T.C.
KONYA TEKNİK ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

KAPALI ORTAMLARDA OTONOM İNSANSIZ HAVA SİSTEMLERİNİN GELİŞTİRİLMESİ

Muhammet Fatih Aslan

DOKTORA TEZİ

Elektrik Elektronik Mühendisliği Anabilim Dalı

Mayıs- 2022
KONYA

TEZ KABUL VE ONAYI

Muhammet Fatih Aslan tarafından hazırlanan "**Kapalı Ortamlarda Otonom İnsansız Hava Sistemlerinin Geliştirilmesi**" adlı tez çalışması **…/…/2022** 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 Prof. Dr. Mehmet ÇUNKAŞ	……..
Danışman Doç. Dr. Akif DURDU	……..
Yardımcı Danışman Doç. Dr. Kadir SABANCI	……..
Üye Prof. Dr. Ömer AYDOĞDU	……..
Üye Doç. Dr. Seyfettin Sinan GÜLTEKİN	……..
Üye Dr. Öğr. Üyesi Muhammed Fahri ÜNLERŞEN	……..

Yukarıdaki sonucu onaylarım.

Prof. Dr. Saadettin Erhan KESEN
Enstitü Müdürü

Bu tez çalışması TÜBİTAK tarafından 2211/A Yurt İçi Genel Doktora Burs Programı kapsamında desteklenmiştir.

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.

Muhammet Fatih ASLAN

Tarih:

ÖZET

DOKTORA TEZİ

KAPALI ORTAMLARDA OTONOM İNSANSIZ HAVA SİSTEMLERİNİN GELİŞTİRİLMESİ

Muhammet Fatih ASLAN

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

Danışman: Doç. Dr. Akif DURDU

2022, 165 Sayfa

Jüri
Prof. Dr. Mehmet ÇUNKAŞ
Doç. Dr. Akif DURDU
Prof. Dr. Ömer AYDOĞDU
Doç. Dr. Seyfettin Sinan GÜLTEKİN
Dr. Öğr. Üyesi Muhammed Fahri ÜNLERŞEN

İnsansı görevlerin robotlara yaptırılma ihtiyacı, tıpkı insanlar gibi, kendi kararlarını veren ve buna göre bir görev gerçekleştiren otonom mobil robot uygulamalarını ortaya çıkarmıştır. Bir otonom robot bulunduğu ortamın geometrik yapısını bilmeli, buna göre kendini konumlandırmalı ve son olarak bu bilgilere dayanarak görev noktasına doğru bir hareket yörüngesi oluşturmalıdır. İnsanlarla aynı ortamı paylaşan otonom robotlar geliştirmek için Küresel Konumlandırma Sistemi (Global Positoning System (GPS))'nin yetersiz olduğu iç ortamlarda, farklı sensörlerle mobil robotu konumlandırmaya yönelik Odometri ve Eşzamanlı Konumlandırma ve Haritalama (Simultaneous Localization and Mapping (SLAM)) çalışmaları mevcuttur. Son on yılda araştırmacılar, işlemci hızındaki gelişmeler nedeniyle, maliyet olarak düşük monoküler kameralar ile gerçekleştirilen Görsel Odometri (Visual SLAM (VO)) ve Görsel SLAM (Visual SLAM (VSLAM)) yöntemlerine odaklanmıştır. Ayrıca son zamanlarda, kameralara ek olarak, düşük maliyetli Atalet Ölçü Birimi (Inertial Measurement Unit (IMU)) sensörleri içeren VISLAM ve VIO çözümleri, konumlandırmaya sağladığı katkı nedeni ile sıklıkla tercih edilmeye başlamıştır. Şimdiye kadarki çözümler, genellikle geleneksel geometrik tabanlı çözümler içerirler. Bu klasik yöntemlerle gerçek karmaşık dünyanın iyi temsili çok zor olduğundan, genellikle güvenilir sonuçlar elde edilmez ve ayrıca elle ayarlanan özelliklere çok bağımlıdırlar. Bu nedenle günümüzde geleneksel çözümlerin yerini, farklı ortamlara uyarlanabilmesi ve uygulama kolaylığı sağlaması açısından Yapay Zekâ tabanlı çözümler almaktadır.

Bu tez çalışması yukarıda bahsedilen bilgiler ışığında, GPS erişimi olmayan iç ortamlarda otonom bir İnsansız Hava Araçları (İHA) geliştirilmesi için üç farklı uygulama önermektedir. İlk uygulama iç ortamda hareket eden bir İHA'nın konumunu tahmin etmek için derin öğrenme tabanlı hibrit bir mimari ile görsel ve IMU bilgilerine dayalı bir çalışma sunmaktadır. İkinci uygulama IMU bilgisini görüntüye dönüştüren ve İHA'nın konumu yanında açı bilgisini de başarılı bir şekilde tahmin eden yapay zekâ tabanlı farklı bir VIO uygulamasını farklı bir füzyon tekniğiyle gerçekleştirmektedir. Son uygulama bir üç boyutlu ortamda, konum bilgisi bilinen bir İHA için yeni bir yol planlama yöntemi önermektedir. Üstelik yeni bir yol planlama yönteminin yanında, önerilen yöntem için optimizasyon ve Yapay Zeka tabanlı bir uygulama geliştirilmiş, ve sonuçta gerçek zamanlı bir yol planlaması sağlanmıştır. Üç uygulama da iç ortamda otonom bir İHA geliştirilmesi için yeni yöntemler sunmaktadır. Tüm yöntemler önceki çalışmaların büyük bir kısmına üstünlük sağlayacak performans gösterirler. Ayrıca gerçekleştirilen uygulamalar gerçek zamanlı sistemlerde çalışabilecek niteliktedir.

Anahtar kelimeler: Derin Öğrenme, Otonom İHA, VIO, VISLAM, Yol Planlama

ABSTRACT

Ph.D. THESIS

DEVELOPMENT OF AUTONOMOUS UNMANNED AERIAL SYSTEMS IN INDOOR ENVIRONMENTS

Muhammet Fatih ASLAN

Konya Technical University
Institute of Graduate Studies
Department of Electric and Electronic Engineering

Advisor: Assoc. Prof. Dr. Akif DURDU

2022, 165 Pages

Jury
Prof. Dr. Mehmet ÇUNKAŞ
Assoc. Prof. Dr. Akif DURDU
Prof. Dr. Ömer AYDOĞDU
Assoc. Prof. Dr. Seyfettin Sinan GÜLTEKİN
Asst. Prof. Dr. Muhammed Fahri ÜNLERŞEN

The need for humanoid tasks to be performed by robots has led to autonomous mobile robot applications that make their own decisions and perform a task accordingly, just like humans. An autonomous robot must know the geometric structure of its environment, position itself accordingly, and finally, based on this information, it must create a movement trajectory towards the task point. In order to develop autonomous robots sharing the same environment with humans, there are Odometry and Simultaneous Localization and Mapping (SLAM) studies to localize the mobile robot with different sensors in indoor environments where the Global Positioning System (GPS) is insufficient. In the last decade, researchers have focused on Visual SLAM (VO) and Visual SLAM (VSLAM) methods performed with cost-effective monocular cameras due to improvements in processor speed. In addition to cameras, VISLAM and VIO solutions, which include low-cost Inertial Measurement Unit (IMU) sensors, have recently been preferred because of their contribution to localization. Solutions so far often include traditional geometric-based solutions. Because the good representation of the real complex world is very difficult with these classical methods, reliable results are often not obtained and they are also very dependent on manually adjusted parameters. Therefore, nowadays, traditional solutions are replaced by Artificial Intelligence-based solutions in terms of adaptability to different environments and ease of application.

In the light of the above-mentioned information, this thesis proposes three different applications for the development of an autonomous Unmanned Aerial Vehicle (UAV) in GPS-denied indoor environments. The first application offers a deep learning-based hybrid architecture and visual and IMU information-based work to predict the position of a UAV moving indoors. The second application implements a different artificial intelligence-based VIO application with a different fusion technique, which transforms the IMU information into an image and successfully estimates the angle information as well as the position of the UAV. The last application proposes a new path planning method for a UAV with known position information in a three-dimensional environment. Moreover, in addition to a new path planning method, an optimization and Artificial Intelligence-based application has been developed for the proposed method, resulting in real-time path planning. All three applications offer new methods for the development of an autonomous UAV in the indoor environment. All methods showed the performance to outperform most previous studies. In addition, the applications realized are capable of working in real-time systems.

Keywords: Deep Learning, Autonomous UAV, VIO, VISLAM, Path Planning

ÖNSÖZ

Öncelikle bugünlere gelmemizde emeği büyük olan bizi yetiştiren anne ve babama, bana mesleği ve kariyeri ile birlikte her konuda örnek olan abime,

Ufku ve yol göstermesi ile her daim benim için ilham kaynağı ve rol model olan sayın danışmanın Doç. Dr. Akif DURDU'ya ve beni üyesi olmakla gururlandırdığı RAC-LAB ailesine,

Tez yazma sürecimde bana desteğini esirgemeyen ve aynı zamanda tez jürisinde yer alan sayın hocam Doç. Dr. Kadir SABANCI'ya

Tez çalışmam süresince 2211/A Yurt İçi Genel Doktora Burs Programı (1649B031900779) kapsamında bana genel bursiyer olarak destek sağlayan Türkiye Bilimsel ve Teknolojik Araştırma Kurumu TÜBİTAK'a,

Evliliğimi güzelleştiren, beni hayata daha çok bağlayan, oğlum Yunus'a,

Ve bana her daim destek olan, başarımın arkasındaki en büyük etken, hayatımı süsleyen, hayatımın sonuna kadar seveceğim sevgili eşim Büşra'ya,

teşekkürlerimi sunmaktan haz duyarım.

Muhammet Fatih ASLAN
KONYA-2022

İÇİNDEKİLER

ŞEKİLLER TABLOSU

ÇİZELGELER TABLOSU

SİMGELER VE KISALTMALAR

Simgeler

t : Zaman Faktörü
k : Zaman Faktörü
u_t : Kontrol Girişi
x_t : Durum Vektörü
z_k : Ölçüm Vektörü
m : Harita Noktaları, Gradyan hareketli ortalaması
P : Olasılık Dağılımı
W_k : Beyaz Gürültü (Sistem Modeli)
V_k : Beyaz Gürültü (Ölçüm Modeli)
$Bel(x_t)$: Sonrasal Olasılık Yoğunluk Fonksiyonu (İnanç)
α : Normalizasyon Faktörü, Öğrenme Oranı, Lineer Hız
$P(z_t|x_t)$: Ölçüm Modeli (Olabilirlik fonksiyonu)
$P(x_t|u_t, x_{t-1})$: Hareket Modeli
m, n : 2B Görüntü Boyutları (satır, sütun)
s : Adım Değeri
p : Dolgu Matrisi
f : Filtre Boyutu
h_t : Gizli Durum
σ : Sigmoid Aktivasyon Foksiyonu
b : Ön Bilgi Değeri
W : Ağ Ağırlıkları
γ : Momentum değeri
$E(\theta_l)$: Kayıp Fonksiyonu
$\widehat{y_1}$: Tahmin Değerleri
θ : Güncellenecek Ağ Parametre Değeri
β_1 : Gradyan Bozulma Faktörü
β_2 : Kare Gradyan Bozulma Faktörü
l : Yineleme Sayısı
∇ : Gradyan
β_2 : Kare Gradyan Bozulma Faktörü
w : Açısal Hız
e : Hata
w : Açısal Hız
N : Normal Dağılım
w : Açısal Hız, Atalet Katsayısı
x_{rand} : Rastgele Örnek
δ : Ağaç Uzama Mesafesi
T : RRT* Ağaç Yapısı
V : Köşe Noktaları
E : Kenarlar
x_{rand} : Rastgele Örnek
c_1, c_2 : İvme Katsayıları

Kısaltmalar

İHA	: İnsansız Hava Araçları
UGV	: İnsansız Kara Aracı
GPS	: Küresel Konumlandırma Sistemi
SLAM	: Eşzamanlı Yerelleştirme ve Haritalama
VSLAM	: Görsel SLAM
VISLAM	: Görsel Atalet SLAM
VO	: Görsel Odometri
VIO	: Görsel Atalet Odometri
IMU	: Atalet Ölçüm Birimi
BA	: Demet ayarlama
CNN	: Konvolüsyonel Sinir Ağı
RNN	: Yinelemeli Sinir Ağları
SVM	: Destek Vektör Makinesi
YSA	: Yapay Sinir Ağları
LSTM	: Uzun Kısa Süreli Bellek
BiLSTM	: Çift Yönlü Uzun Kısa Süreli Bellek
OF	: Optik Akış
GPR	: Gauss Süreç Regresyonu
RRT	: Hızlı Keşfeden Rastgele Ağaç
RRT*	: Hızlı Keşfeden Rastgele Ağaç-Yıldız
GDRRT*	: Hedef Mesafeye Dayalı RRT*
PSO	: Parçacık Sürü Optimizasyonu
SIFT	: Ölçek Değişmez Unsur Dönüşümü
EKF	: Genişletilmiş Kalman Filtresi
GA	: Genetik algoritma
ROS	: Robot İşletim Sistemi
FC	: Tam Bağlı Katman
SGDM	: Momentumlu Stokastik Gradyan İnişi
RMSE	: Ortalama Karekök Hatası

1. GİRİŞ

Robot, insanlara yardım etmek ya da insan görevlerini üstlenmek için önceden programlanmış görevleri ya da otonom görevleri gerçekleştiren elektro-mekanik bir cihazdır. En basit bir yapıdaki robot, hareket etmek için mekanik sistem, kontrol edilmek için elektronik bileşenler ve belirli bir görev gerçekleştirmek için bir yazılım/programlama içerir. Başlangıçta sadece montaj hatlarında belirli bir endüstriyel görevi geliştirmek için geliştirilen robot teknolojisi, günümüze kadar çok sayıda ilerleme kaydederek insanlarla beraber aynı ortamı paylaşmaya başladılar. Dolayısıyla modern robotlar genellikle mobil yapıdadır ve sürekli insanlarla etkileşim halindedir. Bu kapsamda, bugün, bilinmeyen bölgelerin keşfinde, arama-kurtarma çalışmalarında, tarımsal uygulamalarda, otonom araçların geliştirilmesinde, kargo taşımada, sosyal alanlarda (servis robotları olarak), vb. birçok görev için mobil robotlar aktif olarak kullanılmaktadır. Görüldüğü gibi, artık mobil robotlar hayatın her yerinde insanlarla birlikte yer almaya başlamıştır. Dolayısıyla dünyada hızla büyüyen teknolojilerinden biri de mobil robotlardır (Rubio ve ark., 2019; Niloy ve ark., 2021).

