe bakıldığında sapmaların büyük çoğunluğunun yaklaşık (-0,08-0,08) aralığında yoğunlaştığı görülmektedir. Bu durum, tahminler ile gerçek değerler arasındaki farkların genellikle küçük olduğunu göstermektedir. Ayrıca, oldukça düşük σ

değeri, sapmaların ne kadar sınırlı olduğunu göstermekte ve tahminlerin genellikle gerçek verilere yakın olduğunu vurgulamaktadır.



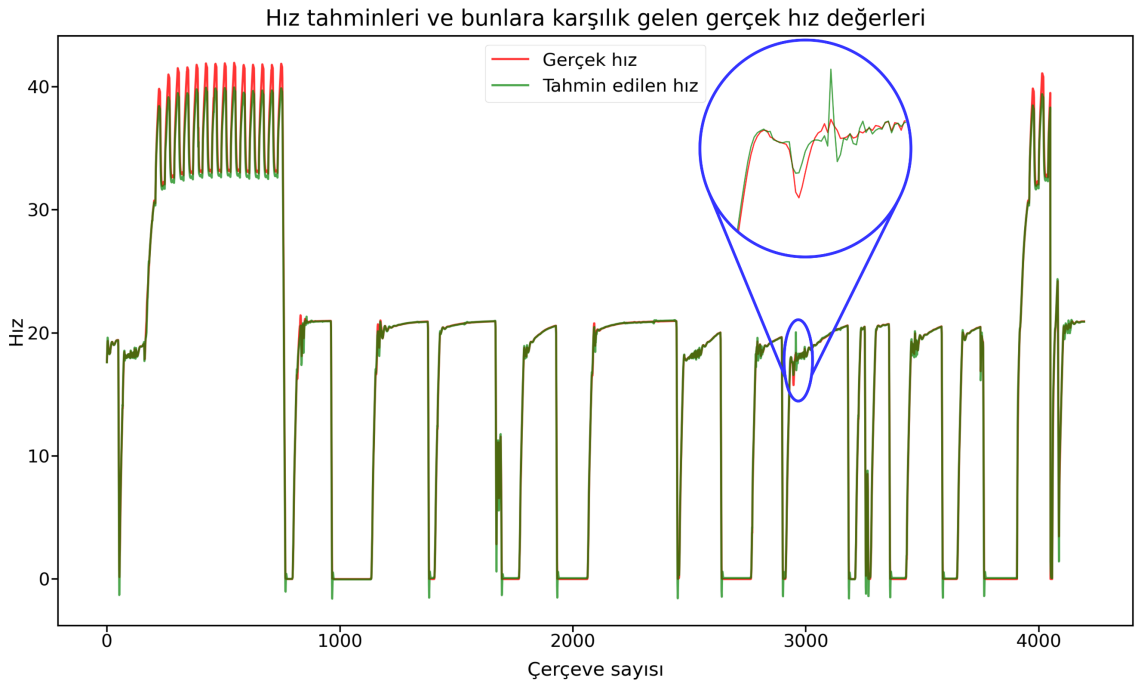
Şekil 5.13 CARLA veri kümesi üzerinde Monodepth2 monoküler derinlik tahmin metodu kullanılarak elde edilmiş hız tahminleri ve bunlara karşılık gelen gerçek hız değerleri. Tahminler yeşil renkle gösterilirken, gerçek hız değerleri kırmızı renkle gösterilmiştir

5.3.2.2 CARLA veri kümesi ile elde edilen hız tahminleri

CARLA veri seti, hız verisi açısından Udacity veri setinden belirgin bir şekilde farklılık göstermektedir. CARLA verileri, otopilot tarafından kontrol edilen bir araçla toplanmıştır ve bu nedenle hız değişimleri daha belirgin ve anidir. Araç şehir içi trafikte veri topladığı için, trafik ışıklarında durarak hareket etmiş ve bu durum hızın bazı noktalarda sıfıra düşüp sonra tekrar artmasına neden olmuştur. Araç, veri toplandığı kasaba ortamında bazı bölgelerde sabit 20 km/sa hızla seyretmişken, diğer bölgelerde 40 km/sa civarında hızlar göstermiştir. Şekil 5.13'te görüldüğü gibi, yapılan hız tahminleri

gerçek hız verileriyle oldukça uyumlu sonuçlar vermiştir. Model, hızın ani değişikliklerinde bile gerçek değerlere oldukça yakın tahminler yapabilmektedir.

Direksiyon açısı tahmininde olduğu gibi hız tahmini için de modelin kendi derinlik tahmin metodu olan Monodepth2'den elde edilen derinlik görüntüleri yerine, CARLA'nın sağlamış olduğu derinlik görüntüleri ile araç hızı tahmin edilmiştir (Şekil 5.14). Yine direksiyon açısı tahmininde olduğu gibi hız tahmininde de Monodepth2 kullanılmadan alınan sonuçlar, Monodepth2 ile alınan sonuçlara çok yakın çıkmıştır. Bu da yine kullanılan monoküler derinlik tahmin yönteminin (Monodepth2), hız tahmini konusunda da başarılı bir şekilde çalıştığını göstermektedir.



Şekil 5.14 CARLA veri kümesi üzerinde Monodepth2 metodunun sağladığı görüntüler yerine CARLA'nın kendi sağladığı derinlik görüntüleri ile elde edilmiş hız tahminleri ve bunlara karşılık gelen gerçek hız değerleri. Tahminler yeşil renkle gösterilirken, gerçek hız değerleri kırmızı renkle gösterilmiştir

5.3.3 Karşılaştırmalı değerlendirme

5.3.3.1 Udacity veri kümesi sonuçlarının karşılaştırılması

Literatürde, Udacity veri kümesi ile direksiyon açısı veya hız hesaplayan çalışmaların büyük bir çoğunluğu, kök ortalama kare hata (RMSE) veya ortalama mutlak hata (MAE) ölçütleri ile sonuçlar sunmuşlardır. Önerilen modelin performansı da mevcut ve gelecekteki çalışmalarla kıyaslama yapabilmek için RMSE ve MAE ölçütleri

kullanılarak niceliksel olarak değerlendirilmiştir. Tablo 5.2'deki sonuçlar, önerilen modelin, direksiyon açısı için 0,0164 ve hız tahmini için 0,0370 RMSE değeriyle, literatürdeki diğer yöntemlere kıyasla daha üstün bir performans sergilediğini göstermektedir. Elde edilen sonuçlar, literatürde mevcut olan en başarılı sonuçlardan, direksiyon açısına göre %44,96 ve hıza göre ise %4,39 daha iyi bir performans elde edildiğini göstermektedir. Bu bulgular, görsel girdi ve zamansal bağımlılıkların otonom sürüşteki önemli potansiyeline ışık tutmaktadır.

Tablo 5.2 Udacity veri kümesi üzerinde elde edilmiş niceliksel sonuçlar

Yöntemler	Direksiyon açısı		Hız	
	RMSE↓	MAE↓	RMSE↓	MAE↓
Deep Steering (Chi and Mu, 2017)	0,0637			
AlexNet†	0,1299			
PilotNet†	0,1604			
VGG-16†	0,0948			
MSINet (T. Wu et al., 2019)	0,0491			
CgNet (Udacity, 2017)	0,1779			
Feudal Steering (Johnson and Dana, 2020)	0,0465			
Multi-modal RNN-LSTM (Abou-Hussein et al., 2019)	0,0479			
MMMTN (Z. Yang et al., 2018)		0,0219*		<u>0,1900</u>
(D. Wang et al., 2019)		0,0205*		
Komanda (Udacity, 2017)	0,0483			
N^2C (Azam et al., 2021)	<u>0,0298</u>		<u>0,0387</u>	
FM-Net (Hou et al., 2019)	0,0411*	0,0283*		
SCMTL (Abbas et al., 2021)	0,0595			
3DCNN-LSTM+SL (Yuan et al., 2021)	0,0408			
TS2DNet (Bu çalışma)	0,0164	0,0101	0,0370	0,0238

Tabloda en başarılı sonuçlar koyu renkle vurgulanmış, ikinci en iyi sonuçlar ise altı çizili olarak verilmiştir. Hançer sembolü (†), sonuçların (Chi and Mu, 2017)'den elde edildiğini göstermektedir. Yıldız (*) ile işaretlenmiş değerler ise, bu değerlerin orijinal çalışmada verilen derece değerlerinden dönüştürüldüğünü gösterir

Tablo 5.2'deki sonuçlar, önerilen modelin literatürdeki diğer yöntemlere kıyasla daha üstün performans sergilediğini göstermektedir. Bu noktada, analiz edilen modeller daha detaylı incelenerek özetlenebilir:

Deep Steering (Chi and Mu, 2017) modeli, ardışık 15 görüntüyü alarak uzamsal-zamansal konvolüsyona sokar. Elde ettiği özellik vektörünü iki farklı kolda kullanır. Birinci kol, tam bağlı dört katman sonunda bir özellik vektörü oluşturur. İkinci kol, bir

LSTM biriminden sonra aynı boyutta özellik vektörü oluşturur. Daha sonra hepsi aynı boyutta olmak üzere, evrişim katmanlarının her birinden elde ettiği özellik vektörlerini, evrişimsel katmanların ucundaki tam bağlı katmanların sonucunda elde ettiği özellik vektörünü ve LSTM'den elde ettiği özellik vektörlerini birleştirerek yine aynı boyutta bir özellik vektörü oluşturur. Ardından elde ettiği bu birleşik özellik vektörünü ardışık bağlı LSTM ve tam bağlı katmandan geçirerek tahmin yapar. Ancak sistemin en büyük eksikliği, giriş verisi olarak $320 \times 240 \times 3 \times 15$ boyutlarında bir veri kullanılmasıdır. Giriş verisi olarak onbeş görüntü kullanan bir sistem için gerçek zamanlı sonuç üretmek, mevcut hesaplama imkanlarıyla zordur. Tablo 5.2'de yer alan Alexnet, PilotNet ve VGG-16 sonuçları Deep Steering çalışmasından alınmıştır.

MSINet (T. Wu et al., 2019) modeli, direksiyon açısı tahmini için farklı ölçeklerden gelen uzamsal bilgileri kodlamayı amaçlayan Çok Ölçekli Uzamsal-Zamansal Entegrasyon (MSI) modülüne sahip Evrişimli Uzun Kısa Süreli Bellek (Conv-LSTM) ağına dayalı uçtan uca bir sürüş modeli önermektedir. Dört adet Conv-LSTM modülü, direksiyon açısını oluşturmak için kodlayıcının farklı seviyelerine bağlanır. Ayrıca, bu çalışma modelinin uzamsal-zamansal özelliklerini artırmak için eğitim sırasında gelecekteki sıralı bilgileri (gelecek dört görüntü) kullanmıştır. MSI-Net modeli de bir önceki çalışma gibi geçmiş n adet görüntüyü giriş verisi olarak kullanır. Bu tez çalışmasında önerilen model ise yalnızca tahmin anındaki RGB görüntüyü kullanmaktadır. Tabloda sonuçları sunulan CgNet ve Komanda modelleri, Udacity yarışmasında başarılı olmuş ağlardır.

Feodal Steering modeli (Johnson and Dana, 2020), iki bölümden oluşmaktadır: bir yönetici ağı ve bir işçi ağı. Yönetici ağ, üst düzey eylemlerin planlanmasından sorumluyken, işçi ağı, arabanın yönlendirilmesi gibi alt düzey eylemleri gerçekleştirir. İşçi ağın mimari yapısı ardışık evrişimsel ağlar, LSTM birimi ve tam bağlı katmanlardan oluşur. Geçmişe dönük m adet direksiyon açısı tahmini yönetici ağına verilir. İşçi ağın girişi ise ardışık m görüntü, yönetici ağdan çıkış olarak alınan hedef direksiyon açısı değeri ve geçmişe dönük m adet direksiyon açısından oluşur. Benzer şekilde buradaki ağ yapısında da giriş verisi olarak çok sayıda görüntü kullanılması durumu vardır.

Multimodal RNN-LSTM modeli ise (Abou-Hussein et al., 2019), beş katmanlı bir ESA ağından ve ardından gelen bir LSTM biriminden oluşur. Burada giriş verisi olarak t anındaki görüntü ve bu görüntüden elde edilmiş optik akış görüntüsünün yatay ve dikey bileşeni, birleştirilmiş üç görüntü olarak kullanılmıştır.

MMMTN modeli (Z. Yang et al., 2018) bir görsel kodlayıcı ve bir hız kodlayıcı içerir. Görsel kodlayıcı bir görüntüyü girdi olarak alırken, hız kodlayıcı önceki hız dizilerini alır. Direksiyon açısı tahmini, bir evrişimsel sinir ağı ile yapılırken, hız tahmin bloğu farklı bir yapıya sahiptir. Burada ardışık hız verileri girdi olarak alınıp bir LSTM biriminden geçirilir. Daha sonra, iki tam bağlı katmandan geçen özellikler, görüntü tahmin ağının ara katmanından gelen özellik vektörü ile birleştirilip hız tahmini yapılır. Bu durumda, hız ve direksiyon açısı tahmin ağları birbirinden ayrı olarak çalışmaktadır. Örneğin direksiyon açısı tahmini, hız bilgisinden hiç faydalanmadan yalnızca giriş görüntüsüne dayalı olarak üretilmektedir. Her iki bloğun tam bağlı katmanları ayrıdır. Ancak bu tez çalışmasında önerilen modelde, LSTM kolundan ve görüntü kollarından gelen yüksek seviye özellikler birleştirildikten sonra tam bağlı katmanlardan geçilmekte ve sonrasında iki tahmin koluna ayrılıp direksiyon açısı ve hız tahmini yapılmaktadır. Bu durumda hem direksiyon açısı hem de hız tahmini yapılırken bütün girişlerden faydalanılmaktadır.

(D. Wang et al., 2019) tarafından tasarlanan model, derin evrişimsel sinir ağını LSTM (CNN-LSTM) ile birleştirmektedir. Ayrıca, sürüş senaryolarının anlaşılmasını artırmak için semantik segmentasyon ve nesne tespiti ile ilgili iki yardımcı görev de eklenmiştir. Bu yardımcı görevler sadece eğitim aşamasında kullanılmış, tahmin aşamasında modele dahil edilmemiştir.