Mobil robotlar arasında son zamanlarda popülerliği artan İnsansız Hava Araçları (İHA) araştırmaları, günümüzde giderek daha popüler hale gelmiş ve dolayısıyla İHA ile ilgili yapılan çalışmalar yaygınlaşmıştır. Özellikle yüksek hareket kabiliyeti ve esnekliği nedeniyle sivil uygulamalar için giderek daha fazla İHA istihdam edilmektedir. İHA'lar arama-kurtarma (Alotaibi ve ark., 2019), savunma sanayi (Roberge ve ark., 2018), yangın söndürme (Spurny ve ark., 2021), kargo hizmetleri (Faust ve ark., 2017), uzaktan algılama (Osco ve ark., 2021), haritalama (Rossi ve ark., 2018), hassas tarım uygulamaları (Ye ve ark., 2020), vb. gibi çok geniş bir uygulama alanı yakalamıştır (Liu ve ark., 2020).

Günümüzün modern mobil robotik görevleri, robotların insanlardan bağımsız olarak görevler gerçekleştirilmesini zorunlu kılmıştır. Onlarca yıldır devam eden çaba, insanlar için zor, tehlikeli ve tekrarlı görevlerin robotlarla hızlı ve güvenilir bir şekilde yapılabilmesini sağlamak içindir. İnsanlardan tamamen bağımsız, yani otonom mobil robotlar geliştirmek için robotların güvenli bir şekilde navigasyonu sağlanmalıdır. Navigasyon için de temel şartlar, robotun çevresini tanıması ve bulunduğu çevreye göre kendini konumlandırmasıdır (Gul ve ark., 2019). Bu iki şart verimli bir şekilde gerçekleştirilemediği müddetçe, başarılı bir navigasyon sağlanamaz. Bu iki şart yerine getirildiğinde ise, mobil robot ile otonom bir görev gerçekleştirmek için robotun kendi hareketini planlaması gerekir (Rubio ve ark., 2019; Alatise ve Hancke, 2020; Aslan ve ark., 2021a).

1.1. Motivasyon

Bir mobil robotun bilinmeyen bir ortamda otonom hareketi için üç ana sorunun çözülmesi gerekir; konumlandırma, haritalama ve yol planlaması. Güçlü bir yol planlaması, başarılı bir konumlandırma ve ortam bilgisine bağlıdır.

Mobil robotlarla dış ortamlarda (outdoor) otonom görevler gerçekleştirmek ya da konumlandırmayı başarılı bir şekilde sağlamak için, Küresel Konumlandırma Sistemi (Global Positioning System (GPS)) sıklıkla kullanılır. Özellikle tarım alanında ve askeri alandaki mobil robot uygulamaları için dış ortamdaki konumlandırma problemi, GPS'in erişilebilir olması nedeniyle büyük ölçüde kolaylaşır. Ancak dış ortamlar arazi ve engebeli yollar içerdiğinden bu tür uygulamalar için mobil robotların mekanik tasarımı güçlü olmalıdır. Bunun yerine erişilmesi zor olan yerlere ulaşması, sağladığı hareket esnekliği, zengin bilgi içeren görüntüleri ve hızlı kontrolü ile İHA'lar, gözetleme ve keşif amacıyla gittikçe popülerlik kazanmaktadır. Özellikle tarım ve savunma alanında İHA'lar günümüzde büyük rol oynamaktadır. Daha önce yaptığımız bir araştırma çalışmasında (Aslan ve ark., 2022), dış ortamlardaki tarımsal İHA uygulamalarının çok popüler olduğu vurgulanmıştır.

Dış ortamlar konumlandırma açısından avantajlı olmasına rağmen, insan etkileşimli görevler için iç ortamlarda otonom çözümlere gereksinim daha fazladır. Bu nedenle ofislerde, evlerde, hava alanlarında, müzelerde, marketlerde, vb. birçok iç ortamlarda da mobil robotlar otonom görevleri gerçekleştirebilme kabiliyetine sahip olmalıdır. Ancak, GPS'in yeraltı veya iç ortamlarda mevcut veya güvenilir olmaması, otonom mobil robot görevleri için GPS'in kullanılamamasına neden olur. Bu durum, iç ortamlarda otonom mobil robotlar geliştirmek için araştırmacıların alternatif çözümler üretmesini sağlamıştır. Bu gibi durumlarda, mevcut konum bilgisini elde etmek için otonom robot üzerine ilave sensörler monte edilir (Gurturk ve ark., 2021).

Mobil robotlar için, konumlandırma ve ortam bilgisinin elde edilmesini içeren haritalandırma problemlerinin eşzamanlı çözümü için Eşzamanlı Yerelleştirme ve Haritalama (Simultaneous Localizing And Mapping (SLAM)) teknikleri uzun süredir araştırma alanıdır. Bunun yanında odometri olarak adlandırılan alan, ortama göre mobil robotu konumlandırır, ortamın geometrik haritasını çıkarmaz. SLAM ve odometri otonom mobil robotik için aktif bir araştırma alanıdır ve başarı oranlarını artırmak için sürekli olarak yeni algoritmalar geliştirilmektedir. Mobil bir robot için iç ortamlarda hassas bir konumlandırma, farklı nitelikteki sensörlerin birlikte verimli kullanımını gerektirir. Sensörlere ait belirsizliklerin, hataların, kameralara ait kalibrasyon ayarlarının

güçlü bir şekilde modellenmesi ve ayarlanması gerekmektedir. Gerçek zamanlı olarak konum belirleme için her bir ölçüm ve algılama değerlendirilmelidir, bu da algoritmanın bilgileri hızlı yorumlamasını gerektirir. Şimdiye kadarki geleneksel geometrik tabanlı SLAM ve odometri çalışmalarında, iç ortamlarda gerçek zamanlı çözümler için çok farklı yöntemler geliştirildi. Kolay çözümler için genellikle daha maliyetli sensörler gerekliydi. Minimum maliyet sağlayan kamera tabanlı çözümler (Görsel SLAM (Visual SLAM (VSLAM)) ya da Görsel Odometri (VO)) için de üç boyutlu (3B) gerçek dünyayı yorumlamak kolay değildi. Özellikle ölçüm belirsizliği, kamera karelerinden 3B bir dünyayı sanal olarak temsil etmeyi zorlaştırıyordu. Bunun için ucuz maliyetli ve erişilmesi mümkün olan Atalet Ölçüm Birimi (Inertial Measurement Unit (IMU)), konumlandırma açısından başarıyı artırarak, VO ve VSLAM çalışmalarına katkı sağladı. IMU sensörü ve kamera ile gerçekleştirilen çözümler ucuz olması ve daha başarılı tahminler üretmesi ile son yıllarda ön plana çıktı (Servières ve ark., 2021). Ancak gerçek dünyada, hatalı sensör ölçümleri, sistemin tam olarak modellenememesi, karmaşık ortamlar, çevresel dinamikler, vb. unsurlar bu tür el yapımı geometrik yöntemlerin doğruluğunu ve güvenilirliğini olumsuz etkiler (Chen ve ark., 2020). Bu nedenle son zamanlarda görüntü işleme ve bilgisayarlı görü alanında çığır açan derin öğrenme tabanlı çözümler geliştirilmeye başladı. Henüz yeni ve az keşfedilmiş bir alan olmasına rağmen, derin öğrenme tabanlı çözümler kolay uygulanması, genelleştirme yeteneğinin yüksek olması, kalibrasyona gerek duymaması, gerçek zamanlı çözümler, vb. avantajlar sağlaması nedeniyle iç ortamlarda otonom robot uygulamaları için sık tercih edilmeye başlandı (Wang ve ark., 2020c). Ancak doğruluk ve genelleştirilebilirlik açısından derin öğrenme tabanlı çözümlerin daha da geliştirilmesi gereklidir.

Başarılı bir konumlandırma sağlandıktan sonra verilen görevi otonom gerçekleştirmek için, robotun hedef konuma nasıl bir yörünge ile ilerleyeceğini belirleyen yol planlama şarttır. Yol planlama problemleri de konumlandırma gibi yıllardır devam eden araştırma alanıdır. Var olan uygulamaların çoğu genellikle iki boyutlu (2B) ortamlar için kurgulanmıştır. Ancak otonom kabiliyetin artık İHA'larla gerçekleştirilme ihtiyacı 3B yol planlaması için bir akım başlatmıştır. Bu kapsamda İHA ile yapılan yol planlama çalışmaları literatürde yer almaya başlamıştır. Ayrıca İHA'ların mevcut batarya kapasitesi problemi, 3B ortamlarda yol planlamayı İHA'lar için zorunlu kılmıştır. Görev noktasına en kısa yoldan ulaşan bir İHA, minimum enerji sarf eder. Bu da İHA'nın görev kabiliyetini önemli ölçüde artırır.

Geçmişten bugüne kadar otonom görevler için, farklı yol planlama algoritmaları önerilmiştir. Önerilen algoritmalar birbirlerine göre farklı açılardan üstünlük gösterir. Kimi algoritma küçük ortamlar için daha etkili iken kimi algoritma büyük ortamlarda daha iyi performans gösterir. Bu tez çalışması İHA tabanlı uygulamalar geliştirdiği için, üç boyutlu bir ortamda yol planlaması sağlamalıdır. Boyutun artması, beraberinde yol planlama probleminde zorluk meydana getirmektedir. Özellikle örnekleme tabanlı yaklaşımlar büyük ortamlar için daha çok tercih edilirken, bu yöntemler genellikle hedefe yavaş yakınsarlar. Ayrıca bulunan yol genellikle fazladan düğümler ve kenarlar içerir (Nasir ve ark., 2013). Bu nedenle mevcut yol planlama çalışmaları, 3B ya da büyük ortamlarda hızlı ve optimum yol bulma açısından yetersizdir.

1.2. Tezin Amacı

Otonom mobil robot görevlerinin gerçekleştirildiği ortam, robot otonomisi performansını önemli ölçüde etkiler. Bu nedenle bu tez çalışması, daha elzem olan ama daha sınırlı çözümlere sahip iç ortamlar için otonom mobil robot uygulamalarına yönelik çözümler sunmaktadır. İç ortamlardaki çoğu SLAM ve odometri uygulamalarında kullanılan tekerlekli robottan farklı olarak, bu tez çalışmasında İHA tercih edilmiştir. Motivasyon bölümünde anlatılan bilgiler, eksiklikler ve gereksinimler temel alınarak iç ortamlarda minimum maliyetli otonom İHA geliştirmek için yeni yöntemler geliştirilmiştir. Bu kapsamda literatürde bulunan çeşitli yöntemler araştırılmış ve daha sonra literatüre katkı sağlayacak yeni farklı yöntemler önerilmiştir.

Yapılan önceki çalışmalar analiz edildiğinde, şimdiye kadarki çalışmaların otonom İHA uygulamaları için umut verici olduğu rahatlıkla görülebilir. Ancak genel olarak ortak çaba, gerçek zamanlı olarak yüksek doğruluk ve optimal çözümler elde etmektir. SLAM ve odometri problemleri için konumlandırma doğruluğu ne kadar yüksek ise, yol planlama uygulamalarında da o denli yüksek başarı sağlanır. Ancak bu yüksek doğruluğa erişilme gereksinimi, sistemin hızını aksatacak yöntemlerle giderilmemelidir. Dolayısıyla doğruluk ve işlem hızı arasındaki denge korunarak, İHA konumlandırması gerçek zamanlı sağlanmalıdır. Sadece hassas konumlandırma ihtiyacının karşılanması, yol planlama probleminin verimliliği açısından yeterli değildir. Ayrıca yol planlaması sağlayacak algoritma büyük ve engellerle dolu bir ortamda hedef noktaya ulaşan yolu hızlıca ve engellere çarpmadan planlayabilmelidir. Bu bilgiler göz önüne alınarak, genel olarak özetlemek gerekirse, bu tezin amacı iç ortamlarda İHA'larla otonom görevler

gerçekleştirmek için gerekli olan konumlandırma ve yol planlama problemlerini verimli bir şekilde gerçek zamanlı çözebilecek yöntemler geliştirmektir.

1.3. Tezin Önemi

İlk ortaya çıktığı 1988 yılından itibaren, yeni yöntemler ve araştırmalar ile birlikte SLAM probleminin verimi ve uygulanabilirliği geliştirilmiştir. Bu kapsamda farklı sensör kullanımları ve farklı yöntem uygulamaları literatürde önerilmiştir. SLAM uygulamalarında kullanılan dış duyarlı (exteroceptive) sensörler lazer, sonar, kamera, GPS, vb. iken, iç duyarlı (proprioceptive) sensörler IMU, enkoder, odometri olarak örneklendirilebilir. Aslında bu sensörler, elde edilmek istenen konum ve harita değerlerini doğrudan ölçebilen sensörlerdir. Ancak bütün sensör verileri gürültülü ölçüm yapar. Aynı şekilde kameralar tarafından elde edilen kareler de gürültü içerir. Bu durum belirsizliği artırır ve ölçülen değerlerin doğrudan kullanımını imkânsız hale getirir. SLAM ve odometri uygulamalarında, bunu önemli bir problem haline getiren durum, sensör ölçümlerinin sürekli, ardışık olarak yapılması gereksinimidir. Bu nedenle, gürültü değeri çok küçük olsa bile, küçük olan gürültü değerleri zamanla büyüyerek belirsizliği artırır. Bu durumu önlemek için ölçülen değerlerin düzeltilmesi ya da iyileştirilmesi gerekir. Bu noktada kullanılacak olan yöntem önemlidir. Bunlar Bayes tabanlı filtre yöntemleri olabileceği gibi, optimizasyon tabanlı Demet Ayarlama (Bundle Adjustment (BA)) metotları da olabilir. Her iki yönteme ait çok sayıda çalışma şimdiye kadar önerilmiştir. Bayes tabanlı yöntemlerin hızlı tepki vermesi ancak doğruluğunun düşük olması, aksine, BA yöntemlerinin doğruluğunun yüksek olması ancak hızının düşük olması söz konusudur. Bu anlamda her iki yöntem birbirine göre üstünlük içerir (Chiu ve Sastry, 2021; Munguia ve ark., 2022). Bu iki yöntem dışında derin öğrenme tabanlı odometri ve SLAM yöntemleri şu an oldukça yeni bir alandır ve ilgi görmektedir. Yapılan çalışmalar incelendiğinde hem hız hem de doğruluk açısından oldukça performansı iyidir, ancak henüz geliştirme ve yeni yöntemlere açık bir alandır. Bu nedenle güncel çalışmalar genellikle farklı derin öğrenme tabanlı yöntemlerle VSLAM ve VO başarı oranlarını artırmaya yöneliktir. Bu bağlamda derin öğrenme tabanlı çözümlerin VSLAM ve VO için uygulanması güncel ve önemli bir konudur.

Zaman kavramı SLAM ve odometri için önemlidir, çünkü robot gerçek zamanlı olarak karar vermek zorundadır. Tabi kullanılacak olan yöntemin hızını etkileyen faktörlerden en önemlisi, kullanılan algılayıcı türüdür. Artık görsel algılayıcı çözümleri olan VO ve VSLAM üzerine çalışmalar yoğunlaşmıştır. Amaç GPS, lazer, ultrasonik,

LIDAR, vb. sensörler yerine maliyeti düşük ve daha zengin bilgi içeren kamera kullanmaktır. Ancak bu da beraberinde zaman sorununa yol açar. Neyse ki, günümüzdeki bilgisayarların gelişmesi, ucuzlaması ve işlemci performanslarının artması, VSLAM ve VO uygulamalarını gerçek zamanlı olarak uygulanabilir hale getirmiştir. Görsel bilgi daha zengin içeriğe sahip olduğundan elde edilen veriler üzerinde derin öğrenme, makine öğrenmesi, vb. konular daha pratik bir şekilde uygulanarak daha faydalı çıktılar sağlanabilir. İç mekân uygulamaları için GPS verimli çalışmayacaktır. Bu durumda sadece görsel bilgi kullanılabilir, ancak bu da ölçek belirsizliği durumu sebebiyle performansı sınırlayacaktır. Bu tür durumlarda IMU ve kamera bilgilerinin füzyonu ucuz ve performanslı bir çözümdür. Bu anlamda erişilmesi kolay ve maliyeti düşük kamera ve IMU sensörleriyle gerçekleştirilen VIO/VISLAM çözümleri oldukça önemlidir.

Günümüzde kullanım alanının artması ve hareket esnekliği nedeniyle oldukça tercih edilen İHA'lar ile otonom görevlerin gerçekleştirilmesi güncel ve aktif bir alandır. Askeri, savunma ve sivil uygulamalarda otonom İHA'ların gelecekte kritik görevler üstleneceği öngörülmektedir. Ancak bu görevlerin gerçekleştirilmesi sırasında sınırlı enerji kapasitesine sahip İHA, hassas konumlandırmaya ek olarak, görev noktaları arasında güçlü bir navigasyon sağlayabilmelidir. Bu nedenle otonom İHA alanına yönelik çalışmalar günümüzün önemli ve güncel teknolojisidir.

Yukarıda bahsedildiği üzere İHA'ları otonomize etmek için gerekli olan V(I)O, V(I)SLAM ve yol planlama gibi problemlerin verimli bir şekilde çözülmesi çok önemlidir. Mevcut çözümler her ne kadar yüksek başarılar sağlasa da, gerçek dünya çözümlerinin genelleştirilmesi ve gerçek zamanlı çözümlerin uygulanması konusunda eksiklikler mevcuttur. Bu tez çalışması, yaptığı uygulamalarla literatüre bu açıdan katkılar sağlamaktadır. Şüphesiz İHA'larla otonom olarak gerçekleştirilecek görevler, gelecekte insan hayatını kolaylaştıracak çok çeşitli teknolojilerin ortaya çıkmasını sağlayacaktır.

1.4. Tezin Kapsamı

Bu tez çalışması, iç ortamlarda otonom İHA görevlerinin geliştirilmesi için gerekli olan konumlandırma ve yol planlama için çözüm önerileri sunmaktadır. Bu kapsamda üç farklı uygulama geliştirilmiştir. İlk iki uygulama, iç ortamda hareket eden, bu hareket esnasında kamera karelerini ve IMU bilgileri toplayan bir İHA veri seti üzerinde denenmiştir. Daha sonra, test amacıyla simülasyon ortamında oluşturduğumuz farklı bir İHA veri seti üzerinde önerilen yöntemler test edilmiştir. Üçüncü uygulamada, 3B

ortamda İHA için yol planlamasına yönelik yeni bir yöntem geliştirilir. Daha sonra bu yeni yöntem derin öğrenme tabanlı uygulanarak hızlı bir yol planlaması sağlanmıştır.

İlk uygulamada, İHA verileri derin öğrenme tabanlı önerilen bir hibrit derin mimari üzerinde kullanılarak İHA'nın gerçek konum değerleri tahmin edilir. Genel sistem üç aşamalı bir öğrenme yaklaşımına sahiptir. İlk adımda, kamera ile elde edilen görüntülerden Konvolüsyonel Sinir Ağı (Convolutional Neural Network (CNN)) aracılığıyla uzamsal özellikler üretilir. Daha sonra, Yinelemeli Sinir Ağları (Recurrent Neural Networks (RNN) tabanlı Çift Yönlü Uzun Kısa Süreli Bellek (Bidirectional Long Short-Term Memory (BiLSTM)) kullanılarak IMU verilerinden zamansal özellikler çıkarılır. Son olarak füzyon adımında uzamsal ve zamansal özellikler birleştirilerek, BiLSTM katmanı ile İHA konumunu tahmin edilir.

İkinci uygulamada İHA'nın konum ve açı değerleri (yani pozunu) derin öğrenme ve makine öğrenmesi tabanlı yaklaşımlar aracılığıyla tahmin edilmiş ve şimdiye kadarki çalışmalardan farklı bir şekilde görsel ve atalet bilgileri birleştirilmiştir. Anlamlı bir birleşim için her iki kare arasındaki IMU verileri, gürültü kaldırma işleminden sonra Savitzky-Golay tekniği ile normalize edilmiş ve son olarak sayısal değerden görüntüye çevrilmiştir. Bu sayede IMU özellikleri görüntü verisi olarak kaydedilmiştir. Bu özellikleri iki kare arasındaki hareket değişikliği ile güçlendirmek için, Optik Akış (Optical Flow (OF)) yöntemi ile hareket değişimini gösteren kareler elde edilmiştir. Bu kareler atalet görüntüleri ve ham kareler ile birleştirilmiştir. Atalet ve kamera verisinden, CNN ile özellikler çıkarıldıktan sonra, bu özellikler bir araya getirilerek Gauss süreç Regresyonu (Gaussian Process Regression (GPR)) aracılığıyla İHA pozları tahmin edilmiştir. Bu tahmin değerlerine uygulanan yumuşatma işlemi sayesinde daha kararlı bir konum tahmini sağlanmıştır.

Üçüncü uygulama, 3B büyük bir iç ortam oluşturarak İHA için başlangıç ve hedef noktaları arasında, çarpışmalardan kaçınmak ve en optimum yolu keşfetmek için Rapidly Random-Exploring Tree Star (RRT*) algoritmasından faydalanır. Amaç RRT*'ın hedefe yavaş yakınsama problemini çözmek ve gerçek zamanlı uygulamalar için uyumlu yol planlaması sağlamaktır. Çalışma uygulama olarak üç aşamada ele alınabilir. İlk aşamada, RRT* algoritmasının hedefe daha hızlı ve doğrudan yaklaşmasını sağlayan Hedef Mesafesine dayalı RRT* (Goal Distance based RRT* (GDRRT*)) yaklaşımı önerilir. İkinci aşamada, başlangıç, bitiş ve engel koordinatları sınırlamalarına dayanılarak GDRRT* tarafından oluşturulan yolun Parçacık Sürü Optimizasyonu (Particle Swarm Optimization (PSO)) ile optimizasyonu ele alınır. Son aşama ise, daha hızlı bir yol

planlama için, PSO ile optimize edilen yolu BiLSTM katmanı kullanarak tahmin etmeye yöneliktir.

1.5. Tezin Organizasyonu

Bu tez çalışması, ana başlık olarak beş bölüme ayrılır. İlk bölüm bu tez çalışmasının motivasyonunu, amacını, önemini, kapsamını ve organizasyonunu alır. Dolayısıyla ilk bölüm iç ortamlarda otonom İHA uygulamalarının gerekliliğini kapsamlı bir şekilde tanıtır. İkinci bölüm, iç ortamlarda otonom İHA uygulamaları için gerekli olan VSLAM, VO, VIO, VISLAM, yol planlama konuları ile ilgili geçmişte yapılan çalışmalara değinir. Üçüncü bölüm materyal ve yöntem başlığı altında bu tez çalışmasının yapılabilmesi için gerekli olan teorik bilgileri geniş bir şekilde ele alır. Dördüncü bölüm kapalı ortamlarda otonom İHA uygulamalarının geliştirilmesine yönelik geliştirilen uygulamaları ve bu uygulamaların sonuçlarını anlatır. Beşinci ve son bölüm tüm geliştirilen uygulamaları değerlendirir, tartışır ve gelecekte yapılacak çalışmalar hakkında bilgi verir.

2. LİTERATÜR TARAMASI

Bu tez çalışması süresince otonom İHA sistemlerinin geliştirilmesine yönelik yöntemleri ya da uygulamaları içeren çok sayıda çalışma incelenmiştir. Bu çalışmalar SLAM ve Odometri çalışmaları ve yol planlama çalışmaları olarak gruplandırılmıştır.

2.1. SLAM ve Odometri Çalışmaları

SLAM ve Odometri çözümlerinin otonom mobil robotik uygulamalar için önemli olduğu anlaşıldıktan sonra, hesaplama problemlerine ilişkin çok sayıda çalışma yapılmıştır. Smith ve Cheeseman (1986) ve Durrant-Whyte (1988) tarafından yapılan çalışmalar, işaretçiler arasındaki ilişkileri tanımlamak ve geometrik belirsizliği gidermek için istatistiksel bir temel oluşturmuştur. Bu çalışmaların en önemli katkısı, haritadaki farklı işaretçilerin konumunun tahminleri arasında yüksek derecede bir ilişkinin (korelasyonun) olması gerektiğini ve bu korelasyonların ardışık gözlemlerle artacağını belirtmesidir. Smith ve ark. (1990) tarafından yapılan çalışma ile de çevresindeki işaretçileri gözlemleyerek bilinmeyen bir ortamda ilerleyen robot için, tahmin edilen robot konumundaki ortak hatadan dolayı işaretçilerin tahmini arasında mutlaka ilişki olduğu vurgulanmıştır. Dolayısıyla SLAM uygulamalarında sonraki her bir işaretçi gözleminin güncellenebilmesi için araç ve işaretçi konumlarından oluşan bir pozisyon bilgisi gereklidir. Ancak bu durumda da istatistiksel tahmin için işaretçi sayısının karesine bağlı bir durum vektörü kullanılmalıdır. Ayrıca bu çalışmalarda tahmin edilen harita hatalarının birleşmeyeceği ve robotun rastgele bir yürüyüş davranışı sergileyeceği kabul edilmiştir. Bu nedenle, haritalama probleminin hesap karmaşıklığı ve haritanın yakınsama davranışının bilinmemesi gibi sebeplerden ötürü SLAM üzerine teorik çalışma geçici olarak durmuş, çalışmalar genellikle haritalama veya yerelleştirmeye ayrı problemler olarak odaklanmıştır. Daha sonra haritalama ve lokalizasyon probleminin aslında yakınsak olduğu ve işaretçiler arasındaki korelasyonların büyüdükçe çözümün daha başarılı olduğu kabul edilmiştir. Bunun üzerine konumlandırma probleminin çözümüne ilişkin çok sayıda çalışma yapılmıştır (Bailey ve Durrant-Whyte, 2006; Durrant-Whyte ve Bailey, 2006).