N²C (Neural Network-based Controller) (Azam et al., 2021) çalışmasında, tahmin için iki ayrı model sunulmuştur. Birinci model, giriş görüntüsünden bir ESA aracılığıyla direksiyon açısını ve hızı tahmin eder. Daha sonra bu modelin çıktıları, düşük seviyeli kontrol parametreleri oluşturmak için kontrolör modeline girdi olarak verilir. LSTM tabanlı kontrolör, bu direksiyon açısı ve hız girdilerine dayalı olarak gaz, fren ve torku tahmin eder. Burada direksiyon açısı ve hız yalnızca görüntü değerlendirilerek tahmin edilmiş, zamansal özelliklerden faydalanılmamıştır. Kontrolör birimi de doğrudan görüntüden faydalanmamış, yalnızca direksiyon açısı ve hıza göre gaz, fren ve tork tahmini yapmıştır.

FM-Net (Hou et al., 2019) modeli, ana yapı olarak ardışık evrişimsel katmanları takip eden tam bağlı katmanlar ve ardından gelen bir LSTM bloğu içermektedir. Ancak FM-Net modeli, ana omurga yanında (D. Wang et al., 2019)'da olduğu gibi yardımcı ağlar içerir. Burada yardımcı ağlardan biri semantik segmentasyonu diğeri ise optik akışı hesaplamaktadır. Yazarlar, yardımcı ağların ara evrişimsel katmanları ile ana ağın evrişimsel katmanları arasında bir ilişki kurmuşlardır. Burada yardımcı ağların

katmanlarındaki özelliklerin ana omurga tarafından taklit edilmesi için eğitim sırasında kayıp hesaplanmıştır. (D. Wang et al., 2019)'da olduğu gibi, burada da test aşamasında yardımcı ağlar kullanılmamıştır. FM-Net'te giriş verisi olarak ardışık 10 görüntü kullanılmış, çıkışta ise direksiyon açısı, hız ve tork tahmin edilmiştir.

SCMTL (Abbas et al., 2021) modeli, istatistiksel olarak ilişkili görevler için bir öğrenme mimarisi önermektedir. Ancak, çalışmada anlatılan ve grafikleri sunulan veriler arasında (negatif korelasyonlu hızlanma-fren çifti dışında) istatistiksel bir ilişki gözlemlenmemiştir. Örneğin, direksiyon açısı ve hız arasında bir ilişki bulunmamaktadır. Bu modelde tek bir giriş görüntüsü kullanılarak belirli bir katmana kadar evrimsel konvolüsyon uygulanmıştır. Daha sonra direksiyon açısı, fren basıncı ve hızlanma için üç farklı görev kolu evrimsel konvolüsyondan ayrılmış ve her bir görev için özel katmanlarla tahmin yapılmıştır. Mimarisi, göreve özgü üç kola ayrılmış ESA'ya dayanmaktadır.

3DCNN-LSTM+SL çalışması, dengesiz eğitim kümeleri için tasarlanan SteeringLoss'un (Yuan et al., 2021) etkinliğini üç farklı model üzerinde göstermiştir. Direksiyon açısı verisi, genellikle düz gidiş açısı etrafında yoğunlaşır. Önerilen kayıp fonksiyonu da bu tip bir veriye özgü olarak tasarlanmıştır. Daha sonra bu fonksiyon, üç farklı model ile değerlendirilmiş ve üç tip deney yapılmıştır: Birinci deney, önerilen kayıp fonksiyonunun dengesiz eğitim kümesinde kullanımını; ikinci test, kayıp fonksiyonunun kullanımı ile ilgili tarifleri ve ince ayarları; üçüncü test ise kayıp fonksiyonunun farklı modellerle kullanımında elde edilen sonuçları göstermektedir.

5.3.3.2 CARLA veri kümesi sonuçlarının karşılaştırılması

CARLA sürüş simülatörü üzerine gerçekleştirilen yayınların büyük çoğunluğu, CARLA'nın düzenlediği yarışmalarda (Benchmark) başarılı olmuş çalışmaların çıktılarıdır. Bu yarışmalar arasında ilk çıkan orijinal CARLA Benchmark yarışmasıdır, daha sonrasında ise NoCrash Benchmark olarak bilinen kazasız sürüş yarışması gelmiştir. Bu yarışmalardaki değerlendirme kriterleri, sürüş performansı üzerine yoğunlaşmıştır. Örneğin, aracın sola dönüşleri, düz seyirleri, boş veya yoğun trafikteki başarı oranları gibi kriterler, performansı değerlendirmede kullanılmıştır. Özellikle belirli bir başarı yüzdesine ulaşma, çeşitli trafik senaryolarında veya yol koşullarında başarılı sürüşler sergileme önemli değerlendirme kriterleri arasında yer almıştır. Ancak, bu değerlendirme kriterleri daha çok sürüşün başarılı tamamlanması üzerine yoğunlaştığından, çalışmalar

genellikle sürüş performansı ve senaryo başarıları üzerinde durmuş, yayınlarda direksiyon açısı tahmini pek az paylaşılmıştır.

Genellikle CARLA veri kümesi ile yapılan yayınlarda, veri toplama süreçleri belirli kasabalarda sınırlı kalarak, örneğin *Town01* ve *Town02* gibi yerlerde yoğunlaşmıştır (Agarwal et al., 2021; Behl et al., 2020; Codevilla et al., 2018; Huang et al., 2021; Sallab et al., 2018; Q. Wang et al., 2019; Xiao et al., 2022; L. Yang et al., 2018). Bu süreçte toplanan verilerin bir kısmı test verisi olarak ayrıldıktan sonra direksiyon açısı veya hız tahmini gibi analizlere tabi tutulmuştur. Dolayısıyla test verisindeki görüntüler, eğitim verisindeki görüntülerle benzerlikler gösterebilmektedir.

Ancak bu tez çalışmasında, önceki çalışmalardan farklı bir yaklaşım benimsenmiştir. Kullanılan test verisi, eğitim verisinden tamamen farklı bir kasabada toplandığı için, test için seçilen görüntüler yol yapısı, sürüş çevresi gibi faktörler açısından eğitim verisinden oldukça farklılık göstermektedir. Bu durum, tez çalışmasındaki test sürecini daha zorlu hale getirmektedir ve modelin genelleme yeteneğini daha zorlayıcı bir şekilde değerlendirmeye olanak tanımaktadır.

Tablo 5.3 CARLA veri kümesi üzerinde elde edilmiş niceliksel sonuçlar

Yöntemler	Veri sayısı	Kasaba	Direksiyon açısı		Hız	
			RMSE↓	MAE↓	RMSE↓	MAE↓
Random†			0,4640			
Constant†			0,3480			
Inception†			0,2970			
MobileNet†			0,2860			
VGG-16†			0,3160			
DroNet†			0,3330			
FADNet (Nguyen et al., 2021)	73.235	belirsiz	0,2030			
PilotNet‡				1,2080		
cGAN‡				1,2060		
DU-Drive (L. Yang et al., 2018)	732.200	01,02		1,0680		
CNN-GRU (Martinez et al., 2021)	60.000	01			3,2700	
3D-CNN (Martinez et al., 2021)	60.000	01			1,2600	
TS2DNet (Bu çalışma)*	42.000	01,02,03,04,05	<u>0,0250</u>	0,0063	<u>0,6064</u>	0,2756
TS2DNet (Bu çalışma)	42.000	01,02,03,04,05	0,0242	<u>0,0072</u>	0,5388	<u>0,2821</u>

Tabloda en başarılı sonuçlar koyu renkle vurgulanmış, ikinci en iyi sonuçlar ise altı çizili olarak verilmiştir. Hançer sembolü (†), sonuçların (Nguyen et al., 2021)'den alındığını; çift hançer (‡) ise sonuçların (L. Yang et al., 2018)'den alındığını göstermektedir. Yıldız (*) ile işaretlenmiş model, CARLA'nın sağladığı derinlik görüntüleri kullanılarak elde edilen hata değerlerini gösterir

CARLA veri kümesi ile direksiyon açısı veya hız hesaplayan çalışmalar, her iki büyüklük için de kök ortalama kare hata (RMSE) veya ortalama mutlak hata (MAE) ölçütleri ile sonuçlar sunmuşlardır. Önerilen modelin performansı da mevcut ve gelecekteki çalışmalarla kıyaslama yapabilmek için RMSE ve MAE ölçütleri kullanılarak niceliksel olarak değerlendirilmiştir. Tablo 5.3'teki sonuçlar, önerilen modelin, direksiyon açısı için 0,0242 ve hız tahmini için 0,5388 RMSE değeriyle, CARLA veri kümesi üzerinde literatürde sunulan diğer yöntemlere kıyasla daha üstün bir performans sergilediğini göstermektedir. Bu sonuçlar, mevcut literatürde elde edilen en başarılı sonuçlardan %88,18 daha iyi bir direksiyon açısı tahmini ve %77,6 daha iyi bir hız tahmini yapıldığını göstermektedir. Bu bulgular, Udacity veri kümesi ile elde edilen başarının, veri kümesine özgü olmadığını ve modelin genelleme yeteneğinin yüksek olduğunu göstermektedir. Ayrıca CARLA veri kümesi ile toplanan eğitim ve test verilerinin büyük farklılıklar gösterdiği de belirtilmelidir. Test verisinin eğitim verisiyle benzeşmediği durumlarda dahi modelin en başarılı sonuçları üretmesi, modelin veriye esnek ve etkili bir şekilde adapte olabildiğini ortaya koymaktadır.

5.3.3.3 Sully Chen veri kümesi sonuçlarının karşılaştırılması

Sully Chen veri kümesi, bu çalışmada test edilen ve sonuçları karşılaştırılan üçüncü veri kümesidir. Udacity ve CARLA'da elde edilen başarıların ardından Sully Chen veri kümesi de benzer şekilde test edilmiş ve değerlendirilmiştir. Elde edilen sonuçlar, Sully Chen veri kümesi üzerinde de modelin yüksek performansını doğrulamıştır. Modelin, bu farklı veri kümesiyle de tutarlı ve başarılı bir şekilde çalışabilmesi, genelleme yeteneğinin güçlü olduğunu göstermektedir. Bu kapsamlı karşılaştırmalar, önerilen modelin farklı veri kaynakları üzerindeki güvenilirliğini ve etkinliğini vurgulamaktadır.

Sully Chen veri kümesi, ardışık görüntülerle birlikte yalnızca direksiyon açısı bilgisi sunmakta olup, hız verisi içermez. Bu nedenle Sully Chen veri kümesi için yapılan karşılaştırmalarda model, çok girdili ama tek görevli çalışacak şekilde yalnızca direksiyon açısı bilgisi üretmiştir.

Sully Chen veri kümesi ile elde edilen sonuçlar da hem RMSE hem de MAE ölçütleri kullanılarak niceliksel olarak değerlendirilmiştir. Tablo 5.4'teki sonuçlar, önerilen modelin, direksiyon açısı için 0,0058 RMSE değeriyle, Sully Chen veri kümesi üzerinde literatürde sunulan diğer yöntemlere kıyasla daha üstün bir performans

sergilediğini göstermektedir. Bu sonuçlar, mevcut literatürde elde edilen en başarılı sonuçlardan %97,13 daha iyi bir direksiyon açısı tahmini yapıldığını göstermektedir. Bu bulgular, Udacity ve CARLA veri kümeleri ile elde edilen başarının, veri kümesine özgü olmadığını ve modelin genelleme yeteneğinin yüksek olduğunu göstermektedir. Test verisinin eğitim verisiyle benzeşmediği durumlarda dahi modelin en başarılı sonuçları üretmesi, modelin veriye esnek ve etkili bir şekilde adapte olabildiğini ortaya koymaktadır.

Tablo 5.4 Sully Chen veri kümesi üzerinde elde edilmiş niceliksel sonuçlar

Yöntemler	Direksiyon açısı	
	RMSE↓	MAE↓
Autopilot†		0,0930
PilotNet†		0,2350
SCMTL (Abbas et al., 2021)		<u>0,0560</u>
PilotNet‡	0,2280	
MDNN (Qian et al., 2018)	<u>0,2024</u>	
Autopilot§	0,2116	
Comma AI§	0,2347	
TS2DNet (Bu çalışma)	0,0058	0,0035

Tabloda en başarılı sonuçlar koyu renkle vurgulanmış, ikinci en iyi sonuçlar ise altı çizili olarak verilmiştir. Hançer sembolü (†), sonuçların (Abbas et al., 2021)'den alındığını; çift hançer (§) sonuçların (Qian et al., 2018)'den alındığını; bölüm sembolü (§) ise sonuçların (Machiraju and Balasubramanian, 2020)'den alındığını göstermektedir

5.3.4 Çalışma süresi analizi

Otonom araç sistemlerinde sensör verilerinin alınmasından nihai kontrol kararlarının verilmesine kadar geçen süre, kritik bir öneme sahiptir. Bu sistemlerin gerçek zamanlı çalışması, nihai kontrol çıktılarının hızlı ve etkin bir biçimde üretilmesini gerektirir. Bu çalışmada, alt ağların ve birimlerin operasyonel süreleri titizlikle analiz edilmiş ve sürekli olarak iyileştirilmeye çalışılmıştır. Gecikmelere neden olan birimler iyileştirilme ya da daha hızlı alternatif yöntemlerle değiştirilme adına incelenmiştir. Bu çalışma, önceki araştırmalardan farklı olarak alt birimlerin ve bütün modelin operasyonel sürelerini kapsamlı bir şekilde ele almaktadır. Literatürde yapılan incelemeler neticesinde, model ve model içindeki alt birimlerin işlem sürelerinin ayrıntılı şekilde sunulduğu bir çalışmaya rastlanılmamıştır. Bu nedenle sunulan sonuçlar, gelecekteki çalışmalar için öngörü sağlaması açısından önemli bir kaynak niteliği taşımaktadır.