Mobil robotlar kullanılarak gerçekleştirilen otonom uygulamalar, iç mekân haritalama, alan şartları veya güvenlik nedenlerinden dolayı insan erişiminin kısıtlanabileceği yerlerde oldukça gereklidir. Mobil robotu konumlandırmaya yönelik SLAM ve odometri, iç – dış ortamlarda, deniz altında (Trabes ve Jordan, 2017), uzayda (Chen ve ark., 2017) ve yer altı (Ren ve ark., 2019) ile ilgili çalışmalarda sıklıkla tercih

edilir. Tekerlekli robot uygulamaları dışında, düşük maliyetlerine ve çok yönlülüklerine bağlı olarak, haritalamaya ek olarak son zamanlarda trafik izleme, kentsel planlama, sivil güvenlik uygulamaları, ormancılık ve tarım gibi çeşitli endüstriler ve araştırma alanları arasında İHA kullanımına olan ilgi artmıştır. İHA ile ilgili çok sayıda uygulama mevcut olmasına rağmen iç mekân navigasyonu İHA için hala önemli bir zorluktur. Özellikle GPS sinyalinin çalışmadığı ortamlarda (yıkılmış veya yarı çökmüş binalar gibi) gözetim, afet yardımı veya kurtarma gibi iç mekân uygulamalarına yönelik çalışmalar artmaktadır (López ve ark., 2017; Dowling ve ark., 2018).

2.1.1. Görsel SLAM (VSLAM)/ Görsel Odometri (VO) Çalışmaları

Bu bölümde kamera içeren VSLAM ve VO çalışmalarının bir kısmı hakkında bilgi verilmiştir.

VSLAM için en önemli atılımlarından birini gerçekleştiren Klein ve Murray (2007), ORB-SLAM'in esin kaynağı olan optimizasyon tabanlı Paralel Takip ve Haritalama (Parallel Tracking and Mapping (PTAM)) algoritmasını önermişlerdir. İzleme ve haritalamayı paralel iş parçacıklarına bölme fikrini ortaya koyan ilk çalışmadır. PTAM, tipik bir ana kare tabanlı monoküler VSLAM çalışmasıdır ve doğrusal olmayan BA optimizasyon kullanır. Eşleme modülünde, anahtar kareler seyrek olarak seçilir ve bu anahtar kareler tarafından gözlemlenen harita noktaları haritalama için kullanılır. İzleme modülü, ön uçta kamera izleme görevlerini yerine getirir ve mevcut karenin hareketini gerçek zamanlı hesaplamalar için hızlı bir şekilde hesaplayabilir. Buna rağmen, PTAM büyük döngüleri algılayabilme yeteneğinden yoksundur.

Dolaylı yöntemlerden farklı olarak özellik çıkarma gereksinimi duymadan daha güçlü harita çıktıları üretmek amacıyla Engel ve ark. (2014), doğrudan (direct) SLAM yöntemleri arasında son derece popüler olan monoküler Large-Scale Direct SLAM (LSD-SLAM) algoritmasını tanıtmışlardır. Doğrudan SLAM algoritmaları görüntüdeki anahtar noktaları aramaz, bunun yerine konumu ve haritayı tahmin etmek için görüntü yoğunluklarını kullanır. Yani özellik tabanlı yöntemlere (örn. PTAM, ORB-SLAM, vs.) göre daha sağlamdırlar ve ayrıntılıdırlar, ancak bu da hesaplama maliyetinin fazla olmasına sebep olur. Çevrenin haritası, kamera görüntüsü içeren belirli anahtar karelere, bir ters derinlik haritası ve ters derinlik haritasının varyansına dayanarak oluşturulur. Önerilen algoritmayı uygulamak için, LSD-SLAM izleme, derinlik haritası tahmini ve harita optimizasyonu olmak üzere üç bölüme ayrılmıştır. İzleme aşamasında yeni görüntü alınır ve fotometrik hatayı minimize etmek suretiyle mevcut anahtar kare pozuna göre

mevcut kamera pozu tahmin edilir. Derinlik haritası tahmininde önce görüntünün ana kare olup olmadığı belirlenir. Yeni alınan kare, ana kare değilse mevcut ana kareyi iyileştirmek için kullanılır. Yeni kare ana kare ise bu ana karenin derinlik haritası başlatılır Harita optimizasyonu kısmında ise, ortalama bir ters derinliğe sahip farklı ölçeklerdeki ana karelerin ölçeklerine göre hizalanması gerçekleştirilir. Bunun için de derinlik ve fotometrik hata dikkate alınır. Döngü kapatma için, ana kare eklendikten sonra, döngü algılamaya yetecek kadar birkaç ana kare kullanılır.

Dolaylı ve doğrudan VSLAM/VO çalışmalarının avantajlarını birleştirmek isteyen Forster ve ark. (2014), oldukça hızlı ve güçlü Yarı Doğrudan Görsel Odometri (Semi-Direct Visual Odometry (SVO)) algoritmasını önermişlerdir. SVO, VO'yu oldukça maliyetli kılan özellik çıkarma ve eşleştirme tekniklerini ortadan kaldırır. SVO, özellik tabanlı yöntemlerin (birçok özelliğin takibi, paralel izleme ve haritalama, ana kare seçimi) başarı faktörlerini doğrudan yöntemlerin doğruluğu ile birleştirdi. Doğrudan yöntemler, hareketi ve yapıyı, doğrudan görüntüdeki yoğunluk değerlerinden tahmin eder. Bu yüzden zayıf dokulu, odaklanmamış ve hareket bulanıklığı olan görüntülerde, özellik tabanlı yöntemlere göre daha başarılıdır. Algoritmanın biri kamera hareketini tahmin etmek için diğeri ise ortam araştırılırken haritalamak için iki paralel işlem gerçekleştirir. Hareket tahmininde, seyrek model tabanlı görüntü hizalaması ile poz başlatma gerçekleştirilir. Haritalamada ise, 3B noktaya karşılık gelen her 2B özelliği için bir olasılıksal derinlik filtresi başlatılır. Bir özelliğin derinlik tahmini, olasılık dağılımı ile modellenmiştir. Her bir yeni gözlem alındıkça Bayes tabanlı olarak güncelleme yapılır. Gerçekleştirilen algoritma, GPS'in olmadığı kapalı ortamda mikro hava aracı ile test edilmiştir ve oldukça hızlı çıktılar alınmıştır.

Dolaylı çözümler için PTAM hızlı çözümler sağlamasına rağmen büyük ortamlarda yetersizdi. Bu dezavantajı gidermek için PTAM'ın temel işleyişini temel olarak alan Mur-Artal ve ark. (2015), yüksek performanslı iç ve dış mekanlar için özellik tabanlı monoküler optimizasyon tabanlı ORB-SLAM'ı sunmuştur. Ana tasarım fikirlerinden biri, haritalama ve takip tarafından kullanılan aynı özelliklerin, yeniden konumlandırma ve döngü algılamayı gerçekleştirmek için yer tanımada da kullanılmasıdır. Özellik çıkarma için, 256-bit tanımlayıcı (descriptor) ile yönlendirilmiş çok ölçekli FAST köşeleri olan ORB kullanılmıştır. ORB, bakış açısına göre iyi bir değişmezlik sağlarken, hesaplama ve eşleştirmesi son derece hızlıdır. Bu da BA'nın doğruluğunu artırır. Sistem, paralel olarak çalışan izleme, yerel haritalama ve döngü kapatma iş parçacıklarını birleştirmektedir. İlk olarak, bir önceki kare ile bir başlangıç

özellik eşlemesi için ORB kullanılır. İzleme, kamerayı her kareyle lokalize etmek ve ne zaman yeni bir ana kare (keyframe) eklenileceğine karar vermekle sorumludur. Özellikler önceki çerçeveyle eşleştirilir ve poz yalnızca hareketli (motion-only) BA kullanılarak optimize edilir. Takip iş parçacığı yeni bir ana kare eklenip eklenmediğine karar verir. Her bir harita noktası, ORB özelliklerini temsil eder. Dolayısıyla harita noktaları FAST ile tespit edilen köşelerin üçgenleştirilmesi sonucu oluşur. Döngü kapatma, her yeni ana kare ile döngüler arar. Bir döngü algılanırsa, döngüde biriken sapma hakkında bilgi veren bir benzerlik dönüşümü hesaplanır. Bir döngüyü düzeltmek ve küresel tutarlılığı sağlamak için, döngü kapatma hatasını grafik boyunca dağıtan bir poz grafiği optimizasyonu (Pose Graph Optimization (PGO)) kullanılmıştır. Sistemde, döngü tespiti ve yeniden lokalizasyon gerçekleştirmek için, dağıtılmış kelime çantası (Distributed Bag of Words (DBoW)), yer tanıma modülü kullanılmıştır. Bu algoritmalar kullanılarak gerçekleştirilen ve farklı veri setleri üzerinde yapılan deneylerde ORB-SLAM üstün özellik göstermiştir.

Diğer optimizasyon tabanlı yöntemlerden farklı olarak Munguía ve ark. (2016) İHA'lar için Genişletilmiş Kalman Filtresi'ne (Extended Kalman Filter (EKF)) dayanan hızlı durum tahmini sağlayan, bir filtre tabanlı konumlandırma yöntemi önermişlerdir. Önerilen çalışma, bir yönlendirme sensörü (Attitude and Heading Reference System (AHRS)), bir konum GPS sensörü ve bir monoküler kamera tarafından elde edilen ölçümleri birleştirir. Konum sensörü, yalnızca ortamın ilk metrik ölçeğini elde etmek için başlangıç kısmında kullanılır. Ardından, işaretçilerin tahmini haritası, konum sensörü bulunmadığında tamamen görmeye dayalı bir navigasyon gerçekleştirmek için kullanılmıştır. Bu sebeple, bu çalışmada GPS sinyalinin işlem başında kısa bir süre boyunca bilindiği varsayılmaktadır. Tamamen monoküler SLAM yaklaşımlarından farklı olarak, bu çalışmada, monoküler SLAM sistemleri ile ilgili bazı teknik zorlukların üstesinden gelmek için İHA'larda yaygın olarak bulunan sensör setinden yararlanmıştır. Bir monoküler kamera kullanıldığında, derinlik bilgisi tek bir çerçevede alınamaz. Bu çalışmada, bu amaçla yeni bir stokastik üçgenleme yöntemi geliştirilmiştir. Simülasyonlar ve gerçek verilerle elde edilen deneysel sonuçlar, kamera ölçümlerinin sisteme dahil edilmesinin faydalarını göstermektedir.

Özellik tabanlı yöntemler dokusuz ortamlarda hatalı çalışır. Gerçek dünya üzerinde yeterli dokuya sahip olmayan ortamlar çoktur. Bu durumu ele alan Pumarola ve ark. (2017), düşük dokulu sahneler, nokta benzerliklerinin ve özellikle görsel SLAM algoritmalarının zayıf noktası olduğundan, VSLAM için PL-SLAM yöntemini

önermişlerdir. Çalışmaya göre, çizgiler noktalar kadar literatürde iyi çalışılmamıştır ve güvenilir değildir. Ancak, çizgiler, iki aşamalı bir optimizasyon işleminden sonra görüntü düzlemindeki tam konumu tahmin edilen uç noktalarına göre parametrelendirilirse, SLAM için nokta özellikleri gibi kullanılabilir. Amaç, noktalar ve çizgileri birleştirerek, noktaların çoğu giriş görüntülerinden kaybolduğunda da çalışabilecek bir SLAM yöntemi geliştirmektir. Çalışmada, ORB-SLAM hem nokta hem de çizgi benzerliklerini ele alacak şekilde genişletilmiştir. Aynı zamanda hesaplama maliyeti de korunmuştur. Nokta ve çizgi birleştirmesi işlemi için EKF-SLAM kullanılmıştır. Yöntem çok dokulu ve düşük dokulu senaryolar için, TUM RGB-D veri seti kullanılarak uygulanmıştır. Sonuç olarak da çizgilerin kullanımı ile zayıf dokulu karelerde orijinal ORB-SLAM'in performansının arttığı belirtilmiştir.

Zhang ve ark. (2018b) yeni bir VSLAM yönteminden ziyade, var olan yöntemi daha da kapsamlı hale getirmeyi amaçladı. Bu kapsamda, geometrik SLAM'a dayalı nesne seviyesinde semantik haritalar oluşturulmuş ve semantik haritalara göre lokalizasyon doğruluğunu iyileştiren bir RGB-D semantik SLAM sistemi sunulmuştur. Önerilen sistem, ORB-SLAM temel alınarak inşa edilmiştir. Uygulamada elde edilen haritalar, engeller ve konumlarıyla ilgili bilgiler içerir. Uygulama açısından, sistem iki ana modüle sahiptir: birinci modül, nesnelerin konumları hakkında bilgi sağlayan ve aynı zamanda 3B haritasını oluşturan seyrek özelliğe dayanan RGB-D SLAM'dir. Diğeri ise derin öğrenme yöntemiyle gerçekleştirilen nesne algılamadır. Çalışmada anlamsal ve geometrik bilgili ayrı nesne modelleri içeren, anlamsal bilgiye sahip bir ortamın nokta bulutları haritası oluşturulur. Ayrıca, sistemin hesaplama etkinliğini arttırmak için, FLR (Fast Line Rasterization) Algoritmasına dayanan geliştirilmiş bir Octomap oluşturulmuştur.

Genel olarak VSLAM/VO çalışmaları için ortamın tamamen statik olduğu kabul edilir. Ancak bazı çalışmalarda, ortamda dinamik bir nesnenin varlığı da göz önüne alınır. Wang ve ark. (2019)'ın yaptığı çalışma dinamik nesneleri tanımlamak ve ardışık karelerdeki gereksiz bilgileri elemeyi amaçlayan semantik tabanlı bir çalışma sundu. Bu amaçla, yazarlar özellik noktalarının sayısını artırarak ve özellik noktalarını eşit şekilde dağıtarak göreceli konumlandırma doğruluğunu geliştirmek için arama tablosu (Lookup Table (LUT)) yöntemi önermişlerdir. Bu sayede gereksiz veriler azaltılarak harita oluşturmanın verimliliği artırılmıştır. Uygulamada, Kinect veya RealSense gibi bir RGB-D kamera sensörü kullanılır. RGB-D giriş verileri bir çevrimdışı eğitim derin öğrenme ağı (YOLOv3) ile işlenerek dinamik bir sahnedeki yayalar veya bir masadaki bardaklar

gibi sahnedeki belirli nesneler algılanabilir ve SLAM işleminde hareketli nesneler kaldırılabilir. Çalışma sonucu, önerilen SLAM çözümünün harita oluşturma için zaman tüketimini 2/3 oranında azaltarak hesaplama açısından verimli olduğunu göstermiştir.