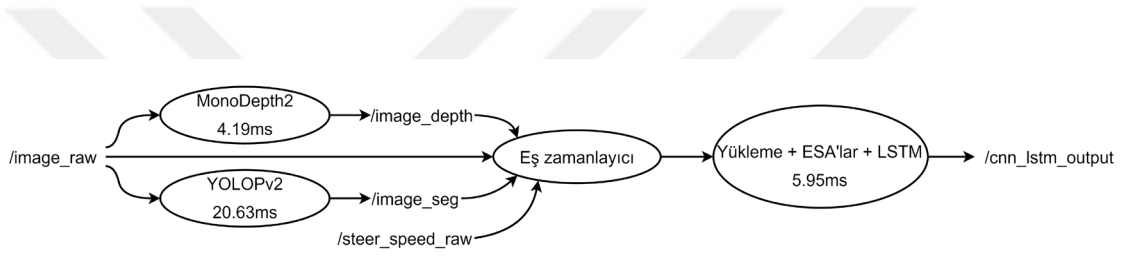
Bu tez çalışmasında modelin çıkarım süresini hızlandırmak ve alt birimlerin daha kısa sürede işlem yapmasını sağlamak için “ONNX Runtime” kullanılmıştır. ONNX Runtime, ONNX standardında tanımlanan modelleri yüksek performanslı bir şekilde çalıştırmak için kullanılır. ONNX (Open Neural Network Exchange- Açık Sinir Ağı Değişimi), derin öğrenme modellerini taşımak ve farklı derin öğrenme çerçeveleri arasında paylaşmak amacıyla oluşturulan açık bir standarttır. Modelin mimarisi, ağırlıkları ve diğer özellikleri ONNX formatında ifade edilir. Modelin eğitildiği çerçeveden bağımsız olarak modellerin evrensel bir formatta temsil edilmesini sağlar. ONNX modeli, yapının mimarisi, ağırlıklar ve parametrelerden oluşan bir sinir ağını temsil eder. Modeller, standart bir formatta depolanır ve farklı çerçeveler arasında kolaylıkla dönüştürülebilir. Operatörler, ONNX modellerinin yapı taşlarını oluşturur. Yapay sinir ağı içinde gerçekleştirilen matematiksel işlemleri (evrişim, havuzlama, aktivasyon fonksiyonları gibi) temsil eder. ONNX, çeşitli çerçeveler tarafından desteklenen ortak operatörlerin bir kümesini tanımlar. Keras, Pytorch, Tensorflow, MATLAB çerçeveleri ONNX’i destekler. ONNX bu kütüphaneler ve platformlar arasında model değişimi sağlar. ONNX’in kullanıcılara sunduğu, bazı konularda büyük kolaylıklar sağlayan araçları ve kütüphaneleri mevcuttur. Örneğin “ONNX Model Zoo” içinde çeşitli veri kümelerinde eğitilmiş hem dil hem de bilgisayarlı görü konularında hazır modeller mevcuttur. Ayrıca modelin geliştirme aşamasında kayıp ve özellik haritası gibi özelliklerini izlemek için kullanılan görüntüleme araçları sunulmuştur.

ONNX Runtime, ONNX modelleri üzerinde kullanılan, platformlar arası bir çıkarım ve makine öğrenimi hızlandırıcısıdır. ONNX modellerini, modelin eğitildiği çerçeveden bağımsız olarak çalıştırabilme esnekliği sunar. Açık kaynaklı bir çerçeve olup, özellikle yapay zekâ modelinin dağıtımını ve çeşitli platformlarda çalıştırılmasını kolaylaştırmak için tasarlanmıştır. Bu, farklı derin öğrenme çerçeveleri arasında sorunsuz bir geçiş yapılabilmesine olanak tanır. ONNX Runtime’ın temel özellikleri, modelin evrensel bir formatta temsil edilmesi ve çeşitli çerçevelerde çalıştırılabilmesi gibi avantajları içerir.

Bu tez çalışması kapsamında Keras ile geliştirilen model, önce ortak platform olan ONNX formatına, daha sonra da ONNX Runtime formatına dönüştürülerek tahminler elde edilmiş ve çalışma süresi analizleri yapılmıştır. Modelin üç giriş görüntüsünden ikisini oluşturan derinlik ve semantik bölümlenmiş görüntülerin elde edildiği Monodepth2 ve YOLOv2 modelleri de ONNX Run time ile çalıştırılmıştır. Monodepth2 tarafından üretilen derinlik görüntüsü 4,19ms’de elde edilirken, YOLOv2 tarafından

üretilebilen bölümlenmiş görüntünün elde edilmesi 20.63ms'de tamamlanmaktadır (Şekil 5.15). Görüldüğü gibi her ikisi de RGB görüntüden elde edilen derinlik görüntüsü ve bölümlenmiş görüntüler aynı hızda üretilmemektedir. Alt ağ çıktılarının eşit sürede üretilmemesi, verilerin zaman açısından senkronize olamamasına ve birleştirmeye başlandığında zaman senkronizasyonu açısından tutarsızlık oluşmasına neden olur. Farklı zaman damgalarına sahip veriler birleştirilmeye başlanacağından çok girdili sistem sağlıklı çalışmaya başlar ve işlem süreci bozulur.

Önerilen model, farklı zaman aralıklarında çalışan paralel yapılar içermektedir. Çalışmada alt ağları entegre etmek, veri uyumsuzluğunu önlemek ve zamanlama problemlerini gidererek doğru tahminler sunmak adına ROS (Robot İşletim Sistemi) (ROS, 2023) kullanılmıştır. Şekil 5.15'da ROS çerçevesi görülmektedir.



Şekil 5.15 Önerilen modelin eşleşen zaman damgalarını doğrulayarak alt ağ verilerini senkronize eden ROS çerçevesi

ROS, robot yazılımlarının geliştirilebilmesi için tasarlanmış, dağıtık ve modüler bir yazılım çerçevesidir. Robotların algılama, kontrol, hareket ve iletişim gibi işlevlerini gerçekleştirmek için kullanılan açık kaynaklı bir projedir. ROS'un temel amacı, robot yazılımının geliştirilmesini kolaylaştırmak ve yazılım bileşenlerinin tekrar kullanılabilirliğini artırmaktır. Bunu yaparken, yazılım modüllerini bağımsız bir şekilde geliştirip test etme, ardından bu modülleri birleştirme ve farklı robot platformlarına taşıma imkânı sağlar. Robotik uygulamalar için kullanılan çeşitli kütüphaneler, araçlar ve hazır bileşenler içerir. Örneğin, görüntü işleme için OpenCV, 3D haritalama için GMapping gibi popüler ROS paketleri mevcuttur.

ROS'un birkaç önemli bileşeni vardır:

1. *Node (düğüm)*: ROS'un temel yürütme birimidir. Her bir *düğüm* bağımsız görevleri yerine getirmek için tasarlanmış bağımsız program parçasıdır.
2. *Topic (konu)*: ROS *düğüm*'leri arasında veri iletişimini sağlayan bir iletişim mekanizmasıdır. Düğümler bir konu ile veriyi yayınlabilir (publish) veya yayınlanmış bir konu ile eşleşiyorlarsa abone olabilirler (subscribe).

3. *Message (message)*: Mesajlar ROS konuları üzerinden iletilirler. Haberleşme için tanımlanmış ve tipi kesin olarak belirlenmiş veri yapısıdır.
4. *Service (servis)*: Düğümlerin kendi aralarında bire bir haberleşmesini sağlar. Servis aracılığıyla bir düğüm, başka bir düğümden talepte bulunabilir. Düğümler bir servis sağlayabilir ya da bir servis kullanabilir.
5. *Package (paket)*: Paketler, ROS içindeki yazılımı organize ederler. Bir paket içinde bir veya birden fazla düğüm, mesaj, yapılandırma dosyası ve servis olabilir.

Bu bileşenler kullanılarak çok fazla alt birim veya modül içeren karışık sistemler kolaylıkla yürütülebilir. Birimler arasında sağlıklı bir haberleşme sistemi tesis edilerek senkronizasyon sağlanır.

Bu çalışmada ROS, aynı zaman damgasına sahip veriler arasında senkronizasyonu sağlayarak ağın üretim hızını en yavaş alt ağa uyarlamaktadır. Özellikle 4,19ms (Monodepth2) ve 20,63ms (YOLOPv2) işlem sürelerine sahip paralel alt ağların bulunduğu ön işleme aşamasının hızını belirlemede, en uzun işlem süresine sahip YOLOPv2 modeli belirleyici bir rol oynamaktadır. Bu durum, 5,95 ms'lik yükleme ve ESA birimlerinin dahil edilmesiyle birlikte tek bir görüntüden direksiyon açısı ve hız tahminlerinin 26,58 ms'de üretilebileceğini göstermektedir. Dikkat çekici bir şekilde, işlem süresinin yaklaşık %75'ini bölümlenmiş görüntü çıkarımının oluşturduğu görülmektedir. Bu elde edilen sonuçlar, önerilen modelin yaklaşık 38 fps hızında çalışabildiğini ortaya koymaktadır. Bu da şehir içinde 50 km/sa hızla seyreden bir araç için, yaklaşık 0,37m'de bir direksiyon açısı ve hız tahmini yapabilme anlamına gelir.

Otonom araçlarda, genellikle 30 fps kare hızı, gerçek zamanlı çalışacak sistemler için kabul edilebilir bir minimum gereksinim olarak görülmektedir. Bu bağlamda, önerilen modelin, gerçek zamanlı uygulamalarda başarıyla kullanılacak bir performans seviyesine sahip olduğu ifade edilebilir.

6. RAPIDSENSE VERİ KÜMESİ

Geliştirilen model, genellenebilirliğini ve sağlamlığını göstermek amacıyla Udacity veri kümesi dışında iki farklı veri kümesi ile daha test edilmiştir. Bunlardan birisi, CARLA Otonom Sürüş Simülatörü kullanılarak bu tez çalışması için oluşturulmuş veri kümesidir. CARLA veri kümesinin detayları, oluşturulma süreci ve uygulaması önceki bölümlerde ayrıntılı olarak açıklanmıştır. CARLA simülatörünün gelişmiş grafik motoru ve zengin çevre birikimi sayesinde, gerçekçi senaryolar oluşturmak ve veri toplamak mümkün olmuştur. Bu da veri kümesinin güvenilirliğini ve gerçek dünya koşullarını yansıtmaya yeteneğini artırmıştır.

CARLA'nın yetenekleri ve veri toplama sürecinde sağladığı esneklikler ve geniş sensör kapasitesi, veri kümesinin oluşturulması sırasında fark edilmiştir. Ancak, CARLA sürüş simülatörüyle çalışmaya başlamak ve ihtiyaçlara uygun veri toplama becerisi kazanmak, zaman alan bir öğrenme süreci gerektirmektedir. Ayrıca, verinin toplanması güçlü bilgisayar kaynakları gerektirir. Bununla birlikte, toplanan verinin kullanılabilir hale getirilmesi de zaman gerektiren bir süreçtir. Bu yüzden, farklı amaçlar doğrultusunda CARLA üzerinde çalışmak isteyen araştırmacıların kolayca erişebileceği, çeşitli amaçlara hitap edebilecek bir veri kümesinin oluşturulmasına karar verilmiştir. Bu veri kümesi, kullanıma hazır bir şekilde sunulmuştur, böylece araştırmacılar verilere ulaştıktan sonra hemen kullanabilirler. Söz konusu yaklaşım, araştırmacıların veri kümesi oluşturmak için zaman ve efor harcamadan hızlı bir şekilde veri kümesi ile çalışmalarını sürdürmelerine olanak tanır. Bu bölümde, oluşturulan veri kümesi detaylı bir şekilde anlatılacaktır.

Veri kümesinin kapsayıcılığını artırmak için bilgisayar kapasitesi ile paralel olarak, veri toplama hızı 30 FPS üzerinde kalacak şekilde maksimum sayıda sensörden veri toplanmıştır. Veriler, CARLA'nın oto pilot özelliği kullanılarak senkron olarak toplanmıştır. Bu esnada araştırmacıların kullanmak isteyebileceği çevresel koşullar da göz önünde bulundurulmuştur. Böylece, otonom sürüş sistemlerinin farklı hava koşulları, yol şartları ve trafiğe nasıl tepki verdiği gibi gerçek dünya senaryolarını daha iyi yansıtabilmek amaçlanmıştır.

Bu yaklaşımla, araştırmacılar ve geliştiriciler, CARLA üzerinde çalışmak istedikleri farklı amaçlar için uygun, geniş kapsamlı ve erişilebilir bir veri kaynağına sahip olacaklardır. Bu veri kümesi sayesinde, otonom sürüşle ilgili görüntü segmentasyonu, derinlik çıkarımı, sahne anlama, çarpışma sakınımı, sürücü davranışı, odometri, konumlama ve haritalama gibi daha birçok farklı konuda çalışmalar yapılabilir.

Bu da otonom sürüş teknolojileri üzerindeki araştırma ve geliştirme çalışmalarının ilerlemesine katkı sağlayacaktır.

Oluşturulan veri kümesinin detayları aşağıda anlatılmıştır.



Şekil 6.1 Aracın dört yönünü gösteren RGB kameralar. Üst sıra, soldan sağa, ön ve sağ kamera; alt sıra soldan sağa, arka ve sol kamera

6.1 Sensör Düzeni

Veri kümesinin, çok daha fazla araştırmacıya hitap edebilmesi için bilgisayar kaynaklarının imkân verdiği ölçüde çok sayıda sensörden veri toplanmıştır. Şekil 6.1 ve Şekil 6.2’de farklı sürüş sahnelerinde sensörlerden elde edilen görüntüler görülmektedir. Veri toplanan sensörlerin özellikleri ve belirlenen parametreleri aşağıda detaylı olarak açıklanmıştır:

- **4×RGB kamera:** Aracın ön, arka, sağ, sol taraflarını gösterecek şekilde dört adet RGB kamera kullanılmıştır. Bu kameraların görüş açısı 90° ’dir. Dolayısıyla dört kamera görüntüsü birlikte aracın 360° etrafını gösterebilmektedir. Bu yapının, kuş bakışı görüntü üzerine çalışan araştırmacılar için de kullanışlı olması düşünülmüştür. Toplanan görüntülerin boyutu 640×480 olarak belirlenmiştir.

Tablo 6.1 Kamera veri yapısı

Veri Yapısı		
çerçeve ismi	zaman damgası	ham veri

- **1×Radar:** Radar parametrelerinde CARLA'nın varsayılan değerleri kullanılmıştır. Ponits/sec: 1500, mesafe: 100m, düşey görüş açısı: 30°, yatay görüş açısı: 30°

Tablo 6.2 Radar veri yapısı

Veri Yapısı		
çerçeve ismi	zaman damgası	ham veri (altitude, azimuth, derinlik, hız)

- **1×LiDAR:** Kanal sayısı: 64, mesafe: 100m, Ponits/sec:250,000, dönme frekansı: 20Hz, görüş alanı üst sınırı: 10°, görüş alanı alt sınırı: -30°. Bunların dışında CARLA'nın varsayılan değerleri kullanılmıştır.

Tablo 6.3 LiDAR veri yapısı

Veri Yapısı		
çerçeve ismi	zaman damgası	ham veri (32-bitlik float dizisi (her bir nokta için XYZI))

- **1×DVS Kamera:** Görüntü boyutu 640×480 olarak belirlenmiş, bunun dışındaki dğerlerde CARLA'nın varsayılan değerleri kullanılmıştır.

Tablo 6.4 DVS Kamera veri yapısı

Veri Yapısı		
çerçeve ismi	zaman damgası	ham veri (x,y,t,pol)

- **GNSS:** CARLA'nın varsayılan değerleri kullanılmıştır.

Tablo 6.5 GNSS veri yapısı

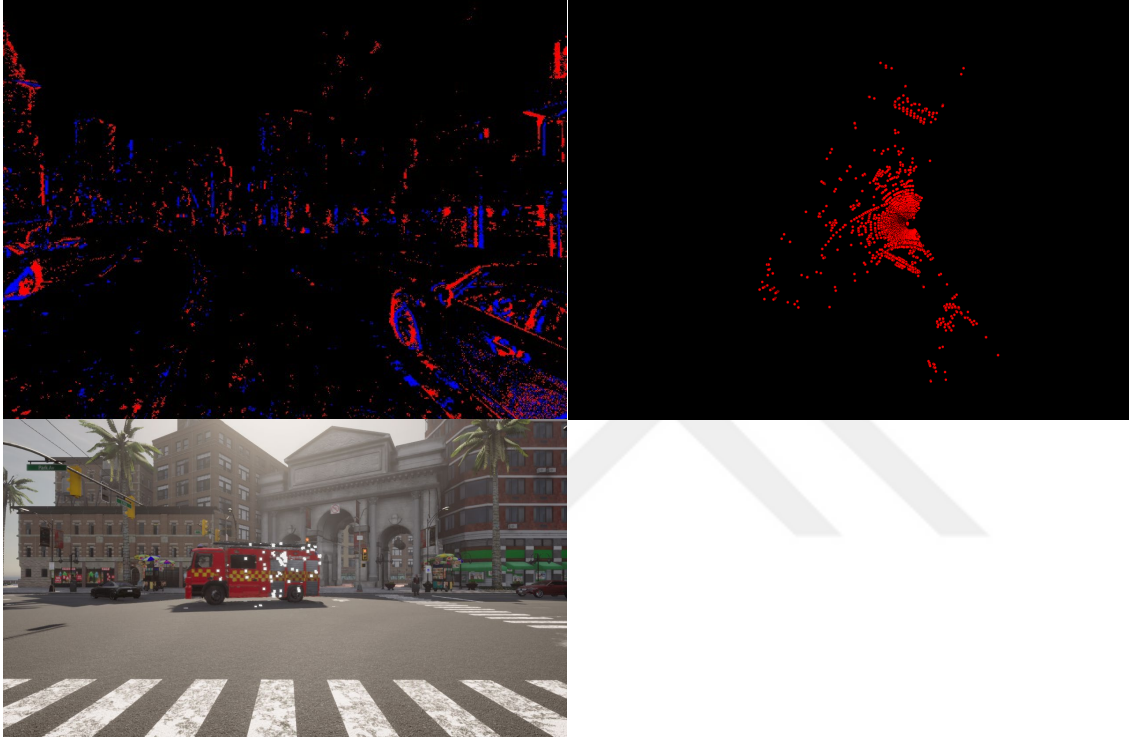
Veri Yapısı		
çerçeve ismi	zaman damgası	ham veri (latitude, longitude, altitude)

- **IMU:** CARLA'nın varsayılan değerleri kullanılmıştır.

Tablo 6.6 IMU veri yapısı

Veri Yapısı		
çerçeve ismi	zaman damgası	ham veri (ivmeölçer (m/s ²), cirooskop (rad/sn), kumpas(rad))

Sensörler araç üzerinde $x=0$, $y=0$, $z=2,4$ konumuna yerleştirilmişlerdir.



Şekil 6.2 Sol üst köşede DVS kamera, sağ üst köşede LiDAR görüntüsü, altta ise kamera görüntüsü üzerine yansıtılmış radar verisi görülmektedir

6.2 Veri Tanımlaması

Veri kümesinin ham verisine ve veri kümesi ile ilgili dokümantasyona <https://github.com/sazak1/RapidSense> adresinden ulaşılabilir. Oluşturulan veri kümesi, hem farklı kullanım amaçlarına hitap etmek hem de veri kümesine çeşitlilik sağlamak amacıyla CARLA simülatörünün sunmuş olduğu farklı kasabalardan toplanmıştır. Bazı kasabaların boyutlarının çok küçük olması veya çevresel görüntülerinin ve yol yapılarının diğer kasabalarla benzerlik göstermesi nedeniyle belirli kasabalardan veri toplanmamıştır.

Veri kümesinin oluşturulması için kullanılan kasabalar şunlardır: *Town02*, *Town03*, *Town04*, *Town05*, *Town06*, *Town07*, *Town10*, *Town12*. Her bir kasabanın yapısal özellikleri ve harita boyutları göz önünde bulundurularak, farklı hava koşullarında ve farklı sayılarda veri toplanmıştır. Kasabaların yapısal özellikleri ve detayları şu şekildedir;



Şekil 6.3 *Town02* kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)

Town02, çok sayıda T-tipi kavşak ve çeşitli binalar içeren ve diğer kasabalara kıyasla nispeten küçük olan bir kasabadır (Şekil 6.3). Çevrede iğne yapraklı ağaçlar, bir yerleşim ve ticaret alanı, benzin istasyonu ile bir park bulunmaktadır. Yollar genellikle gidiş-geliş şeklindedir.



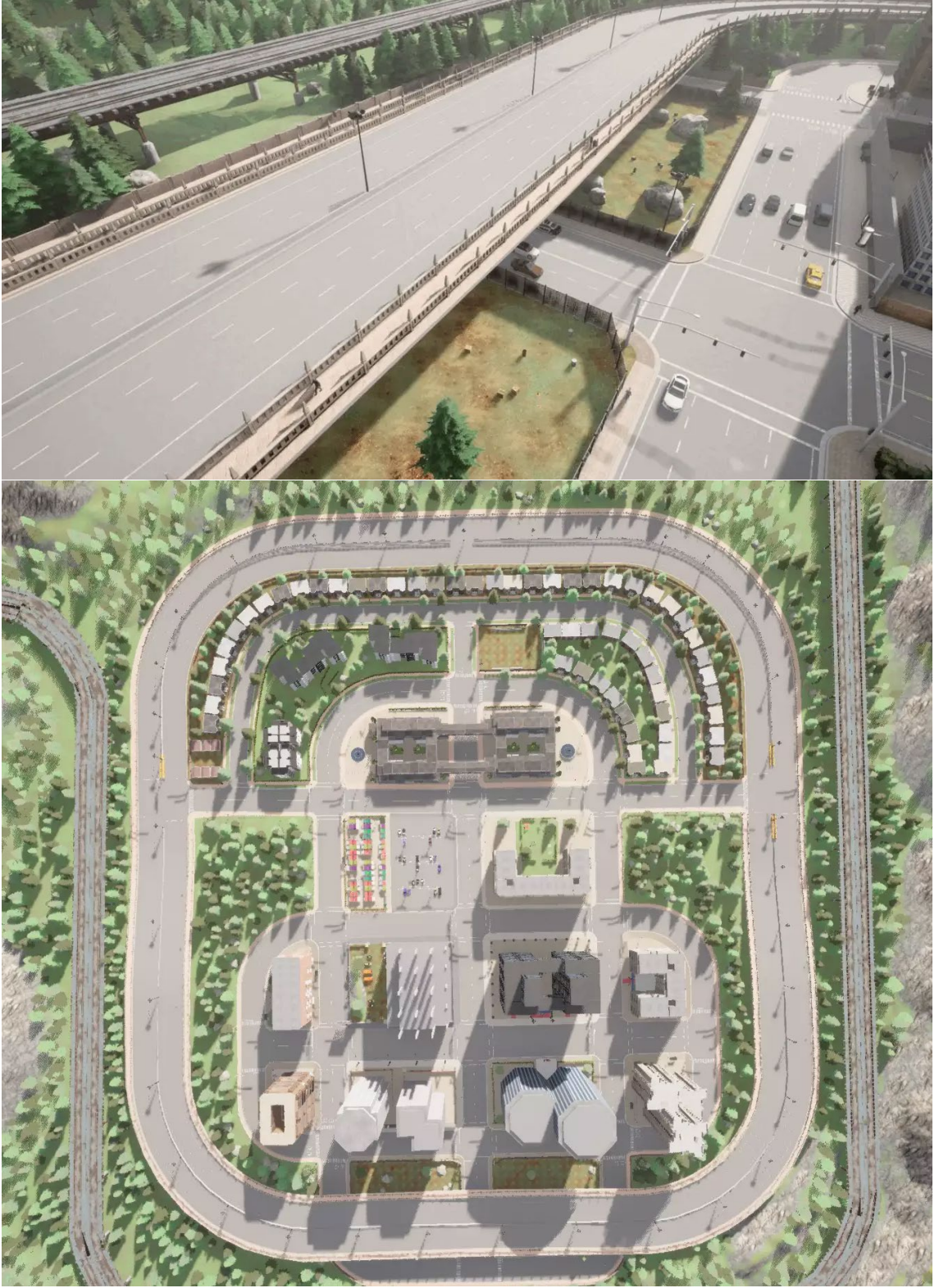
Şekil 6.4 *Town03* kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)

Town03, *Town02*'ye kıyasla biraz daha büyüktür (Şekil 6.4). Hem çevresel görüntüler hem de yollar bakımından biraz daha şehir merkezi görünümü vardır. Yol bünyesinde, döner kavşak, alt geçitler ve üst geçitler gibi farklı yol ağı özelliklerini içerir. Kasabada ayrıca yükseltilmiş bir metro yolu ve inşaat halindeki büyük bir bina bulunmaktadır. Yollar genellikle çok şeritlidir ve bölünmüş yollar da içermektedir.



Şekil 6.5 *Town04* kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)

Town04, karla kaplı bir dağı ve küçük bir kasabayı çevrelemektedir (Şekil 6.5). Dağlık alanda ağaçlar mevcuttur. Yollar genellikle çok şeritlidir.



Şekil 6.6 *Town05* kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)

Town05 etrafında yükseltilmiş bir otoyol dönmektedir (Şekil 6.6). Geniş çok şeritli yollar ve kavşaklar vardır. Ortada kentsel bir alan bulunmaktadır. Arka planda ise dağlık ve ağaçlık alanlar mevcuttur.



Şekil 6.7 *Town06* kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)

Town06, 4-6 şeritli ve uzun düzlüklere sahip yolları olan bir ortamdır (Şekil 6.7). Düz yolda ilerlerken şerit değiştirme ve diğer sürücülerin davranışı üzerine yapılacak çalışmalar için uygun olduğundan veri kümesine eklenmiştir. Orta da çok küçük bir kentsel alanı olsa da genellikle etrafı ağaçlarla kaplı bir yol yapısına sahiptir.



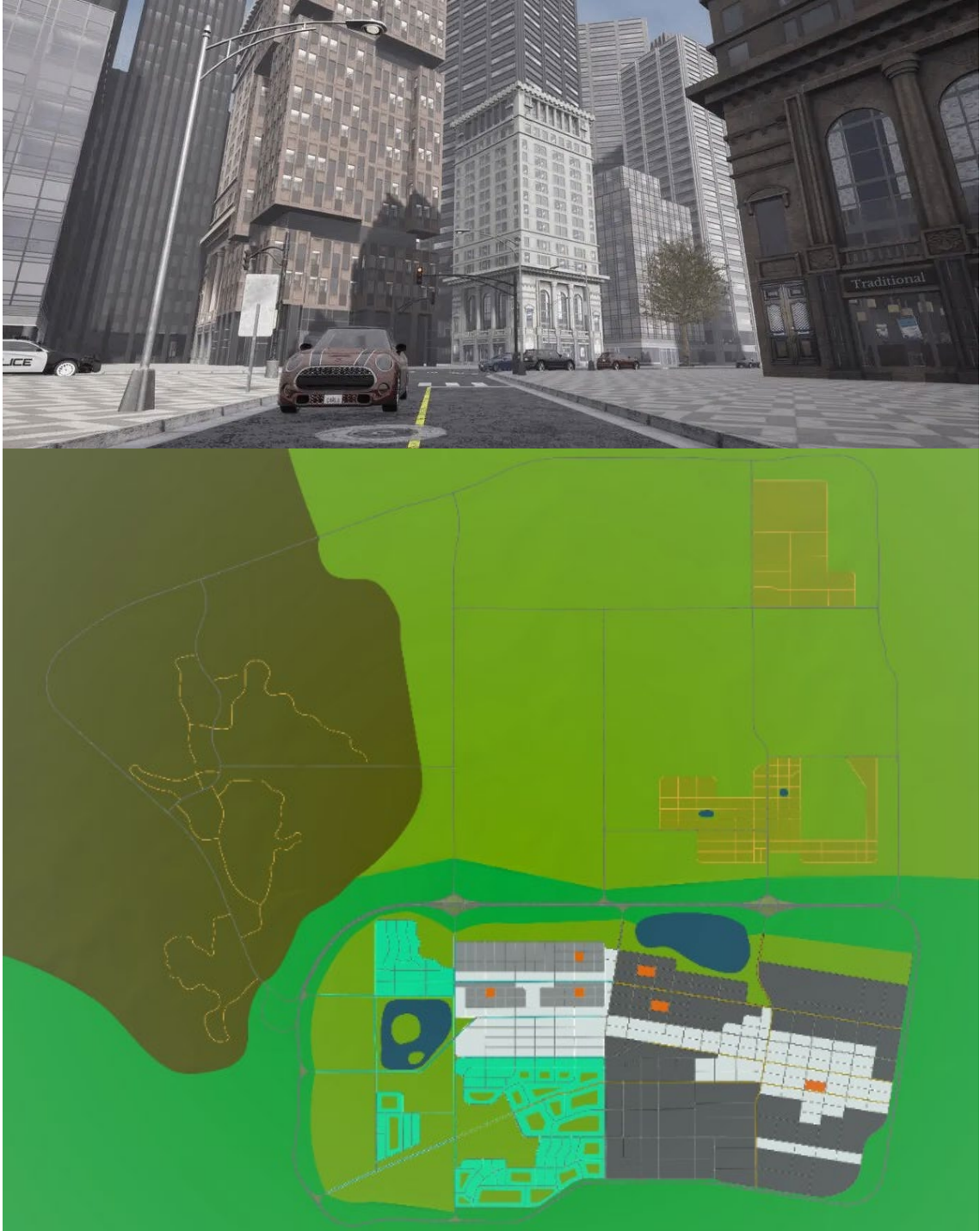
Şekil 6.8 *Town07* kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)

Town07, kırsal bir çevre üzerine kurulmuştur (Şekil 6.8). Kentte görülen bina yapıları yoktur. Mısır tarlaları, ağaçlar ve çiftlik yapıları, silolar vardır. Bazı yolların kenarlarında ayırıcı şerit çizgileri dahi yoktur. Yol kenarında hemen mısır tarlaları başlar. Dolayısıyla zorlu denebilecek sürüş sahneleri içerir. Kasaba içindeki yollar gidiş-geliş şeklindedir.