Qin ve ark. (2019) maliyeti göz önüne almadan oldukça fazla donanıma sahip bir navigasyon uygulaması gerçekleştirdi. GPS'in çalışmadığı 3 boyutlu bilinmeyen ortamlarda, otonom keşif, haritalama ve navigasyon için İHA ve insansız kara aracını (Unmanned Ground Vehicles (UGV)) birlikte kullandılar. Sistemin dokulu ve dokulu olmayan ortamlarda sağlamlığını artırmak için nokta bulutu tabanlı lokalizasyon ve haritalama kullanılmıştır. Algılanan noktalar için gerçekleştirilen görevler özellik çıkarma, özellik ilişkilendirme, hareket tahmini ve haritalamadır. Heterojen araçların avantajlarını kullanmak için keşif ve haritalama görevleri iki katmana ayrılmıştır. İlk katmanda, bir 3B LIDAR monte edilen UGV ile bir ön keşif yapmak ve bir kaba haritalama üretmek amaçlanmıştır. Bu işlem tamamlandığında, ilk katmanda inşa edilen birincil harita, İHA tarafından temel bir ortam modeli ve bir navigasyon referansı olarak alınır. Bunun için UGV tarafından oluşturulan harita İHA ile paylaşılır. İkinci katmanda İHA, eğimli bir 2B lazer modülü ve görüş sensörleri kullanarak tamamlayıcı bir ince haritalama gerçekleştirmek için mobilite avantajından yararlanır ve ilk katmanda kalan boşlukları doldurur. Gerçekleştirilen uygulama hem simülasyon hem de deneysel olarak uygulanmıştır.

2.1.2. Görsel-Atalet SLAM (VISLAM)/ Görsel-Atalet Odometri (VIO) Çalışmaları

Bu bölümde kameranın yanında IMU sensörü de içeren VISLAM ve VIO çalışmalarının bir kısmı hakkında bilgi verilmiştir.

Clement ve ark. (2015), ego hareket tahmininde kullanılan iki modern yaklaşım olan MSCKF ve Kayan Pencere Filtresi (Sliding Window Filter (SWF)) performanslarını, filtrelerin doğruluğunu ve tutarlılığı açısından karşılaştırmıştır. Her iki filtrede de bir aracın hareketini tahmin etmek için bir IMU kullanılır ve daha sonra bu tahmin bir monoküler kamera görüntüsünden çıkarılan belirgin özelliklerle düzeltilir. Karşılaştırma, elde tutulan hareketli bir sensör teçhizatından alınan veriler ve KITTI veriseti kullanılarak yapılmıştır. Uygulamada kullanılan iki veri kümesi de bir stereo kamera ve IMU'dan elde edilen verileri içerir. Algoritmalar monoküler kamera için tasarlandığından, stereo kameradaki sadece sol kameradan gelen görüntüler kullanılmıştır. Özellik yoğunluğu baz alınarak yapılan deney sonucunda, özellik sayısı arttıkça MSCKF'nin performansı artarken, SWF'nin performansı artan özellik sayısından önemli ölçüde etkilenmemiştir.

Yani, MSCKF'nin özellik yoğunluğuna SWF'den daha duyarlı olduğu kanıtlanmıştır. Ayrıca nispeten özelliksiz ortamlarda, salt IMU entegrasyonu her iki filtreden daha iyi performans göstermiştir. SWF, çoğu durumda MSCKF'den daha doğru poz tahminleri üretmiş olmasına rağmen, MSCKF, SWF'den daha az hesaplama yoğunluğuna sahiptir. Sonuç olarak, hesaplama kaynakları sınırlı ve ortam zengin özelliklere sahipse MSCKF'nin, doğruluğun çok önemli olduğu durumlarda ise SWF'nin tercih edilmesi gerektiği çıkarımı yapılmıştır.

Mur-Artal ve Tardós (2017a), döngüleri kapatabilen ve zaten haritalanmış alanlarda sıfır-kaymalı lokalizasyon elde etmek için haritasını yeniden kullanabilen (map reuse), sıkıca bağlanmış (tightly coupled) yeni bir monoküler VISLAM yöntemi, VI-ORB yöntemini tanıtmışlardır. Ayrıca ölçeği, yerçekimi yönünü, hızı, jiroskop ve ivmeölçer sapmalarını kısa sürede ve yüksek doğrulukla hesaplayan yeni bir IMU başlatma yöntemi önerilmiştir. Takip işleminde, sabit bir harita varsayılarak mevcut kare optimize edilir ve arka uçta ana kare tabanlı yerel BA gerçekleştirilir. Tam yumuşatmanın (full smoothing) aksine, bu yaklaşım sabit zamanlı yerel BA'ya izin verir ve geçmiş durumları marjinalleştirmeyerek bunları yeniden kullanabilir. Yer tanıma kullanarak büyük döngüler tespit edilir ve ayrı bir iş parçacığında, gerçek zamanlı çalışmaya müdahale etmeyecek şekilde, tam BA kullanarak, hafif bir poz grafik optimizasyonu ile bu döngüler düzeltilir. Bu aşamadan önce doğru durum tahminleri sağlayan güvenilir bir görsel ataletsel başlatma gereklidir. Bunun için, en uygun çözümü sağlayan görsel ataletsel bir tam BA gerçekleştirilmesi önerilmiştir.

Daha hızlı ve faklı bir yöntemle navigasyon sağlamak isteyen Wu ve ark. (2017), VINS sistemi için Invariant-EKF (IEKF) tabanlı bir yöntem önermişlerdir. IEKF fikri, sistemin simetrilerinden yararlanmak için EKF denklemlerinin değiştirilmesidir. IEKF, simetrilere (veya değişmezliklere (invariances)) sahip doğrusal olmayan sistemler için EKF'nin bir versiyonudur. Geleneksel EKF esaslı VINS, yerçekimi yönü etrafında bir çeviri (translation) ve dönüş ile ilişkili stokastik gözlemlenemez dönüşüm altında değişmez (invariant) değildir. Bu da temel fiziksel sisteme aykırı olduğundan tutarsız durum tahminleri meydana gelebilir. Bu sorunu çözmek için Right Invariant Error-EKF (RIEKF-VINS) filtresi önerilmiştir. Değişmezlikten kastedilen, filtrenin çıktısının herhangi bir stokastik gözlemlenemez dönüşüm altında değişmemesidir. VINS sistemi için gözlemlenemez dönüşüm, yerçekimi yönü etrafındaki çeviri ve dönüştür. Önerilen filtre tutarlı bir durum tahmincisi elde etmek için MSCKF'ye entegre edilmiştir. Yapılan gerçek dünya deneyleri ile algoritmanın üstünlüğü vurgulanmıştır.

Mobil cihazlarda gerçek zamanlı Artırılmış Gerçeklik (Augmented Reality (AR)) uygulamaları üzerine çalışma sunan Piao ve Kim (2017), adaptif bir monoküler VISLAM yöntemi sunmuşlardır. Önerilen yöntem, ORB özellik tabanlı SLAM olan monoküler ORB-SLAM'a dayalı modifikasyon ve uygulama içerir. İlk olarak, bir kamera girişi ile IMU sensörünü sıkı bağlı olarak birleştiren bir VIO yöntemi tasarlanmıştır. İkinci olarak da mobil cihaz ortamında AR uygulamalarının gerçek zamanlı daha hızlı izlenmesini desteklemek için, OF tabanlı hızlı VO modülü tasarlanmış ve mevcut ORB-SLAM sistemi ile birleştirilmiştir. Son olarak, IMU sensör değerindeki değişikliğe göre izleme modülünü adaptif olarak seçen bir yöntem önermişlerdir. Önerilen Adaptif VISLAM'in ana mimarisi, ORB-SLAM ile tutarlı olarak izleme, yerel haritalama ve döngü kapatma gibi modüllerini içerir. Çalışma deneysel olarak uygulanmış ve başarılı sonuçlar elde edilmiştir.

Mevcut yöntemleri ilave sensör ölçümleri ile güçlendirmeyi amaçlayan López ve ark. (2017), IMU, kamera ve lazer sensörlerine dayalı Mikro Hava Araçları (Micro Aerial Vehicles (MAV)) için SLAM sisteminin geliştirilmesine ve farklı SLAM yöntemlerinin birleştirilmesine yönelik bir çalışma gerçekleştirmişlerdir. Bu çalışmada gerçekleştirilen uygulama IMU ve lazer ölçümlerine dayanır, ve EKF ile tahmin yapar. Daha sonra önerilen bu yöntem ORB-SLAM ve LSD-SLAM olarak iki yeni ve daha hassas yöntem üzerinde karşılaştırılmıştır. ORB-SLAM, iyi aydınlatılmış ve yeterince özellikli ortamlarda daha doğru çalışmıştır. LSD-SLAM'in özelliksiz ortamlarda daha iyi bir seçim olabileceği belirtilmiştir. VSLAM'ın tahminini, yerleşik sensör ölçümleriyle birleştirme önerisinin poz tahmininde önemli bir iyileşme sağladığı kanıtlanmıştır.

VO uygulamalarını radar verileriyle füzyon etmeyi amaçlayan Mostafa ve ark. (2018), GPS erişimi olmayan kapalı ortamlarda İHA navigasyonu için Radar odometrisi (RO) ile VO'yu birleştirmiştir. RO yağmur, sis ve toz gibi çevresel koşullardan az etkilendiğinden, bu bilgiyi VO ile birleştirmek verimlidir. Bu anlamda kamera görüntülerinin olumsuz etkilendiği yerlerde RO çıktısının doğruluğuna güvenilir. Ayrıca bu çalışma, RO'dan elde edilen hız bilgisini ve monoküler VO'yu bir EKF aracılığıyla IMU, barometre ve manyetometre ölçümleriyle birleştirir. Deneysel sonuçlar, önerilen sistemin GPS sinyal kesintisi durumunda 3B konumlandırma doğruluğunu artırdığını göstermektedir.

Qin ve ark. (2018), bir kamera ve bir düşük maliyetli IMU kullanarak 6 eksenli durum tahmini için monoküler bir görsel atalet sistemi (Visual-Inertial System (VINS)) önermişlerdir. VIO, başlangıçta doğru bir başlangıç tahmini gerektiren doğrusal olmayan

bir sistemdir. Başlangıç değerleri, IMU değerlerinin, salt görsel (vision-only) yapısı ile örtüştürülmesi suretiyle elde edilmiştir. Tahmincinin başlatılmasından sonra, yüksek doğruluk ve sağlam durum tahmini için kayan pencere tabanlı monoküler VIO gerçekleştirilmiştir. IMU ölçümlerini ve gözlemlenen özellikleri birleştirmek için doğrusal olmayan optimizasyon tabanlı bir yöntem kullanılmıştır. Görsel ölçümler için birbirini takip eden kareler arasındaki özellikler izlenmiş ve en son karedeki yeni özellikler tespit edilmiştir. IMU ölçümleri için bunlar ardışık iki kare arasına önceden entegre edilir. Her yeni görüntü için mevcut özellikler KLT seyrek OF algoritması ile izlenir. Aykırı noktalar RANSAC ile elenir. Ana kare seçimi için iki kriter kullanılmıştır. Birincisi, özelliklerin önceki ana kareden farklı ortalama mesafede olmasıdır. Diğer kriter kaliteyi izlemektir. Yani izlenen özellik sayısı belirli bir eşiğin altına düşerse, bu kareyi yeni bir ana kare olarak ele alınır. VIO ve poz grafiği optimizasyon modülleri, ayrı iş parçacıklarında eşzamanlı olarak çalışır.

Mevcut VIO uygulamalarının İHA tabanlı kapsamlı bir karşılaştırması Delmerico ve Scaramuzza (2018) tarafından yapılmıştır. Yazarlar monoküler VIO ile sınırlı olacak şekilde, uçan robot sistemlerine özgü donanım konfigürasyonları üzerinde açık kodlu VIO algoritmalarının değerlendirmesini yapmıştır. Monoküler olmasının sebebi, diğer sensör konfigürasyonlarına göre düşük ağırlık ve güç tüketimi nedeniyle uçan robotlar için popüler olmasından kaynaklanır. Kullanılan yöntemler Multi-State Constraint Kalman Filter (MSCKF) (Mourikis ve Roumeliotis, 2007), Open Keyframe-based Visual-Inertial SLAM (OKVIS) (Leutenegger ve ark., 2015), Robust Visual Inertial Odometry (ROVIO) (Bloesch ve ark., 2015), Monocular Visual-Inertial System (VINS-Mono) (Qin ve ark., 2018), Semidirect visual odometry (SVO) (Forster ve ark., 2016b) + MultiSensor-Fusion (MSF) (Lynen ve ark., 2013) (SVO-MSF) (Faessler ve ark., 2016) ve SVO + GTSAM (Georgia Tech Smoothing and Mapping Library) (SVO-GTSAM) (Forster ve ark., 2016a)'dır. Bu algoritmalar, uçan robotlara özgü altı serbestlik dereceli (6DoF) yörünge içeren EuRoC MAV veri setleri üzerinde gerçekleştirilmiştir. Donanım olarak masaüstü bilgisayar (Intel NUC), tek kart gömülü bilgisayar (Up Board, ODROID) ve Intel Core i7, 2.80 GHz, 32 GB RAM özellikli dizüstü bilgisayarı kullanılmıştır. Sonuç olarak VINS-Mono'nun performansı, tüm donanım platformlarında en tutarlı ve doğru sonuç vermiştir. Ancak Up Board veya ODROID gibi, hesaplama olarak kısıtlanmış bir donanım platformu göz önüne alındığında, SVO + MSF, en doğru performansı verir. Karşılaştırmada, kare başına işlem süresini, CPU kullanımı ve hafıza kullanımı ölçütleri dikkate alınmıştır.