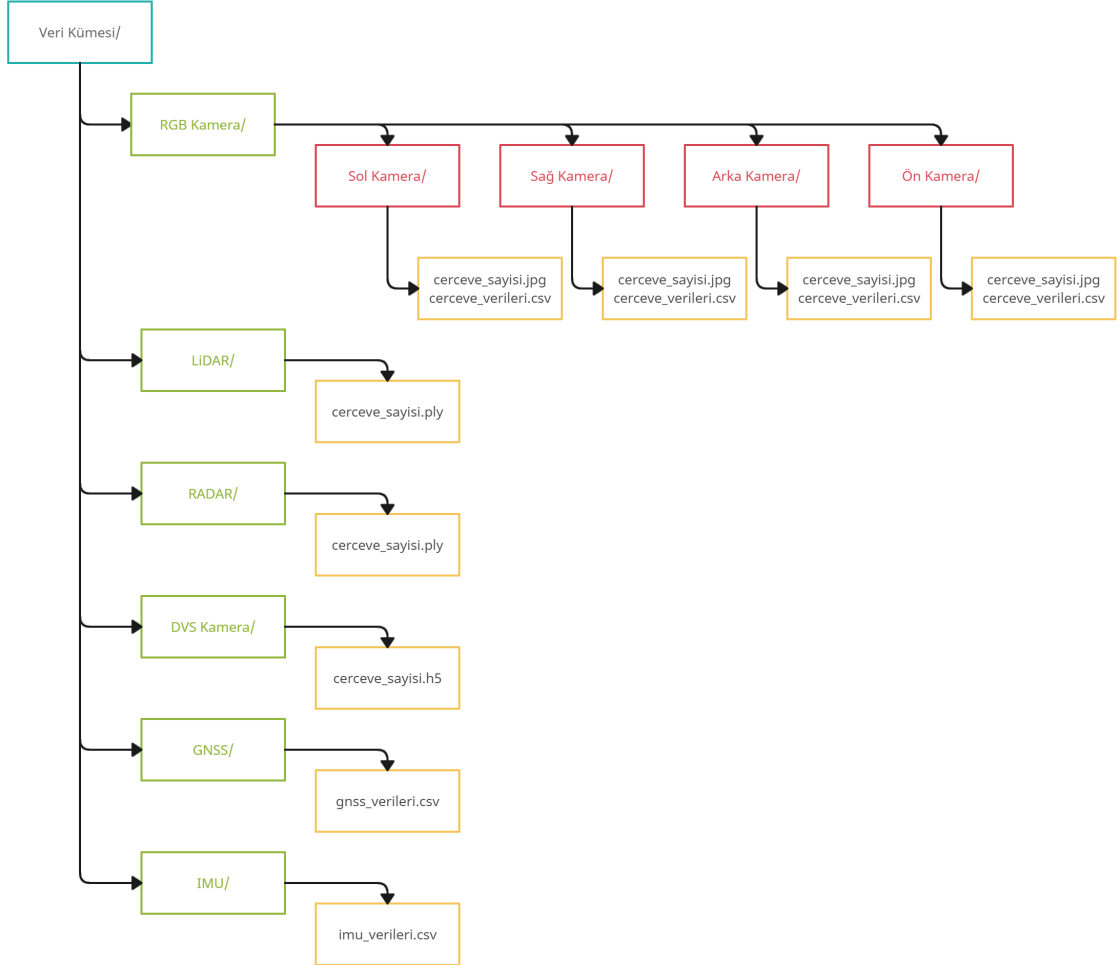
Şekil 6.9 Town10 kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)

Town10, sahil kenarında yürüyüş yolları, apartman blokları ve farklı tipte binalar içeren bir kasabadır (Şekil 6.9). Bazı yolların kenarları ağaçlarla çevrilidir, bulvarların kenarlarında bulunan gökdelenler ve endüstriyel binalarla çevresel görüş açısından diğer kasabalara göre farklı bir karışım sunmaktadır. Town 10'un yol ağı, çeşitli şerit işaretleri, geçitler ve sinyal türleriyle farklı kavşak düzenlemeleri barındırır.



Şekil 6.10 *Town12* kasabasının genel görünüşü (üst) ve kuş bakışı görüntüsü (alt)

Town12, 10×10 km² boyutlarında diğer kasabalara göre çok daha büyük bir haritaya sahiptir (Şekil 6.10). Yol düzeni Amerika Birleşik Devletleri'nin Teksas eyaletinden esinlenilerek yapılmıştır. Kentsel yerleşim alanında farklı yapıda çeşitli yollar ve kasabayı çevreleyen bir otoyol sistemine sahiptir. Mimari yapı da değişiklik gösterir. Kasabada, yüksek binalar, az katlı konutlar, kırsal alandaki evler olmak üzere çeşitli yapıda binalar bulunur.



Şekil 6.11 Veri kümesinin dosya düzeni ve veri yapısı

Toplam sekiz adet kasabadan veri toplanmıştır. Her kasabadan aynı miktarda veri toplanmamış, toplanan veri miktarında kasabaların büyüklükleri belirleyici olmuştur. Kasabalarda, büyüklüğe göre yaya sayısı 30-60 arasında ve araç sayısı 40-80 arasında belirlenmiştir. Veri toplanan sekiz şehrin hepsinde farklı aydınlanma ve farklı hava koşulları tercih edilmiştir. Bu aydınlanma-hava koşulları çiftleri, <gündüz-açık>, <gündüz- bulutlu, ıslak zemin>, <gündüz- orta yağmurlu>, <gündüz- şiddetli yağmurlu>.

<gün batımı- açık>, <gün batımı- ıslak zemin>, <gün batımı- orta yağmurlu>, <gün batımı- şiddetli yağmurlu>, <gece-açık>, <gece- bulutlu, ıslak zemin>, <gece- orta yağmurlu>, <gece- şiddetli yağmurlu> şeklindedir. Burada üç farklı aydınlanma koşulunda ve dört farklı hava şartlarında veri toplanmıştır. Toplanan verinin detayları,

Tablo 6.7’de sunulmuştur.

6.3 Veri Yapısı

Oluşturulan veri kümesi, farklı sensör türlerinden elde edilen çeşitli veri formatlarını içermektedir. Kamera görüntüleri JPG formatında sunulurken, bu görüntülere karşılık gelen kare adları ve zaman damgaları bir CSV dosyasında bulunmaktadır. LiDAR verisi ply (Polygon) dosya formatında kaydedilirken, radar ve DVS kamera verisi hdf5 formatında sunulmuş, aynı zamanda H5 formatında kaydedilip gzip sıkıştırma yöntemiyle boyutu küçültülmüştür. GPS ve IMU verileri ise CSV dosyası olarak sunulmaktadır. Oluşturulan veri kümesinin dosya yapısı ve veri formatları Şekil 6.11’da ayrıntılı olarak gösterilmiştir.

Literatürdeki diğer CARLA sürüş simülatörü tabanlı veri kümeleri incelendiğinde, RapidSense veri kümesinin çeşitli avantajlara sahip olduğu görülmektedir (Tablo 6.8). Özellikle sensör sayısı bakımından en yakın veri kümesi olan CarlaScenes (Kloukiniotis et al., 2022) beş sensöre ait veri sunarken RapidSense veri kümesi dokuz adet sensörden elde edilen veriyi içermektedir. Ayrıca toplanan veri miktarı bakımından da en yakın veri kümesinin 2,5 katı büyüklüktedir.

Veri kümesinin toplandığı kasaba sayısı bakımından da yine en üstün veri kümesi RapidSense’dir. Özellikle verinin sekiz farklı kasabadan toplanmış olması, araştırmacılara geniş bir veri çeşitliliği sunmaktadır. Sunulan hava ve aydınlanma şartları, diğer veri kümeleriyle kıyaslandığında üstün özelliklere sahip olması, RapidSense veri kümesini araştırmacılar için çekici kılan önemli bir faktördür.

RapidSense veri kümesinin geniş sensör yelpazesi ve çeşitliliği, araştırmacılara farklı alanlarda kapsamlı çalışmalar yapma imkânı tanıyacak ve otonom sürüş sistemleri üzerine yapılan araştırmalara önemli katkılarda bulunacaktır. Bu veri kümesi, az sayıda ve kısıtlı mekanlarda toplanan veri kümeleriyle karşılaştırıldığında, yüksek veri çeşitliliği ve kalitesi ile öne çıkmaktadır. Bu da araştırmacıların RapidSense veri kümesini tercih etmelerine ve kendi alanlarında daha derinlemesine çalışmalarına olanak sağlayacaktır.

Tablo 6.7 Farklı aydınlanma durumu ve farklı hava şartlarına göre kasabalardan toplanan veri miktarı

Kasaba	Gündüz				Gün batımı				Gece				Toplam
	Açık	Bulutlu, ıslak zemin	Orta yağmurlu	Şiddetli yağmurlu	Açık	Bulutlu, ıslak zemin	Orta yağmurlu	Şiddetli yağmurlu	Açık	Bulutlu, ıslak zemin	Orta yağmurlu	Şiddetli yağmurlu	
<i>Town02</i>	3.000	1.000	1.000	1.000	2.250	750	750	750	1500	500	500	500	13.500
<i>Town03</i>	6.000	2.000	2.000	2.000	4.500	1.500	1.500	1.500	3.000	1.000	1.000	1.000	27.000
<i>Town04</i>	4.500	1.500	1.500	1.500	3.375	1.125	1.125	1.125	2.250	750	750	750	20.250
<i>Town05</i>	4.500	1.500	1.500	1.500	3.375	1.125	1.125	1.125	2.250	750	750	750	20.250
<i>Town06</i>	4.500	1.500	1.500	1.500	3.375	1.125	1.125	1.125	2.250	750	750	750	20.250
<i>Town07</i>	9.000	3.000	3.000	3.000	6.750	2.250	2.250	2.250	4.500	1.500	1.500	1.500	40.500
<i>Town10</i>	6.000	2.000	2.000	2.000	4.500	1.500	1.500	1.500	3.000	1.000	1.000	1.000	27.000
<i>Town12</i>	9.000	3.000	3.000	3.000	6.750	2.250	2.250	2.250	4.500	1.500	1.500	1.500	40.500
Toplam	46.500	15.500	15.500	15.500	34.875	11.625	11.625	11.625	23.250	7.750	7.750	7.750	209.250

Tablo 6.8 CARLA ile oluşturulmuş veri kümelerine ait özet bilgiler

Veri kümesi	Sensörler	Veri sayısı (kare) / Kasaba	Hava şartları	Zaman	Çevre
OmniScape (Sekkat et al., 2020)	2×Balık gözü kamera, 2×Katadioptrik kamera	/ <i>Town 01-05</i>	Açık, yağmurlu	Gündüz, gün batımı	kentsel, tünel, otoyol, karmaşık sahneler, sonsuz döngü
KITTI-CARLA (Deschaud, 2021)	2×Kamera, 1×LiDAR	35.000 / <i>Town 01-07</i>		Gündüz	kentsel, kırsal, tünel, otoyol, karmaşık sahneler, sonsuz döngü
Steering Dataset (Hassan, 2021)	1×Kamera	86.500 / <i>Town 01-03</i>	Açık, bulutlu, yağmurlu	Gündüz, gece	kentsel, tünel, karmaşık sahneler, sonsuz döngü
THI dataset (Alan Naoto Tabata, 2020)	1×Kamera	19.500 / <i>Town 01-05</i>	Açık, bulutlu, yağmurlu	Gündüz, gün batımı, gece	kentsel, tünel, karmaşık sahneler, sonsuz döngü
CarlaScenes (Kloukiniotis et al., 2022)	1×Kamera, 2×LiDAR, 1×GPS, 1×IMU	Erişime açık değil	Açık, bulutlu, yağmurlu, ıslak zemin	Gündüz, gün batımı, gece	kentsel, tünel, otoyol, karmaşık sahneler, sonsuz döngü
RapidSense (Bu çalışma)	4×Kamera, 1×LiDAR, 1×Radar, 1×DVS kamera, 1×GPS, 1×IMU	209.250 / <i>Town 02-07, 10, 12</i>	Açık, bulutlu-ıslak zemin, orta yağmurlu, şiddetli yağmurlu	Gündüz, gün batımı, gece	kentsel, kırsal, tünel, otoyol, karmaşık sahneler, sonsuz döngü

7. SONUÇLAR, TARTIŞMA VE ÖNERİLER

Bu tez çalışmasında, monoküler kamera görüntüsüne dayalı olarak geliştirilen görsel algılama odaklı bir model ile sürüş sahnelerinin analiz edilerek kontrol çıktılarının üretilmesi amaçlanmıştır. Bu kapsamda RGB görüntü, derinlik görüntüsü ve anlamsal olarak bölümlenmiş görüntü yardımıyla otonom sürüş görevlerinde direksiyon açısı ve hız tahmini yapabilen bir model önerilmiştir. Özellikle, tek bir sensör olan kamera görüntüsü üzerinden elde edilen bu görüntülerin entegrasyonu ile geliştirilen model, sadece görsel verilere dayanarak direksiyon açısı ve hız tahmini yapabilme yeteneğini göstermiştir. Bu çok girdili yapılandırmanın, modelin genel tahmin doğruluğunu önemli ölçüde artırdığı gözlemlenmiştir. Çalışma, sadece kamera görüntüsüne dayalı derin öğrenmeye odaklanarak, güvenli görsel sürüş şartlarının oluşturulabileceğini ve bu sayede maliyeti düşürerek sürüş güvenliğinin artırılabilceğini göstermiştir. Bu çalışmanın temel motivasyon kaynağı, insan sürücülerin yüksek maliyetli sensörlere ihtiyaç duymadan sadece görsel algılarına dayalı olarak gece ve gündüz şartlarında araç kullanabilme potansiyeli olmuştur. Elde edilen sonuçlar, insan görme sistemine benzer şekilde oluşturulan (doku, derinlik ve anlamsal bölümlenmiş alan algılayabilen) bir görsel algı modeli ile kontrol komutlarının yüksek doğrulukla tahmin edilebileceğini ortaya koymuştur.

7.1 Sonuçlar ve Tartışma

Modelin geliştirilme aşamasında uygulanan eksiltme çalışması, her bir bileşenin gerekliliği ve katkısı hakkında önemli sonuçlar sağlamıştır. İlk olarak modelde kullanılan görüntü girdilerinin gerekliliği sorgulanmış, en başarılı sonuçların her üç girdinin de kullanıldığı durumda elde edildiği gözlemlenmiştir. Her bir girdi olmadan yapılan analizler, en kötü sonuçların RGB modülünün olmadığı durumda ortaya çıktığını ve hem hızı hem de direksiyon açısını olumsuz etkilediğini ortaya koymuştur. Daha sonra farklı Özellik Kümesi Uzunlukları (ÖKU) seçilerek bireysel alt modellerin ana modele katkıları incelenmiştir. RGB girdisinin ağırlığı (ÖKU değeri ile) diğer girdilere kıyasla artırıldığında hem direksiyon açısı hem de hız için en başarılı sonuçlar elde edilmiştir. Böylece RGB görüntü girdisinin sistemdeki en önemli bileşen olduğu ortaya çıkmaktadır. Ancak tek başına RGB görüntü ile elde edilen sonuçların da zayıf kaldığı, derinlik ve semantik segmentasyon ile desteklendiği takdirde daha düşük hatalar elde edildiği

görülmüştür. Böylece RGB görüntü yanında derinlik ve bölümlenmiş görüntü ile desteklenmiş ve RGB ağırlığı artırılmış üç girdili bir yapının daha başarılı sonuçlar ortaya koyduğu sonucuna varılmıştır. Böylelikle başlangıçta savunulan, insan görme sistemine benzer yapıda, renk dokusu, derinlik ve anlamsal alanları algılayabilen bir modelin başarısı doğrulanmıştır.

LSTM biriminin de dahil edilmesiyle, modelin performansında belirgin bir iyileşme elde edilmiştir. Bu, modelin zaman içindeki dizi bilgisini başarıyla işleyebildiğini ve bu bilgiyi çok girdili yapıya başarılı bir şekilde entegre edebildiğini göstermektedir.

Modelin her bir girdisini işleyecek olan ESA ağına belirlenmesi için de farklı özellik çıkartma yöntemleri denenmiş ve EfficientNetV2 ağı ile en başarılı sonuçlar elde edilmiştir. Bu ağ, güncel literatürdeki en başarılı ağlardan biridir ve parametre sayısı diğer birçok modele kıyasla daha azdır. Görsel girdilerden, ESA ağları ile elde edilen özellik vektörleri için de ağırlık katsayıları belirlenmiş ve farklı versiyonlar denenmiştir. Görsel girdilerin eşit katsayı ile çarpıldığı, LSTM çıktısının ise tüm görsel girdilerin yarısı büyüklüğünde bir ağırlığa sahip olduğu versiyon en başarılı sonuçların alındığı modelde gözlenmiştir.

Modelde farklı kayıp fonksiyonlarını deneyebilmek amacıyla ağırlıklı toplam şeklinde bir kayıp fonksiyonu tanımlanmış ve direksiyon açısı ile hız için elde edilen kayıp fonksiyonları ağırlıklandırılarak toplanmıştır. Ancak en başarılı sonucun eşit ağırlıklı kayıp fonksiyonu ile elde edildiği gözlemlenmiştir.

Ayrıca, veri normalleştirme ve veri artırma dahil olmak üzere kapsamlı ön işleme tekniklerinin kullanılmasıyla modelin bazı veri kümeleri için sağlamlığı ve genelleştirilebilirliği artırılmıştır. Bu teknikler, modelin farklı koşullarda daha tutarlı ve güvenilir performans sergilemesine yardımcı olmuştur. Udacity veri kümesinde uygulanan veri artırma teknikleri modelin başarısını artırmıştır. Ancak CARLA ve Sully Chen kümelerinde, veri artırma yapılmasının sonuca pek katkısı olmamıştır. Veri normalleştirmenin de 1'den büyük değerler için fayda sağladığı, ama 1 civarındaki değerler için etkili olmadığı gözlenmiştir. Örneğin CARLA veri kümesindeki direksiyon açısı değerleri için normalleştirmenin bir katkısı olmamıştır.

Model mimarisinin detaylı tasarımı, güncel sinir ağı bileşenleri ve optimizasyon algoritmalarının seçimi, görsel otonom sürüş görevinin karmaşıklıklarını etkili bir şekilde ele almıştır. Bu yöntemler, deneysel çalışmalarla desteklenerek, önerilen modelin gerçek dünya uygulamalarında kullanılabilirlik açısından önemli bir adım olduğunu göstermiştir.

Geliştirilen model, oluşturulan yapının başarısının veri kümesine özgü olmadığını, genelleme yeteneğinin yüksek olduğunu göstermek adına üç farklı veri kümesi üzerinde test edilmiştir.

Udacity veri kümesi, gerçek bir araçla toplandığı için ve hem hız hem de direksiyon açısı sağladığı için tez çalışmasında temel veri kümesi olarak kullanılmıştır. Bu veri setindeki görüntüler, günün farklı saatlerinde toplandığı için koyu gölgeler, aşırı parlak alanlar, farklı renkte yol rengi gibi tespit açısından oldukça zorlu sürüş sahneleri içerir. Veri kümesi, değişik yoğunlukta trafik, virajlı veya düz yollar içeren geniş yelpazede bir veri sunar. Ayrıca bu durum, otonom sürüş için gerçek dünya senaryolarını temsil etme açısından önemlidir. Model bileşenlerinin seçimi ve değerlendirmesinin yapıldığı eksiltme çalışmaları da bu veri seti üzerinde yapılmıştır.

Udacity veri kümesi üzerinde yapılan testler hem direksiyon açısı hem de hız için literatürdeki en başarılı sonuçları vermiştir. Yapılan tahminler incelendiğinde, bazı dönüşlerde, yol görüntüsünün neredeyse kaybolduğu durumlarda veya aşırı güneşten ya da aşırı karanlıktan yol çizgilerinin görünmediği gibi zorlu durumlarda dahi modelin başarılı tahmin yaptığı görülmüştür. Bu tür durumlarda eksikliği oluşan girdilerin boşluğunu bazen diğer girdiler doldurmuş bazen de LSTM verisi modelin doğru tahmin yapmasına yardımcı olmuştur. Örneğin yol görüntüsünün neredeyse tamamen kaybolduğu bazı durumlarda LSTM verisinin bu eksikliği tamamladığı veya aşırı parlak ya da karanlık nedeniyle çok bozuk bir RGB görüntü alındığında bunun derinlik ve segmentasyon görüntüsü ile desteklendiği görülmüştür. Böylece modelin aslında tek bir sensöre ya da girdiye bağlı değilmiş gibi hareket ettiği gözlenmiştir. Eksiltme çalışmaları sırasında özellikle daha az sayıda LSTM hücresinin kullanılmasının hız tahminini olumsuz etkilediği görülmüştür. Ancak bu sayının artırılmasının da her iki tahmine olumsuz etkisi olmuştur. Udacity üzerinde yapılan direksiyon açısı tahmininde, açının zıt yöne değişmeye başladığı noktalarda, yani aracın istikametini değiştirmeye başladığı yerlerde, direksiyon açısının çok kısa süreli olarak fazla miktarda saptığı ancak hemen bu durumu toparlayarak daha düşük sapmalarla bu dönüşü başarılı bir şekilde yaptığı görülmüştür.

Önerilen model ikinci olarak CARLA veri kümesi üzerinde test edilmiştir. CARLA veri kümesi sürüş simülatörü ile toplandığı için görüntüleri gerçek dünya verisi değildir. Ancak kamera görüntüleri, farklı araçlar, yayalar, binalar gibi gerçeğe çok yakın üretilmiş içerikler bulundurulur. Sürüş simülatörlerinin avantajı, veri sayısının ve türünün sınırlı olmamasıdır. Ayrıca CARLA sürüş simülatörü, araç veya yaya sayısının

belirlenmesi, hava koşullarının, gün ışığının veya veri toplanacak haritanın seçimi gibi pek çok esneklik sunar. Bu nedenle Udacity veri kümesinden sonra CARLA sürüş simülatörü ile de bir veri kümesi oluşturulup model denenmiştir. Oluşturulan bu veri kümesi de Udacity gibi görüntülerle birlikte direksiyon açısı ve hız bilgisi içermektedir. CARLA ortamında derinlik görüntüsü de hazır olarak sunulduğu için, modelde kullanılan Monodepth2 metodunun performansını değerlendirmek amacıyla CARLA'nın sağladığı derinlik görüntüleri de toplanmıştır.

Literatürdeki pek çok CARLA çalışmasının aksine, veriler farklı kasabalardan toplanmış, eğitim, doğrulama ve test verilerinin hepsi ayrı kasabalardan alınmıştır. Kasabaların yol yapıları, trafik düzenleri ve çevresel görüntüleri birbirinden tamamen farklı olduğu için bu hem veri çeşitliliği oluşmasına hem de modelin, daha önce görülmemiş sahnelerle test verisi üzerinde tahmin yapmasına neden olmuştur.

Oluşturulan veri kümesi üzerinde hem CARLA'nın hem de modelin monoküler derinlik tahmin metodunun sağladığı derinlik görüntüleri ile direksiyon açısı ve hız tahmini yapılmıştır. Her iki yöntemle elde edilen sonuçlar birbirine çok yakın çıkmıştır. Bu da kullanılan monoküler derinlik tahmin yönteminin başarılı bir şekilde model içindeki görevini yerine getirdiğini göstermektedir. CARLA veri kümesi üzerinde elde edilen hata sonuçları da Udacity değerlendirmesinde olduğu gibi literatürdeki en düşük hata değerleri olmuştur. Udacity'den sonra farklı bir veri kümesinde daha en başarılı sonuçların alınması, modelin genelleme yeteneğinin yüksek olduğunu göstermektedir. Test verisinin eğitim verisiyle benzeşmediği durumlarda dahi modelin en başarılı sonuçları üretmesi, modelin veriye esnek ve etkili bir şekilde adapte olabildiğini ortaya koymaktadır.

Son olarak, önerilen yöntem Sully Chen veri kümesi ile de test edilmiştir. Bu veri kümesi, önceki veri setlerinden farklı olarak hız bilgisi sağlamamaktadır ve genellikle verinin büyük bir kısmı şehir dışı olarak tanımlanabilecek yollarda toplanmıştır. Sully Chen veri kümesi, Udacity gibi günün farklı saatlerinde birkaç parça şeklinde değil tek seferde kesintisiz olarak toplanmıştır. Toplanan tek parça veri, eğitim ve test verisi olarak bölünmüştür. Bu veri kümesindeki sürüş sahneleri de Udacity veri kümesinde olduğu gibi güçlü ışıklar ve koyu gölgeler içerir.

Sully Chen veri seti üzerinde yapılan direksiyon açısı tahminleri, Udacity ve CARLA üzerinde yapılan tahminlerden daha başarılı olmuştur. Bu başarıda, veri kümesinin tek seferde toplanmış olması ve eğitim ile test verisinin benzer sürüş sahneleri

içermesinin etkisi büyüktür. Tahmin edilen direksiyon açısı değerleri gerçek direksiyon açıları ile büyük oranda uyumludur.

Bu tez çalışması, modelin geliştirilme süreci boyunca yapılan analizlerin, her bir bileşenin önemini ve katkısını aydınlatarak önemli sonuçlar ortaya koyduğunu göstermiştir. Yapılan deneyler, geliştirilen modelin çoklu görüntü girdilerini etkili bir şekilde kullanarak sürüş sahnelerini başarılı bir şekilde analiz ettiğini ve kontrol çıktıları üretebildiğini ortaya koymuştur. Modelin genelleme yeteneği ve farklı veri kümelerinde gösterdiği başarılar, gerçek dünya koşullarında otonom sürüş sistemlerinin geliştirilmesinde umut vaat eden bir potansiyel sunmaktadır. Bu çalışma, monoküler kamera görüntüsüne dayalı derin öğrenme algoritmalarının kullanımının, karmaşık sürüş senaryolarında güvenli ve etkili sürüş kararları üretmede potansiyelini vurgulamaktadır.

Bu tez çalışması kapsamında, CARLA sürüş simülatörü kullanılarak kapsamlı bir sentetik veri kümesi oluşturulmuş ve topluluğun açık erişimine sunulmuştur. Literatürde CARLA ile oluşturulmuş mevcut sentetik veri kümelerinin kısıtlı olması, veri çeşitliliğinin yetersiz kalması nedeniyle sunulan geniş kapsamlı veri kümesinin, araştırmacılar tarafından tercih edileceği ve otonom sürüş alanındaki eksikliği gidererek bu alandaki çalışmalara katkı sağlayacağı düşünülmektedir.

7.2 Öneriler

Gelecekteki çalışmalar, modelin daha geniş veri kümeleri ve değişken sürüş koşullarıyla başa çıkma yeteneğini, gerçek zamanlı uygulamalarda performansını ve güvenilirliğini daha da geliştirmeyi hedeflemelidir. Modelin görüntü girdileri daha başarılı sonuçlar üretecek girdilerle değiştirilebilir. Ayrıca, eksiltme çalışmalarının kapsamı artırılarak modele yapılacak ince ayarlar neticesinde performansı biraz daha artırılabilir. Geçmiş zamandaki hız ve direksiyon açısı dizisini inceleyen LSTM yapısı yerine daha uzun bir geçmişi tutabilen görsel dönüştürücüler (Visual Transformers) denenebilir. Görsel bilgi girişini artırmak ve daha yüksek doğruluğa sahip kontrol çıktıları üretmek için çoklu kameralar kullanılabilir. Eğitimler, daha büyük veri kümeleri ve daha çeşitli verilerle yapılmalı, bu geniş veri kümesinde eğitilen model dünyanın her yerinde başarılı tahminler üretebilmelidir.