Hızlı bir VIO uygulaması tanıtan Quan ve ark. (2019b), robotların dinamik hareketine hızlı tepki vermesini sağlamak amacıyla, hareketi izlemek için EKF tabanlı VIO uyguladı. Birinci iş parçacığı, daha az zaman gecikmesi ve sağlam hareket takibi sağlamak için EKF-VIO ve geri bildirim mekanizmasını içerir. İkinci iş parçacığı harita oluşturmak ve yerel BA yapmak için, üçüncüsü ise döngü kapatmayı gerçekleştirmek için kullanılır. Son iki iş parçacığı tutarlı bir küresel harita oluşturmak için kullanılmıştır. Birinci iş parçacığındaki EKF doğrusallaştırma hataları nedeniyle, tutarsız olma eğilimindedir, bu da büyük tahmin hatası ve sistemin sapmasına neden olur. BA teknikleri filtreleme tekniklerinden daha iyi doğruluk sağlayabilir. Bu nedenle, tutarlı bir küresel harita oluşturmak için BA tekniğini kullanılmıştır. BA zaman alıcı olduğundan, her kare için EKF-VIO'yu gerçekleştirdikten sonra ana kareler seçilmiştir. İkinci iş parçacığında, küresel (global) haritayı oluşturmak için ana kareler kullanılmış ve lokal haritayı optimize etmek için ana kareye dayalı görsel ataletsel bir lokal BA gerçekleştirilmiştir. Üçüncü iş parçacığında önceden eşlenmiş bir alana geri dönerken biriken kaymayı azaltmak için bir döngü kapatma modülü gerçekleştirilmiştir. Önerilen algoritmanın üstünlüğü deneylerde doğrulanmıştır.

VIO için yeni bit hibrit filtre tabanlı çözüm geliştiren Heo ve ark. (2019), Lokal-Optimal-Çok Durumlu Kısıtlamalı Kalman Filtresi (Local-Optimal-Multi-State Constraint Kalman Filter (LOMSCKF)) isimli bir ölçüm modeli tasarlamışlardır. Bu model ile, VIO gerçekleştirmek için MSCKF ile doğrusal olmayan optimizasyon yöntemi füzyon edilmiştir. Ayrıca MSCKF'nin aksine, kayan pencerede mevcut olan ölçümler ve bilgilerin tamamı kullanılmıştır. LOMSCKF, kayan penceredeki ön hataların yanı sıra, yansıma hatalarını ve IMU hatalarını en aza indirerek kamera durumlarını ve yapıları kayan pencere üzerinde optimize eder. Bunun için ölü takip görsel ölçümleri, ön bilgi ve önceden entegre edilmiş IMU ölçümleri kullanılarak grafik optimizasyonu gerçekleştirilir. Optimize edilmiş bu sonuçlar ölçüm bilgisini oluşturur. Sonuç olarak bu çalışma, MSCKF sistemi altında görsel ataletsel bilgilerin verimli bir şekilde işlenmesine yönelik yeni bir yöntem sunmuştur. Uygulama Matlab ortamında gerçekleştirilmiştir. Hem sanal hem de gerçek dünya veri kümeleri kullanılarak önerilen algoritmanın performansı kanıtlanmıştır.

2.1.3. Derin Öğrenme Tabanlı V(I)O/V(I)SLAM çalışmaları

Yukarıda bahsedilen geleneksel yöntemlerin yerini günümüzde veri tabanlı yapay zekâ yöntemleri almaya başladı. Özellikle derin öğrenme tabanlı sistemlerin görsel

bilgilerden yüksek seviyeli özellikler elde etmesi ve ölçek tahmini, elle özellik çıkarma, kalibrasyon, aykırı değer reddi gibi adımlara gerek duyulmamasını sağladı. Bu da başarılı, özelliksiz ortamlara karşı dirençli, dinamik ışıklandırmaya dayanıklı ve kolay çözümlerin önünü açtı (Chen ve ark., 2018; Chen ve ark., 2020; Wang ve ark., 2020c; Zhao ve ark., 2021). Bu bölümde daha modern olan derin öğrenme tabanlı V(I)O/V(I)SLAM çalışmalarından bahsedilmiştir.

CNN ve RNN yapısı kullanarak DeepVO'yu ilk tanıtan Wang ve ark. (2017), KITTI (Geiger ve ark., 2013) veri setinden ardışık kareleri kullanarak poz tahmini gerçekleştirdi ve derin öğrenme tabanlı V(I)SLAM/V(I)O akımını başlattı. Ayrıca bu çalışma geleneksel yöntemlere göre öğrenme tabanlı yöntemlerin farkını ve üstünlüğünü ele aldı. Daha sonra oldukça rağbet gören bu alanda önemli yaklaşımlar geliştirildi. Şimdiye kadarki çalışmalarda genel olarak benimsenen teknik DeepVO'da olduğu gibi bağıl (relative) poz tahminidir. Bu çalışmalar genellikle iki ardışık karenin özelliklerini, CNN ve/veya RNN aracılığı ile çıkararak mevcut karenin konumunu tahmin eder. DeepVO, ilk olarak, ardışık iki monoküler kamera karesini yığın halinde birleştirir ve tasarladığı bir CNN mimarisi ile özellikler çıkarır. Daha sonra bu özellikler, sıralı öğrenme (sequential learning) sağlamak için RNN tabanlı LSTM katmanları ile işlenir ve altı çıkışlı poz tahmini gerçekleştirilir. Sonuçlar modern geometri tabanlı yöntemlerle rekabet edebilecek bir performans göstermiştir.

Derin öğrenme tabanlı VO yaklaşımına atalet bilgisini de ilave ederek daha başarılı performans sağlamak isteyen Clark ve ark. (2017) VINet mimarisini önerdi. Çalışmada uçtan uca (end-to-end) yaklaşımı ile görsel bilgilere ek olarak atalet bilgileri de kullanan bir mimari (VINet) tasarlandı. Mimaride CNN katmanları ile ardışık karelerden, RNN tabanlı LSTM katmanları aracılığıyla ise atalet sayısal verilerinden özellikler çıkarıldı. Son olarak bu özellikler birleştirilerek LSTM ve SE(3) katmanı ile odometri tahmini gerçekleştirildi. VINet yaklaşımı, KITTI ve EuRoC veri seti üzerinde yaptığı deneysel çalışmalarla, optimizasyon tabanlı bir sensör füzyon yaklaşımı olan OKVIS (Leutenegger ve ark., 2015)'ten daha iyi performans gösterdi.

Danışmanlı öğrenme yöntemlerinden farklı olarak, Han ve ark. (2019), DeepVIO olarak adlandırılan çalışmalarında, kendi kendini denetleyen bir uçtan uca mimari kullanarak stereo kamera kareleri aracılığıyla bir çalışma gerçekleştirdiler. Stereo kareleri ile her sahnenin derinliğini ve 3B nokta bulutu tahmin edilir. Daha sonra ağ denetleyicisi olarak 3B OF ve 6-DoF pozu içeren geometrik kısıtlamaları elde eder. Daha sonra bu kısıtlamalardan faydalanılarak, 2B OF CNN ağının çıktısı ve IMU girişli LSTM ağının

çıktısını birleştirdiler. Deneysel çalışmalar KITTI (Geiger ve ark., 2013) ve EuRoC veriseti üzerinde uygulandı.

Benzer şekilde Almalioglu ve ark. (2019) sıklıkla tercih edilen danışmanlı (supervised) yöntemler yerine veri sınırlaması engellemek için, kendi kendini denetleyen (self-supervised) derin öğrenme tabanlı VIO (SelfVIO) modelini tanıttı. Etiketlenmemiş monoküler RGB görüntüleri ve IMU ölçümlerini sırası ile CNN ve LSTM aracılığı ile işleyerek, poz ve derinlik tahmini gerçekleştirdi. Görsel-ataletsel özellikleri çıkarılan ağın en uygun özellik kombinasyonunu otomatik olarak öğrenmesi için bir dikkat (attention) mekanizması kullanılmıştır. Sonuçları değerlendirmek için EuRoC (Burri ve ark., 2016), KITTI (Geiger ve ark., 2013) ve Cityscapes (Cordts ve ark., 2016) veri setleri kullanıldı.

Öğrenme tabanlı VIO çıktılarını ilave sensörle güçlendirmek isteyen Saputra ve ark. (2020), kamera ile elde edilen görüntüleri, IMU'ya ek olarak termal görüntülerle birleştiren DeepTIO yaklaşımını geliştirdi. VI birleşim için VINet çalışmasından faydalanıldı. Bu mimarideki ham kamera görüntülerinden çıkarılan özellikler tasarlanan bir halüsinasyon ağı ile termal görüntülerden çıkarılan özelliklerle seçici füzyon tekniği ile birleştirildi. Çalışma sonucunda, oluşturulan veri seti için oldukça başarılı bir odometri tahmini sağlanmıştır.

Derin öğrenme kullanan bazı farklı çalışmalar yukarıda bahsedilenlerden farklı olarak ego hareket tahmini için OF tabanlı bir stratejiyi benimser. Bu sayede ardışık görüntüler arasındaki hareket değişiminin baz alınması sağlanır. İki ya da daha fazla ardışık görüntünün OF görüntüsü, geleneksel yöntemlerdeki geometri tabanlı poz tahminine daha yakındır. Bu sayede geleneksel ve öğrenme tabanlı yaklaşımların birlikte kullanılması sağlanır. Aşağıdaki çalışmalar OF tabanlı VIO içerir.

Ban ve ark. (2020) derin öğrenme ve geleneksel yaklaşımların avantajlarını birleştirmek için monoküler DL_Hybrid VO modelini tanıttı. Modelde öncelikle OF ve derinlik bilgisini elde etmek için iki adet derin mimari tasarlandı. Daha sonra OF ve derinlik bilgileri kullanılarak 2B-2B ve 3B-2B geometrik dönüşümler kullanılarak odometri gerçekleştirildi. Deneysel çalışmalar KITTI üzerinde uygulandı.

Zhao ve ark. (2021) ardışık görüntülerden OF alanlarını çıkarmak için PWC-Net (Sun ve ark., 2018) ağını kullandı. Daha sonra bu OF karelerinin özellikleri, kodlayıcı (encoder) yapısındaki CNN mimarisi ile çıkarıldı. Bu özellikler daha sonra ardışık özellikleri analiz edebilen RNN tabanlı mimariye verildi. RNN 6 DoF bilgilerini tahmin ederken, aynı zamanda alt yolda, ağın kodlayıcının kaybıyla denetimsiz bir şekilde ayrı olarak eğitilebilmesi için OF'yi yeniden yapılandıran bir kod çözücü (decoder) kullanılır.

Deneysel çalışmalar KITTI (Geiger ve ark., 2013) ve Malaga (Blanco-Claraco ve ark., 2014) veri setleri üzerinde gerçekleştirildi.

Son olarak, OF kullanan başka bir çalışmada ise Pandey ve ark. (2021), poz tahmini için ham görüntüler yerine OF karelerini kullandılar. Ardışık karelerin OF alanlarını LiteFlowNet (Hui ve ark., 2018) ağı ile elde ettiler. Daha sonra bu OF alanlarını kendi oluşturdukları CNN-BiLSTM ağına beslediler. Önerilen yöntem KITTI (Geiger ve ark., 2013) veri seti üzerinde uygulanmıştır.

2.2. Yol Planlama Çalışmaları

Otonom uygulamalar için başarılı bir konumlandırmadan sonraki ilk adım görev noktaları arasındaki en uygun yol planlama koordinatlarını oluşturmaktır. Daha hızlı ve verimli yol planlama algoritması geliştirmek için çalışmalar son zamanlarda gittikçe artmaktadır. Yol planlama algoritmalarında araştırmacıların amacı, büyük ortamlarda bile hızlı şekilde en kısa yolu bulan algoritmalar geliştirmektir. Genel olarak şimdiye kadarki yapılan çalışmalar çoğunlukla şu ikisinden birine odaklanır: Mevcut yol planlama yöntemlerini algoritmik olarak değiştirerek daha optimum yöntem geliştirmek, ya da mevcut yol planlama yöntemleri ile birlikte optimizasyon veya yapay zeka algoritmalarını birleştirerek daha doğru/hızlı sonuçlar üretmek. Her ikisi için yapılan önceki çalışmalar sırasıyla bölüm 2.2.1 ve bölüm 2.2.2'de anlatılmıştır.

2.2.1. RRT* Tabanlı Literatür Çalışmaları

RRT*'ı temel alarak önerilen yeni yaklaşımlar arasından, Adiyatov ve Varol (2013), bellek tüketiminin artmasını engelleyerek optimal yolu bulmak için RRT*FN'yi tanıttılar. Çünkü, RRT*'da düğüm sayısı arttıkça, optimal yolu bulma ihtimali artar, fakat her bir yeni düğüm yeni bir depolanması gereken değişkendir. Bu nedenle sınırlı belleğe sahip gömülü sistemler için bellek tüketimini sınırlandırarak tasarruf sağmayı amaçladılar. Düğüm sayısının azalması, ya da sabit tutulması ayrıca arama sürecini hızlandırır. Fakat az düğüm hedefe daha yakın yolu sağlayabilecek düğümlerin kaldırılmasına neden olabilir, bu da RRT*'dan daha uzun/maliyetli yolların bulunmasıyla sonuçlanır. Sonuç olarak bellek tüketimini azaltma pahasına, yol kalitesinin düşmesine izin verilmiştir. Ayrıca deneyler 2B ortamlarla gerçekleştirilmiştir.