Oluşturulan veri kümesi, kullanım alanlarına özgü olarak özelleştirilip genişletilebilir ve farklı versiyonlar şeklinde araştırmacıların kullanımına sunulabilir.

KAYNAKLAR

- Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., ... Zheng, X., 2016, TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems, *ArXiv ID: Cs.DC/1603.04467*.
- Abbas, W., Khan, M. F., Taj, M., & Mahmood, A., 2021, Statistically correlated multi-task learning for autonomous driving, *Neural Computing and Applications*, 33(19), 12921–12938. <https://doi.org/10.1007/s00521-021-05941-8>
- Abou-Hussein, M., Muller, S. H., & Boedecker, J., 2019, Multimodal spatio-temporal information in end-to-end networks for automotive steering prediction, *Proceedings - IEEE International Conference on Robotics and Automation, 2019-May*, 8641–8647. <https://doi.org/10.1109/ICRA.2019.8794410>
- Agarwal, P., de Beaucorps, P., & de Charette, R., 2021, Sparse Curriculum Reinforcement Learning for End-to-End Driving, *ArXiv*. <http://arxiv.org/abs/2103.09189>
- Aggarwal, C. C., 2018, *Neural Networks and Deep Learning*, Springer. <https://doi.org/10.1007/978-3-319-94463-0>
- Alan Naoto Tabata, 2020, *Object Detection and Monocular Depth Estimation with a Custom Synthetic Automotive Dataset* [Technische Hochschule Ingolstadt]. <https://www.thi.de/forschung/carisma/c-isafe/thi-synthetic-automotive-dataset/>
- Anonymous, 2018, *WHO Global status report on road safety 2018*. <https://www.who.int/publications/i/item/9789241565684>
- Azam, S., Munir, F., Rafique, M. A., Sheri, A. M., Hussain, M. I., & Jeon, M., 2021, N2C: Neural Network Controller Design Using Behavioral Cloning, *IEEE Transactions on Intelligent Transportation Systems*, 22(7), 4744–4756. <https://doi.org/10.1109/TITS.2020.3045096>
- Behl, A., Chitta, K., Prakash, A., Ohn-Bar, E., & Geiger, A., 2020, Label Efficient Visual Abstractions for Autonomous Driving, *ArXiv*. <http://arxiv.org/abs/2005.10091>
- Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., Zieba, K., Testa, D. Del, Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., ... Zieba, K., 2016, End to End Learning for Self-Driving Cars, *ArXiv ID: Cs.CV/1604.07316*. <http://arxiv.org/abs/1604.07316>
- Bozkaya, F., Yusefi, A., Tığlıoğlu, S., Kaya, A. K., Kazancı, O., Akmaz, M. Y., Durdu, A., & Sungur, C., 2021, Otonom Sistemlerde Veri Çoğaltma Yöntemleri Kullanılarak İyileştirilmiş Gerçek Zamanlı Nesne Tespiti, *Avrupa Bilim ve Teknoloji Dergisi*, 30, 83–87. <https://doi.org/10.31590/ejosat.1006408>
- Brock, A., De, S., Smith, S. L., & Simonyan, K., 2021, High-Performance Large-Scale Image Recognition Without Normalization, *Proceedings of Machine Learning Research*, 139, 1059–1071.
- Cai, P., Sun, Y., Chen, Y., & Liu, M., 2019, Vision-Based Trajectory Planning via Imitation Learning for Autonomous Vehicles, *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*, 2736–2742. <https://doi.org/10.1109/ITSC.2019.8917149>
- Cai, P., Wang, S., Sun, Y., & Liu, M., 2020, Probabilistic End-to-End Vehicle Navigation in Complex Dynamic Environments with Multimodal Sensor Fusion, *IEEE Robotics and Automation Letters*, 5(3), 1–1. <https://doi.org/10.1109/LRA.2020.2994027>

- Caltagirone, L., Bellone, M., Svensson, L., & Wahde, M., 2019, LIDAR–camera fusion for road detection using fully convolutional neural networks, *Robotics and Autonomous Systems*, 111, 125–131. <https://doi.org/10.1016/j.robot.2018.11.002>
- Chen, C., Rosa, S., Miao, Y., Lu, C. X., Wu, W., Markham, A., & Trigoni, N., 2019, Selective Sensor Fusion for Neural Visual-Inertial Odometry, *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019-June, 10534–10543. <https://doi.org/10.1109/CVPR.2019.01079>
- Chen, C., Seff, A., Kornhauser, A., & Xiao, J., 2015, DeepDriving: Learning Affordance for Direct Perception in Autonomous Driving, *2015 IEEE International Conference on Computer Vision (ICCV)*, 2722–2730. <https://doi.org/10.1109/ICCV.2015.312>
- Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L., 2018, DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4), 834–848. <https://doi.org/10.1109/TPAMI.2017.2699184>
- Chen, L., Wu, P., Chitta, K., Jaeger, B., Geiger, A., & Li, H., 2023, *End-to-end Autonomous Driving : Challenges and Frontiers*, JUNE.
- Chen, Z., & Huang, X., 2017, End-to-end learning for lane keeping of self-driving cars, *2017 IEEE Intelligent Vehicles Symposium (IV)*, Iv, 1856–1860. <https://doi.org/10.1109/IVS.2017.7995975>
- Chi, L., & Mu, Y., 2017, Deep Steering: Learning End-to-End Driving Model from Spatial and Temporal Visual Cues, *ArXiv:1708.03798v1 [Cs.CV]*, 9–16. <https://doi.org/10.1145/3132734.3132737>
- Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y., Merri, B. Van, & Fellow, C. S., 2014, Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing ({EMNLP})*, 1724–1734. <https://doi.org/10.3115/v1/D14-1179>
- Chollet, F., 2015, *Keras*, <https://keras.io/> [Ziyaret Tarihi: 1 Şubat 2024] .
- Chollet, F., 2017, *Deep Learning with Python*, Manning.
- Codevilla, F., Müller, M., López, A., Koltun, V., & Dosovitskiy, A., 2018, End-to-End Driving Via Conditional Imitation Learning, *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 4693–4700. <https://doi.org/10.1109/ICRA.2018.8460487>
- Codevilla, F., Santana, E., Lopez, A. M., & Gaidon, A., 2019, Exploring the Limitations of Behavior Cloning for Autonomous Driving, *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Cvc, 9329–9338.
- Cultrera, L., Seidenari, L., Becattini, F., Pala, P., & Del Bimbo, A., 2020, Explaining autonomous driving by learning end-to-end visual attention, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2020-June, 1389–1398. <https://doi.org/10.1109/CVPRW50498.2020.00178>
- Deschaud, J.-E., 2021, *KITTI-CARLA: a KITTI-like dataset generated by CARLA Simulator*, 1–2. <http://arxiv.org/abs/2109.00892>
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N., 2020, An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, *International Conference on Learning Representations*. <https://openreview.net/forum?id=YicbFdNTTy>
- Dosovitskiy, A., Ros, G., Codevilla, F., López, A., Koltun, V., Lopez, A., & Koltun, V., 2017, CARLA: An Open Urban Driving Simulator, *Proceedings of the 1st Annual Conference on Robot Learning, CoRL*, 1–16.

- Dowling, K., Guzikowski, R., Ladd, J., Pangels, H., Singh, S., & Whittaker, W. (Eds.), 1987, *NAVLAB: An Autonomous Navigation Testbed* (Number CMU-RI-TR-87-24). https://doi.org/10.1007/978-1-4613-1533-9_12
- Dozat, T., 2016, Incorporating Nesterov Momentum into Adam, *ICLR Workshop*, 1, 2013–2016.
- Eric Espi  , Christophe Guionneau, Bernhard Wymann, Christos Dimitrakakis, R  mi Coulom, & Andrew Sumner, 2005, *TORCS, The Open Racing Car Simulator*. <https://torcs.sourceforge.net/>
- Fukushima, K., 1975, Cognitron: A self-organizing multilayered neural network, *Biological Cybernetics*, 20(3–4), 121–136. <https://doi.org/10.1007/BF00342633>
- Godard, C., Aodha, O. Mac, Firman, M., & Brostow, G., 2019, Digging Into Self-Supervised Monocular Depth Estimation, *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019-Octob(2), 3827–3837. <https://doi.org/10.1109/ICCV.2019.00393>
- Goodfellow, I., Bengio, Y., & Courville, A., 2016, *Deep Learning*, MIT Press.
- Han, C., Zhao, Q., Zhang, S., Chen, Y., Zhang, Z., & Yuan, J., 2022, YOLOPv2: Better, Faster, Stronger for Panoptic Driving Perception, *ArXiv: Cs.CV/2208.11434*. <http://arxiv.org/abs/2208.11434>
- Han, L., Lin, Y., Du, G., & Lian, S., 2019, DeepVIO: Self-supervised Deep Learning of Monocular Visual Inertial Odometry using 3D Geometric Constraints, *IEEE International Conference on Intelligent Robots and Systems*, 6906–6913. <https://doi.org/10.1109/IROS40897.2019.8968467>
- Han, L., Wu, L., Liang, F., Cao, H., Luo, D., Zhang, Z., & Hua, Z., 2022, A novel end-to-end model for steering behavior prediction of autonomous ego-vehicles using spatial and temporal attention mechanism, *Neurocomputing*, 490, 295–311. <https://doi.org/10.1016/j.neucom.2021.11.093>
- Hassan, I. U., 2021, *Steering Dataset*, Kaggle. <https://doi.org/10.34740/KAGGLE/DS/1285878>
- He, K., Zhang, X., Ren, S., & Sun, J., 2014, Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition, In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), *Computer Vision -- ECCV 2014* (Vol. 45, Number 4, pp. 346–361), Springer International Publishing. <https://doi.org/10.1023/B:KICA.0000038074.96200.69>
- Hecker, S., Dai, D., & Van Gool, L., 2018, End-to-end learning of driving models with surround-view cameras and route planners, *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11211 LNCS, 449–468. https://doi.org/10.1007/978-3-030-01234-2_27
- Hochreiter, S., & Schmidhuber, J., 1997, Long Short-Term Memory, *Neural Computation*, 9(8), 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
- Hou, Y., Ma, Z., Liu, C., & Loy, C. C., 2019, Learning to steer by mimicking features from heterogeneous auxiliary networks, *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, 2, 8433–8440. <https://doi.org/10.1609/aaai.v33i01.33018433>
- Huang, Z., Lv, C., Xing, Y., & Wu, J., 2021, Multi-Modal Sensor Fusion-Based Deep Neural Network for End-to-End Autonomous Driving With Scene Understanding, *IEEE Sensors Journal*, 21(10), 11781–11790. <https://doi.org/10.1109/JSEN.2020.3003121>

- Jahromi, B. S., Tulabandhula, T., & Cetin, S., 2019a, Real-time hybrid multi-sensor fusion framework for perception in autonomous vehicles, *Sensors (Switzerland)*, 19(20), 1–23. <https://doi.org/10.3390/s19204357>
- Jahromi, B. S., Tulabandhula, T., & Cetin, S., 2019b, Hybrid Sensor Fusion Framework for Perception in Autonomous Vehicles, *Machine Learning for Autonomous Driving Workshop at the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*.
- Jia Deng, Wei Dong, Socher, R., Li-Jia Li, Kai Li, & Li Fei-Fei, 2009, ImageNet: A large-scale hierarchical image database, *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 248–255. <https://doi.org/10.1109/CVPRW.2009.5206848>
- Jiang, H., Chang, L., Li, Q., & Chen, D., 2020, Deep Transfer Learning Enable End-to-End Steering Angles Prediction for Self-driving Car, *IEEE Intelligent Vehicles Symposium, Proceedings, Iv*, 405–412. <https://doi.org/10.1109/IV47402.2020.9304611>
- Johnson, F., & Dana, K., 2020, Feudal Steering: Hierarchical Learning for Steering Angle Prediction, *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020-June, 4316–4325. <https://doi.org/10.1109/CVPRW50498.2020.00509>
- Kim, J., & Canny, J., 2017, Interpretable Learning for Self-Driving Cars by Visualizing Causal Attention, *Proceedings of the IEEE International Conference on Computer Vision, 2017-Octob*, 2961–2969. <https://doi.org/10.1109/ICCV.2017.320>
- Kim, J., Choi, J., Kim, Y., Koh, J., Chung, C. C., & Choi, J. W., 2018, Robust Camera Lidar Sensor Fusion Via Deep Gated Information Fusion Network, *2018 IEEE Intelligent Vehicles Symposium (IV)*, Iv, 1620–1625. <https://doi.org/10.1109/IVS.2018.8500711>
- Kingma, D. P., & Ba, J. L., 2015, Adam: A Method for Stochastic Optimization, In Y. Bengio & Y. LeCun (Eds.), *3rd International Conference on Learning Representations, {ICLR} 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (pp. 1–15). <http://arxiv.org/abs/1412.6980>
- Klingner, M., Bär, A., Fingscheidt, T., Andreas, B., & Fingscheidt, T., 2020, Improved Noise and Attack Robustness for Semantic Segmentation by Using Multi-Task Training with Self-Supervised Depth Estimation, *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 1299–1309. <https://doi.org/10.1109/CVPRW50498.2020.00168>
- Kloutiniotis, A., Papandreou, A., Anagnostopoulos, C., Lalos, A., Kapsalas, P., Nguyen, D. V., & Moustakas, K., 2022, CarlaScenes: A synthetic dataset for odometry in autonomous driving, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2022-June, 4519–4527. <https://doi.org/10.1109/CVPRW56347.2022.00498>
- Krizhevsky, A., Sutskever, I., & Hinton, G. E., 2012, ImageNet Classification with Deep Convolutional Neural Networks, In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems* (Vol. 25), Curran Associates, Inc.
- LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D., 1989, Backpropagation applied to digit recognition, In *Neural computation* (Vol. 1, Number 4, pp. 541–551). <https://www.ics.uci.edu/~welling/teaching/273ASpring09/lecun-89e.pdf>
- LeCun, Y., Muller, U., Ben, J., Cosatto, E., & Flepp, B., 2005, Off-road obstacle avoidance through end-to-end learning, *Advances in Neural Information Processing Systems*, 739–746.