RRT*'ın geliştirilmesi için önerilen başka bir çalışmada, Gammell ve ark. (2014) düğüm reddetmeye (node rejection) odaklı Informed-RRT*' yöntemini önerdiler. RRT*-SMART'a benzer şekilde, mevcut yolu optimize etmek için ilk yol RRT* ile bulunur.

Bulunan ilk yola bağlı olarak başlangıç ve hedef noktaları arasında, yeni örnekleme noktalarını sınırlamak için bir elipsoit oluşturur. Algoritma tekrar bu elipsoit aralığında çalıştırılır. Bu işlem sürekli tekrarladıkça giderek elipsoitlerin alanı küçülür ve sonuçta daha kısa yol bulunur. Ancak bu ardışık hesaplama, yol bulma süresini önemli ölçüde artırmaktadır.

Diğer bir çalışmada Noreen ve ark. (2016), hedefe yavaş yakınsama problemini çözmek için akıllı örneklemeye ve yol optimizasyonuna dayanan RRT*-SMART yaklaşımını 2B ortamlar üzerinde geliştirdi. İlk yol planlaması RRT* ile yapıldı. Daha sonra bu yol üzerinde, yol düzleştirme tekniği ile yol optimize edilir, yani gereksiz ya da fazladan düğümler ilk yoldan kaldırılır. Daha sonra optimize edilmiş yolun köşeleri, akıllı örnekleme için kullanılır. Sonuçta engel köşelerine yakın düğümler ile en kısa yol üretilir. Yaklaşım, çözüm maliyetini hızla düşürür, ancak diğer homotopi sınıflarının keşfedilmemesiyle de sonuçlanabilir. Ayrıca ilk yol RRT* tarafından bulunduğundan, optimal yolun oluşturulması tamamen bu ilk yola bağlıdır. Bu durumda son yolun optimal olmama ihtimali vardır. Özellikle çok engelin olduğu ortamlarda bu ihtimal daha fazladır, ayrıca engel sayısının fazla olması daha fazla bellek tüketimine neden olur.

Qureshi ve Ayaz (2016) ise hedefe yakınsama hızını artırmak için P-RRT* yöntemini önderdiler. P-RRT* keşif sürecinde örnekleme sürecini yönlendirmek için RRT* yöntemine Yapay Potansiyel Alanı (Artificial Potential Field (APF)) (Khatib, 1986) yöntemini dahil etti. Bu sayede yinelemeyi azaltan ve hedefe daha hızlı yakınsayan bir yöntem geliştirdiler. Ancak P-RRT* ortamın karmaşıklığı arttıkça, yerel bir minimum çözümde takılmaya eğilimlidir.

Jeong ve ark. (2019) optimal çözüme yakınsamayı hızlandırarak yol uzunluğunu optimize eden Q-RRT* yöntemini geliştirdiler. Daha uygun ana düğümü seçmek ve düğümler arasındaki bağlantıyı optimize etmek için RRT*'dan farklı olarak, ata (ancestry) düğümleri dikkate alarak bir arama gerçekleştirir. Bunu üçgen eşitsizliğini kullanarak gerçekleştirdiler. Kısacası arama kapsamını genişleterek daha kısa yol bulmayı amaçlar, ancak yöntem hesaplama süresi ve en kısa yolu bulma açısından gereken yeterliliğe sahip değildir. Ayrıca deneyler sadece 2B ortamlar için geliştirildi.

Son olarak, Liao ve ark. (2021) ise, engellere daha yakın düğümler oluşturarak optimal yolu bulmak amacıyla F-RRT* yöntemini önerdiler. F-RRT* engellerden dolayı doğrudan erişilemez bir nokta ($Parent_{x_{reachest}}$) bulana kadar yalnızca rastgele örneğe en yakın düğüm noktasının ($x_{nearest}$) atalarını arar. $Parent_{x_{reachest}}$ bulunduğunda üçgen eşitsizliğine dayanarak yeni bir düğüm (x_{create}) oluşturulur. x_{create} engele yakın bir

düğümdür ve bu nedenle engele yakın olarak bulduğu düğümlerle oluşturulan yol RRT* ve Q-RRT*'a göre daha başarılıdır. Ancak üçgen eşitliğinin sağlanamadığı durumlarda algoritma doğru çalışmayacaktır.

Bu bölüme çok sayıda farklı RRT* tabanlı çalışmalar eklenebilir. Ancak genel olarak özetlemek gerekirse, RRT* tabanlı çalışmalar daha iyi örnekleme stratejileri geliştirmek ya da ağaç düğümlerini azaltmak/silmek suretiyle daha optimal yollar bulma ya da daha hızlı yakınsama amacı gütmüşlerdir. Ancak her bir RRT* varyantları genellikle RRT*'ın bir eksikliğini giderirken, diğer bir eksiklikten taviz verirler.

2.2.2. Yapay Zekâ ve Optimizasyon Tabanlı Literatür Çalışmaları

Başarıyı artırmak, karmaşık durumlarla başa çıkmak ve daha hızlı yakınsama sağlamak için, son zamanlarda yol planlamayı yapay zekâ ve optimizasyonla çözen çalışmalar artmıştır. Bu bağlamda, diğer bir gurup olan optimizasyon ve/veya yapay zekâ algoritmaları kullanarak yol planlama gerçekleştiren çalışmalar daha günceldir. Yol planlama için çeşitli Takviyeli (ya da Pekiştirmeli) Öğrenme (Reinforcement Learning (RL)) yaklaşımları (Guo ve ark., 2020; Lakshmanan ve ark., 2020; Qu ve ark., 2020) günümüze kadar sıklıkla önerilmiştir. Ancak RL tabanlı yol planlama yaklaşımları çok sayıda çevre ile etkileşimli deneyim gerektirir. Bu, gerçek dünya robotik uygulamaları için eğitimi zorlaştırır. Ayrıca yavaş yakınsama dezavantajına sahiptirler (Qureshi ve Yip, 2018; Qureshi ve ark., 2019).

Bency ve ark. (2019) OracleNet olarak adlandırılan RNN tabanlı bir hareket planlamasını çift kollu 7 DoF bir robot kol için uyguladı. Uçtan uca mimari ile robotun başlangıç pozisyonundan hedef pozisyona kadar 7 DoF eklem açısı bilgileri eğitim için kullanıldı. Eğitim için rastgele konfigürasyonlar üzerinde gerçek (ground-truth) yolların üretilmesi için zaten A* kullanılamazdı, çünkü 7 eksen için bu çok zaman alıcı olurdu. Bu nedenle RRT-Connect ile 40.000 yoldan oluşan veri kümesi için gerçek yollar oluşturuldu. Sonuçta OracleNet, robot kollarının verilen nesne konumuna doğru hareketi için, gerekli eklem açılarının sırasını hızlı bir şekilde üretti. Yazarlar önceki yöntemlerden farklı olarak, gerçek zamanlı ve hızlı bir hareket planlaması gerçekleştirdiklerini ve böylece pratikte, robot harekete başladıktan hemen sonra yolun planlanabildiğini belirttiler. Bu çalışma RNN ile hareket planlama uygulamasını tanıtarak çok hızlı bir planlama sağlasa da, ortam bilgilerini kullanmadan eğitim gerçekleştirmesi en büyük kusurudur. Bu nedenle engellere sahip ortamlar için uygulanabilirliği kanıtlanmamıştır.

Flores-Caballero ve ark. (2020) yolun toplam uzunluğunu amaç fonksiyonu olarak kullanan optimizasyon tabanlı yöntemlerle A* algoritmasını birleştirdi. A* algoritmasını doğrudan kullanmak yerine, daha az hücre içeren aynı uzunluğa sahip yol oluşturan Pruned A-Star algoritmasını kullandı. Altı farklı 3B senaryo tasarlanarak Pruned A-Star algoritması çalıştırıldı ve sonrasına bu yöntemle bulunan yollar olarak PSO (Kennedy ve Eberhart, 1995), Diferansiyel Gelişim (Differential Evolution (DE)) (Price ve ark., 2006), Genetik algoritma (Genetic Algorithm (GA)) (Davis, 1991) yöntemleri ile optimize edildi. Sonuçlar DE algoritmasının üstünlüğünü kanıtladı. Çalışmanın sonuçları başarılı olmasına rağmen, küçük bir haritada uygulanması ve A* kullanması çalışmanın eksik taraflarıdır. Çünkü büyük haritalarda bu yöntem çok maliyetli ve zaman alıcı olacaktır. Üstelik A*'dan sonra uygulanan optimizasyon işlem süresini artırır.

Wang ve ark. (2020a), RRT tabanlı algoritmaların optimum yolu daha hızlı bulmasını sağlayacak RRT*'a dayalı Neural RRT* yol planlama yöntemini sundular. Yöntem için, A* algoritması kullanarak, başarılı yol planlarını içeren bir eğitim veri seti oluşturuldu ve daha sonra bu veri seti tasarlanan bir CNN mimarisine girdi olarak verilerek eğitim gerçekleştirildi. Amaç RRT* örnekleme sürecini yönlendirmek için CNN aracılığıyla tahmin edilen örnekleme dağılımı kullanmaktı. Önerilen NRRT*, RRT* ve Informed-RRT (IRRT*) yöntemlerine göre hızlı bir şekilde yolu elde etmek ve yakınsama hızını hızlandırmak için oldukça etkiliydi. Önerilen yaklaşım, derin öğrenmeyi yol planlama problemine dahil etmesinden dolayı başarılıdır. Ayrıca gerçek yol veri setini oluşturmak için A* algoritmasının kullanılması çok mantıklıdır, çünkü bu algoritmalar küçük haritada en kısa yolu bulmayı garanti ederler. Ancak bu çalışma A* ile gerçek veri üretmek için küçük haritalar kullanmalıdır, dolayısıyla üretilen hedef yollar küçük haritalarla sınırlıdır. Aksi takdirde veri setini oluşturmak daha karmaşık ve daha zaman alıcı olur. Ayrıca tüm deneysel çalışmalar 2B ortam üzerinde gerçekleştirilmiştir. 3B ortam için veri oluşturmak çok daha karmaşık olacaktır.

Qureshi ve ark. (2021), hesaplama açısından hızlı sonuçlar üretmek için öğrenmeye dayalı Hareket Planlama Ağı (Motion Planner Networks (MPNet)) yapısını tanıttı. Uygulamada, derinlik sensörlerinden ortam bilgileri ve ilk ve istenen hedef konfigürasyonları alınır. Bir kodlayıcı ağı (Encoder network (Enet)) ile derinlik kamerasından gelen nokta bulutu bilgileri işlenir. Yani, robotun mevcut ortam bilgisine göre kodlanmış bir çıktı üretilir. Daha sonra bir planlama ağı (Planning network (Pnet)), robotun mevcut durumunu ve hedef durumunu giriş olarak alarak bir sonraki konumu üretir. İki farklı eğitim ile uygulama gerçekleştirilmiştir. Çevrimdışı toplu öğrenme

(Offline batch learning) yöntemi için eğitim aşamasında 400.000 adet yol kullanılmıştır. Uygulamalar hem 2 kollu robot kol ve hem de bir ev modeli için 3B olarak gerçekleştirilmiştir. Sonuçta yazarlar önceki yöntemlere kıyasla zaman açısından verimli olarak yol planlamasını gerçekleştirdiler. Ancak bu uygulamanın en büyük dezavantajı derinlik sensöründen faydalanmasıdır. Zira bu durum maliyet açısından bir dezavantajdır.

2.3. Literatür Çalışmalarının Değerlendirilmesi

Mobil robot otonomisini sağlamak için şimdiye kadar çok sayıda çalışma yapılmıştır. Bu anlamda son on yılda araştırmacılar, bunu minimum maliyetle sağlayan V(I)SLAM veya V(I)O sistemlerine odaklanmışlardır. Ancak yapılan çalışmaların büyük bir kısmında V(I)SLAM veya V(I)O için filtreleme ya da BA optimizasyon metotları kullanılmıştır. Bu tür geleneksel yöntemlerde kamera kalibrasyonu, harita başlatma, özellik çıkarma (örneğin ORB), özellik eşleştirme, aykırı değer reddetme (örneğin RANSAC), hareket tahmini, ölçek tahmini ve BA, PGO gibi optimizasyon işlemlerinden oluşan klasik yapı genellikle benimsenmiştir. Bu tür yöntemler (mesela ORB-SLAM) konumlandırma doğruluğu ve gerçek zamanlı uygulanma açısından üstün performans sağlasa da, karmaşık bir yapıdadırlar. Tüm işlemlerin sahip olduğu parametreler deney yapılacak ortama göre hassas bir şekilde ayarlanmalıdır. Aksi takdirde bir ortamda alınan performans, farklı bir ortamda alınmamaktadır (Wang ve ark., 2017). Aslında konumlandırmaya yönelik yöntemler doğrudan ve dolaylı (özellik tabanlı) olarak sınıflandırılır. Dolaylı yöntemler doğrudan ortamdan çıkarılan özelliklere bağlıdır. Doğrudan yöntemler (örneğin LSD-SLAM) özellik çıkarma, özellik eşleştirme gibi özelliklerin işlenmesi ile ilgili işlem adımlarını içermez. Özellikle dokusuz ya da özelliksiz ortamlara karşı, dolaylı yöntemlere göre daha dayanıklıdır. Ancak doğrudan yöntemler başarılı bir konum hesaplama için yüksek hızlı kare sağlayan ve çok iyi kalibre edilmiş kaliteli bir kamera gerektirir. Ayrıca dolaylı yöntemlere göre işlem yoğunluğu daha yüksektir (Tang, 2020). Genellikle çoğu güncel geometrik tabanlı çalışma optimizasyon tabanlıdır. Bu yöntemler genel olarak karmaşık yapıdadır, hassas ayarlamalara ihtiyaç duyar ve zaman gecikmesine neden olurlar. Buna ek olarak filtreleme tabanlı yöntemler (örneğim MSCKF) daha kolay yapıdadır ve hızlıdır. Ancak konumlandırma probleminin doğrusal olmayan yapısından dolayı, filtre tabanlı yöntemlerdeki doğrusallaştırma adımı doğruluğu önemli ölçüde olumsuz etkiler.