- Lee, S., Rameau, F., Im, S., & So, I., 2022, Self-Supervised Monocular Depth and Motion Learning in Dynamic Scenes : Semantic Prior to Rescue, *International Journal of Computer Vision*, 130(9), 2265–2285. <https://doi.org/10.1007/s11263-022-01641-5>
- Li, C., Wang, S., Zhuang, Y., & Yan, F., 2021, Deep Sensor Fusion Between 2D Laser Scanner and IMU for Mobile Robot Localization, *IEEE Sensors Journal*, 21(6), 8501–8509. <https://doi.org/10.1109/JSEN.2019.2910826>
- Li, X., Liu, Z., Luo, P., Loy, C. C., & Tang, X., 2017, Not all pixels are equal: Difficulty-aware semantic segmentation via deep layer cascade, *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017-Janua(Mc)*, 6459–6468. <https://doi.org/10.1109/CVPR.2017.684>
- Lin, M., Chen, Q., & Yan, S., 2014, Network in network, *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, 1–10.
- Long, Y., Morris, D., Liu, X., Castro, M., Chakravarty, P., & Narayanan, P., 2021, Radar-Camera Pixel Depth Association for Depth Completion, *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 1, 12502–12511. <https://doi.org/10.1109/cvpr46437.2021.01232>
- Maas, A. L., Hannun, A. Y., & Ng, A. Y., 2013, Rectifier nonlinearities improve neural network acoustic models, *In ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 28.
- Machiraju, H., & Balasubramanian, V. N., 2020, A Little Fog for a Large Turn, *Proceedings - 2020 IEEE Winter Conference on Applications of Computer Vision, WACV 2020*, 2891–2900. <https://doi.org/10.1109/WACV45572.2020.9093549>
- Martinez, A. H., Diaz, J. L., Daza, I. G., & Llorca, D. F., 2021, Data-driven vehicle speed detection from synthetic driving simulator images, *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, 2021-Septe*, 2617–2622. <https://doi.org/10.1109/ITSC48978.2021.9564888>
- Martins, D., Van Hecke, K., & De Croon, G., 2018, Fusion of stereo and still monocular depth estimates in a self-supervised learning context, *Proceedings - IEEE International Conference on Robotics and Automation*, 849–856. <https://doi.org/10.1109/ICRA.2018.8461116>
- Müller, M., Dosovitskiy, A., Ghanem, B., & Koltun, V., 2018, Driving Policy Transfer via Modularity and Abstraction, *ArXiv: Cs.RO/1804.09364*. <http://arxiv.org/abs/1804.09364>
- Naik, A., & Rangwala, H., 2018, Multi-task Learning, *SpringerBriefs in Computer Science*, 75, 75–88. https://doi.org/10.1007/978-3-030-01620-3_5
- Narayanan, A., Siravuru, A., & Dariush, B., 2020, Gated Recurrent Fusion to Learn Driving Behavior from Temporal Multimodal Data, *IEEE Robotics and Automation Letters*, 5(2), 1287–1294. <https://doi.org/10.1109/LRA.2020.2967738>
- Nguyen, A., Do, T., Tran, M., Nguyen, B. X., Duong, C., Phan, T., Tjiputra, E., & Tran, Q. D., 2021, Deep Federated Learning for Autonomous Driving, *ArXiv*. <http://arxiv.org/abs/2110.05754>
- OpenDRIVE, 2023, *OpenDRIVE* <https://www.asam.net/standards/detail/opendrive/> [Ziyaret Tarihi: 1 Şubat 2024]
- Person, M., Jensen, M., Smith, A. O., & Gutierrez, H., 2019, Multimodal Fusion Object Detection System for Autonomous Vehicles, *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, 141(7), 1–9. <https://doi.org/10.1115/1.4043222>
- Pomerleau, D. A., 1988, ALVINN: An Autonomous Land Vehicle in a Neural Network, In D. Touretzky (Ed.), *Advances in Neural Information Processing Systems* (Vol. 1, pp. 305–313), Morgan-Kaufmann.

- Qian, D., Ren, D., Meng, Y., Zhu, Y., Ding, S., Fu, S., Wang, Z., & Xia, H., 2018, End-to-End Learning Driver Policy using Moments Deep Neural Network, *2018 IEEE International Conference on Robotics and Biomimetics, ROBIO 2018*, 1533–1538. <https://doi.org/10.1109/ROBIO.2018.8664869>
- ROS, 2023, *Robot Operating System*. <https://www.ros.org> [Ziyaret Tarihi: 1 Şubat 2024]
- Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., & Fei-Fei, L., 2015, ImageNet Large Scale Visual Recognition Challenge, *International Journal of Computer Vision*, 115(3), 211–252. <https://doi.org/10.1007/s11263-015-0816-y>
- Sallab, A. El, Sobh, I., Elmadawy, K., Amin, L., Gamal, M., Gamal, M., Abdeltawab, O., & Abdelkarim, S., 2018, End-To-End Multi-Modal Sensors Fusion System For Urban Automated Driving, *NIPS 2018 Workshop MLITS Submission, CoRL*.
- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C., 2018, MobileNetV2: Inverted Residuals and Linear Bottlenecks, *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4510–4520. <https://doi.org/10.1109/CVPR.2018.00474>
- Sekkat, A. R., Dupuis, Y., Vasseur, P., & Honeine, P., 2020, The OmniScape Dataset, *Proceedings - IEEE International Conference on Robotics and Automation*, 1603–1608. <https://doi.org/10.1109/ICRA40945.2020.9197144>
- Shao, M.-E., Haq, M. A., Gao, D.-Q., Chondro, P., & Ruan, S.-J., 2022, Semantic Segmentation for Free Space and Lane Based on Grid-Based Interest Point Detection, *IEEE Transactions on Intelligent Transportation Systems*, 23(7), 8498–8512. <https://doi.org/10.1109/TITS.2021.3083526>
- Shelhamer, E., Long, J., & Darrell, T., 2017, Fully Convolutional Networks for Semantic Segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4), 640–651. <https://doi.org/10.1109/TPAMI.2016.2572683>
- Sirohi, K., Marvi, S., Büscher, D., & Burgard, W., 2023, Uncertainty-Aware Panoptic Segmentation, *IEEE Robotics and Automation Letters*, 8(5), 2629–2636. <https://doi.org/10.1109/LRA.2023.3256926>
- Sully Chen, 2023, *Sully Chen*. <https://github.com/SullyChen/driving-datasets> [Ziyaret Tarihi: 1 Şubat 2024]
- Sun, L., Bian, J. W., Zhan, H., Yin, W., Reid, I., & Shen, C., 2023, SC-DepthV3: Robust Self-Supervised Monocular Depth Estimation for Dynamic Scenes, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP, 1–12. <https://doi.org/10.1109/TPAMI.2023.3322549>
- Tai, L., Yun, P., Chen, Y., Liu, C., Ye, H., & Liu, M., 2019, Visual-based Autonomous Driving Deployment from a Stochastic and Uncertainty-aware Perspective, *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2622–2628. <https://doi.org/10.1109/IROS40897.2019.8968307>
- Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., & Le, Q. V., 2019, Mnasnet: Platform-aware neural architecture search for mobile, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2019-June*, 2815–2823. <https://doi.org/10.1109/CVPR.2019.00293>
- Tan, M., & Le, Q. V., 2019, EfficientNet: Rethinking model scaling for convolutional neural networks, *36th International Conference on Machine Learning, ICML 2019, 2019-June*, 10691–10700.
- Tan, M., & Le, Q. V., 2021, EfficientNetV2: Smaller Models and Faster Training, *ArXiv: Cs.CV/2104.00298v3*. <http://arxiv.org/abs/2104.00298>
- Teetor, R. R., 1948, *Speed Control Device for Resisting Operation of the Accelerator* (No. 2519859). <https://image-ppubs.uspto.gov/dirsearch->

- public/print/downloadPdf/2519859
- Udacity, 2017, *Udacity* <https://github.com/udacity/self-driving-car> [Ziyaret Tarihi: 1 Şubat 2024]
- Unreal Engine, 2023, *Unreal Engine* <https://www.unrealengine.com/en-US/> [Ziyaret Tarihi: 1 Şubat 2024]
- Wagstaff, B., & Kelly, J., 2021, Self-Supervised Scale Recovery for Monocular Depth and Egomotion Estimation, *IEEE International Conference on Intelligent Robots and Systems*, 2620–2627. <https://doi.org/10.1109/IROS51168.2021.9635938>
- Wan, P., Song, Z., & Lu, J., 2021, Incorporating Orientations into End-to-end Driving Model for Steering Control, *ArXiv: Cs.RO/2103.05846v1*, 1–19. <http://arxiv.org/abs/2103.05846>
- Wang, C., Liao, H. M., & Yeh, I., 2022, Designing Network Design Strategies Through Gradient Path Analysis, *ArXiv: Cs.CV/2211.04800*. <https://doi.org/10.48550/ARXIV.2211.04800>
- Wang, D., Wen, J., Wang, Y., Huang, X., & Pei, F., 2019, End-to-End Self-Driving Using Deep Neural Networks with Multi-auxiliary Tasks, *Automotive Innovation*, 2(2), 127–136. <https://doi.org/10.1007/s42154-019-00057-1>
- Wang, Q., Chen, L., Tian, B., Tian, W., Li, L., & Cao, D., 2019, End-to-End Autonomous Driving: An Angle Branched Network Approach, *IEEE Transactions on Vehicular Technology*, 68(12), 11599–11610. <https://doi.org/10.1109/TVT.2019.2921918>
- Wang, T., Dhiman, V., & Atanasov, N., 2023, Inverse reinforcement learning for autonomous navigation via differentiable semantic mapping and planning, *Autonomous Robots*, 47(6), 809–830. <https://doi.org/10.1007/s10514-023-10118-4>
- Wang, T., Luo, Y., Liu, J., Chen, R., & Li, K., 2022, End-to-End Self-Driving Approach Independent of Irrelevant Roadside Objects with Auto-Encoder, *IEEE Transactions on Intelligent Transportation Systems*, 23(1), 641–650. <https://doi.org/10.1109/TITS.2020.3018473>
- Wei, S., Chen, G., Chi, W., Wang, Z., & Sun, L., 2022, 3D Object Aided Self-Supervised Monocular Depth Estimation, *IEEE International Conference on Intelligent Robots and Systems*, 2022-Octob, 10635–10642. <https://doi.org/10.1109/IROS47612.2022.9981590>
- Wong, K., Gu, Y., & Kamijo, S., 2021, Mapping for Autonomous Driving: Opportunities and Challenges, *IEEE Intelligent Transportation Systems Magazine*, 13(1), 91–106. <https://doi.org/10.1109/MITS.2020.3014152>
- Wu, D., Liao, M.-W., Zhang, W.-T., Wang, X.-G., Bai, X., Cheng, W.-Q., Liu, W.-Y., & Zhang, M. L. W., 2022, YOLOP: You Only Look Once for Panoptic Driving Perception, *Machine Intelligence Research*, 19(6), 550–562. <https://doi.org/10.1007/s11633-022-1339-y>
- Wu, T., Luo, A., Huang, R., Cheng, H., & Zhao, Y., 2019, End-to-End Driving Model for Steering Control of Autonomous Vehicles with Future Spatiotemporal Features, *IEEE International Conference on Intelligent Robots and Systems*, 950–955. <https://doi.org/10.1109/IROS40897.2019.8968453>
- Xiao, Y., Codevilla, F., Gurram, A., Urfalioglu, O., & Lopez, A. M., 2022, Multimodal End-to-End Autonomous Driving, *IEEE Transactions on Intelligent Transportation Systems*, 23(1), 537–547. <https://doi.org/10.1109/TITS.2020.3013234>
- Xie, Z., Zhou, S., Zheng, M., & Pei, F., 2023, Research on self-supervised depth estimation algorithm of driving scene based on monocular vision, *Signal, Image and Video Processing*, 17(4), 991–999. <https://doi.org/10.1007/s11760-022-02303-2>
- Xu, H., Gao, Y., Yu, F., & Darrell, T., 2017, End-to-end learning of driving models from large-scale video datasets, *Proceedings - 30th IEEE Conference on Computer Vision*

- and *Pattern Recognition, CVPR 2017, 2017-Janua*, 3530–3538. <https://doi.org/10.1109/CVPR.2017.376>
- Yang, L., Liang, X., & Xing, E., 2018, *Unsupervised Real-to-Virtual Domain Unification for End-to-End Highway Driving, Figure 1*, 1–16. <http://arxiv.org/abs/1801.03458>
- Yang, Z., Zhang, Y., Yu, J., Cai, J., & Luo, J., 2018, End-to-end Multi-Modal Multi-Task Vehicle Control for Self-Driving Cars with Visual Perceptions, *Proceedings - International Conference on Pattern Recognition, 2018-Augus*, 2289–2294. <https://doi.org/10.1109/ICPR.2018.8546189>
- Yuan, W., Yang, M., Li, H., Wang, C., & Wang, B., 2018, End-to-end learning for high-precision lane keeping via multi-state model, *CAAI Transactions on Intelligence Technology*, 3(4), 185–190. <https://doi.org/10.1049/trit.2018.1025>
- Yuan, W., Yang, M., Li, H., Wang, C., & Wang, B., 2021, SteeringLoss: A Cost-Sensitive Loss Function for the End-to-End Steering Estimation, *IEEE Transactions on Intelligent Transportation Systems*, 22(2), 1104–1113. <https://doi.org/10.1109/TITS.2019.2963003>
- Zhang, H., Wu, C., Zhang, Z., Zhu, Y., Lin, H., Zhang, Z., Sun, Y., He, T., Mueller, J., Manmatha, R., Li, M., & Smola, A., 2022, ResNeSt: Split-Attention Networks, *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2022-June, 2735–2745. <https://doi.org/10.1109/CVPRW56347.2022.00309>
- Zhang, J., Li, W., Gou, H., Fang, L., & Yang, R., 2021, LEAD: LiDAR Extender for Autonomous Driving, *ArXiv:2102.07989v1 [Cs.CV]*. <http://arxiv.org/abs/2102.07989>
- Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J., 2017, Pyramid scene parsing network, *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017-Janua*, 6230–6239. <https://doi.org/10.1109/CVPR.2017.660>
- Zhou, Y. T., & Chellappa, R., 1988, *Computation of optical flow using a neural network*, 86, 71–78. <https://doi.org/10.1109/icnn.1988.23914>
- Zuo, Y., Cui, L., Peng, X., Xu, Y., Gao, S., Wang, X., & Kneip, L., 2021, Accurate depth estimation from a hybrid event-RGB stereo setup, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 6810–6817.