Son yıllarda modern bilgisayarlı görü uygulamalarında derin öğrenme tabanlı yöntemler hem anlaşılabilir olması hem de karmaşık özellikleri iyi ayırt edebilmesi

açısından ilgi görmektedir. Özellikle CNN tabanlı derin mimariler giriş görüntülerinden yüksek seviyeli özelliklerin çıkarılmasını sağlarlar. Bu güçlü özelliklerle ağın eğitilmesi, özelliksiz alanlar, hareket bulanıklığı, ortam aydınlığındaki değişiklikler gibi özellik tabanlı yöntemlerdeki hataya neden olan problemleri kolaylıkla çözebilir. Ayrıca doğrudan ve dolaylı yöntemler için önemli bir hassas ayarlama gerektiren kamera kalibrasyonu problemi, derin öğrenme yöntemleri için geçerli değildir. Çünkü derin öğrenme tamamen veri tabanlıdır. Hem sağladığı kolaylıklar, hem elde edilen sonuçlar hem de geliştirilebilirliği göz önüne alındığında derin öğrenme tabanlı yöntemlerin, mevcut geometrik tabanlı geleneksel yöntemlerin üstesinden gelemediği problemleri çözebileceği düşünülmektedir. Özellikle görsel ortamlardaki nesnelerin algılanarak, akıllı konumlandırma uygulamalarının derin öğrenme ile geliştirilebileceği öngörülmektedir (Chen ve ark., 2018; Chen ve ark., 2020). Bu bağlamda, geliştirilebilirlik, ucuz maliyet, gerçek zamanlı uygulama ve kolaylık açısından derin öğrenme tabanlı çalışmalar son zamanlarda baskın hale gelmiştir. Bu durum, veri kümelerinin çoğalmasını ve yeni derin mimarilerin geliştirilmesini sağlamıştır. Önceki çalışmalarda en çok tercih edilen veri setleri EuRoC (Burri ve ark., 2016), KITTI (Geiger ve ark., 2013), Cityscapes (Cordts ve ark., 2016) , TUM (Sturm ve ark., 2012)'dir. Tüm bunlar dikkate alınarak, gelecekte derin öğrenme tabanlı çözümlerin yeni nesil otonom uygulamaların gelişmesini sağlayacağı yorumu yapılabilir.

Yukarıda anlatılanlara benzer şekilde, geleneksel yol planlama çözümleri için de geometrik modelleme, gerçek dünyanın modellenmesi, algı eksikliği gibi problemler mevcuttur. Ayrıca geleneksel yöntemleri geliştirerek hızlı yol planlama sağlamak için yeni yöntemler önerilse de hala arzu edilen başarı sağlanamamıştır. Çünkü hesaplama hızındaki bazı gelişmelere rağmen, hala boyutun artmasına bağlı olarak hesap süresi katlanarak artmaktadır (Qureshi ve ark., 2021). Bu problem yapay zekâ tabanlı çözümlerle giderilebilir, ancak yapay zekâ tabanlı mevcut çözümler henüz yol planlama için çok yenidir. Ancak elde edilen sonuçlar göstermektedir ki gelecek yol planlama algoritmalarının geliştirilmesinde yapay zekâ çözümleri büyük rol alacaktır. Yapay zekânın, diğer algoritmalara göre en önemli katkısı algıdır. Algı sayesinde karmaşık şekildeki statik ve dinamik engeller kolaylıkla ayırt edilebilecek ve ayrıca engeller için karmaşık bir geometrik modelleme gereksinimi kalmayacaktır. Kısaca insansı bir bakış açısı ile gelecekte yol planlamanın daha kolay hale geleceği öngörülmektedir.

Bu tez çalışmasında İHA'lar için iç ortamda otonom uygulamaların geliştirilmesine katkı sağlayabilecek üç adet uygulama geliştirilmiştir. Her uygulama

geleneksel yöntemlerden farklı olarak modern derin öğrenme yöntemlerini otonom İHA problemine entegre etmiştir. İlk uygulama önerdiği derin mimari ve atalet - görsel bilgilerin işlenmesi açısından yenilik taşımaktadır. İkinci uygulama ile numerik değerler içeren IMU verilerinden daha fazla yararlı özellik çıkarmayı içeren yeni bir teknik tanıtılmıştır. Son uygulamada ise İHA yol planlama uygulamaları için büyük katkı sağlayacak yeni bir yol planlama algoritması önerilmiştir. Yol planlama uygulaması hem yeni bir yöntem içermesi, hem optimizasyon içermesi hem de derin öğrenmeyi yol planlamaya dahil etmesi açısından önemli katkılar içermektedir.

3. MATERYAL VE YÖNTEM

Bu bölümde, kapalı ortamlarda İHA uygulamalarının geliştirilmesine yönelik bilinmesi gerekli olan yöntemler açıklanmaktadır. Bu kapsamda, otonominin mobil robotik ve İHA için önemi, gereksinimler, V(I)SLAM- V(I)O hakkında detaylı bilgiler ve son olarak yol planlama bu bölümde detaylı biçimde ele alınmıştır.

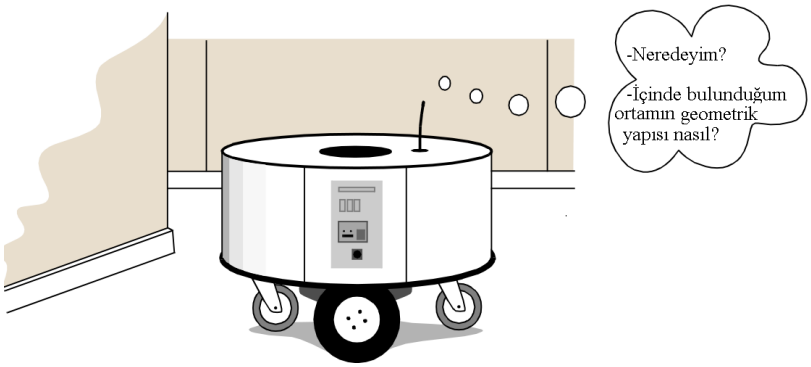
3.1. Mobil Robotlar için Otonomi

Robot, önceden programlanmış veya otonom görevleri yerine getirebilen elektro-mekanik bir cihazdır. Önceden programlanmış görevleri yerine getiren robotlar, genelde endüstriyel ürün üretiminde kullanılmaktadır. Günümüzdeki endüstriyel nitelikteki ürünlerin hızlı olarak üretilmesi, endüstriyel otomasyondaki hızlı değişim ve gelişim sayesindedir. Endüstriyel otomasyon ise bu gelişme sürecini Bilgisayar Destekli Tasarım (Computer-Aided Design (CAD)) sistemleri ve Bilgisayar Destekli Üretim (Computer-Aided Manufacturing (CAM)) sistemlerine borçludur. Bu sayede ortaya çıkan endüstriyel robotlar seri üretim için yüksek hız ve yüksek doğruluk sağlayarak dünya ekonomisinde belirleyici bir rol üstlenmiştir. Endüstri 4.0 devrimi ile birlikte, gelecekte endüstriyel robotlar gittikçe daha fazla önem kazanacaktır. Çünkü artan nüfus daha fazla ihtiyaç doğurmakta, bu da daha hızlı üretim demektir. Ayrıca, çalıştırılan bir insan, artık daha fazla maliyete sebep olmakta ve belirli riskler altında çalışmaktadır. Buna karşın teknolojinin ilerlemesi ile robot maliyetleri gittikçe düşmekte, robot tarafından yapılan işlerin verimi artmakta ve iş kazası gibi istenmeyen durumlar oluşmamaktadır. Bu tür robotların çalışmasında mekanik aksam daha çok ön plandadır. Çünkü bu robotlar genelde uygulama alanına özel tasarlanır ve sadece belirli görevler için yazılım gereklidir. Endüstriyel robotlara örnek olarak, kaynak robotu, boyama robotu, birleştirme (assembly) robotu, paketleme robotu gibi manipulatörler örnek gösterilebilir. Bu robotlar bir noktada sabittir ve üç veya daha fazla eksen içerir. Yani hareket kabiliyetleri sınırlıdır. Dolayısıyla endüstriyel robotlar genelde mobil ve otonom özelliğini taşımaz. Ancak farklı uygulama alanları ve farklı gereksinimler için, otonom ve mobil bir robot artık bir ihtiyaç haline gelmiştir (Craig, 2009; Siegwart ve ark., 2011).

Yazılım teknolojisindeki ilerlemeler ve işlemci hızlarının gittikçe artması mobil otonom robotik uygulamalarını hızlandırmıştır. Son yıllarda, fabrikalarda ve montaj hatlarındaki endüstriyel amaçlı kullanılan robotlardan farklı olarak, robotlar evlerde ve ofislerde de kullanılmaya başlamıştır. Hizmet ya da servis robotları olarak adlandırılan bu robotlar, çalışma alanlarını insanlarla paylaşırken yapılandırılmamış, yani önceden

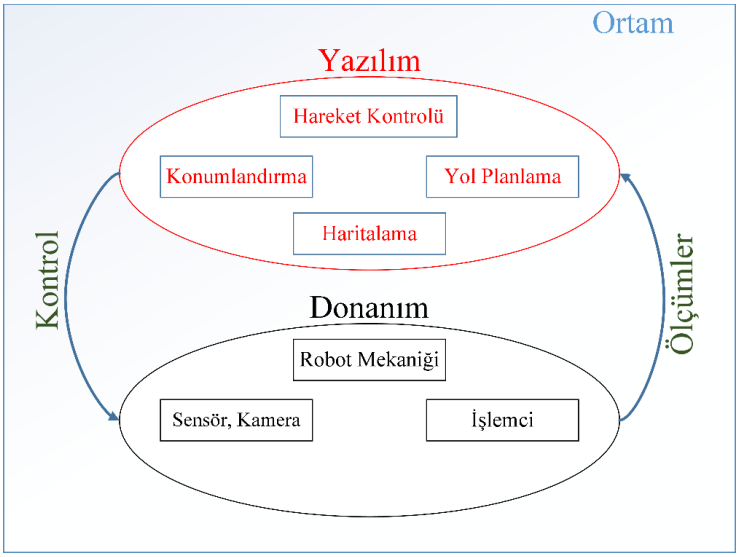
bilinmeyen bir ortamda bağımsız olarak çalışabilecek nitelikte olmalıdırlar. Tabi bağımsız hareketin en temel şartı mobil hareket yeteneğidir. Buna ilişkin tasarımlar ve çalışmalar için genelde insan ve hayvanlardan esinlenilir. Robota kazandırılan mobil kabiliyet sayesinde robot bulunduğu konumu değiştirebilir. Bunu sağlamak için tekerlekli robotlar (Olgun ve ark., 2016), hava robotları (Ruggiero ve ark., 2018), ayaklı robotlar (Hutter ve ark., 2017), yüzen robotlar (Tang ve ark., 2017) gibi çalışmalar şimdiye kadar birçok kez uygulanmıştır.

Kısaca tanımlamak gerekirse, mobil robot, genellikle, sensörlerle donatılmış, ortam boyunca hareket eden bir araçtır. Bu hareket ayaklar, tekerlekler veya kanatlar ile sağlanabilir. Mobil robotikte, hem uygulamasının daha kolay olması hem de uygulama alanının daha geniş olması sebebiyle en çok tekerlekli robotlar tercih edilir. Ancak günümüzde İHA'lar da oldukça ilgi görmeye başlamıştır. Genel olarak tekerlekli veya ayaklı olan arazi robotlarında, insan tarafından yapılan mobil aktivitelerin veya görevlerin robot tarafından yapılması hedeflenir. Bu sebeple, olabildiğince insana benzeyen ya da insan gibi düşünüp, karar verebilen insan odaklı çalışmalar, oldukça aktif bir araştırma alanı olmuştur. İnsan tamamen bağımsız bir şekilde karar verebilir ve bunun sonucunda içinde bulunduğu ortamı tanıyarak bir eylem gerçekleştirebilir. Ayrıca insan, karar aşamasında sadece önceki bilgilerini kullanmaz, ayrıca yeni bilgileri öğrenmeye eğilimlidir. Bundan hareketle gözetim (Di Paola ve ark., 2010), müze rehberi (Sasaki ve Nitta, 2017), alışveriş asistanı (Aslan ve ark., 2017), ev güvenliği (Kim ve ark., 2006), taşımacılık (Palunko ve ark., 2012) vb. görevleri gerçekleştiren mobil robotlar tasarlanmıştır (Rubio ve ark., 2019). Ancak bu çalışmaların genelinde arzulanan hedef, sistemin tamamen otonom çalışmasıdır. Yani robot bulunduğu ortamı tanımalı, buna göre kendi konumunu bulmalı ve son olarak da kendisine verilen görevi gerçekleştirmelidir. Dolayısıyla bir mobil otonom robot, "Neredeyim?" (Konumlandırma) ve "İçinde bulunduğum ortamın geometrik yapısı nasıl?" (Ortamın sınırlarını öğrenme (Haritalama)) sorularını cevaplandırabilmelidir (bkz. Şekil 3.1) (Siegwart ve ark., 2011).



Şekil 3.1. Bilinmeyen bir ortamda hareket eden robot (Siegwart ve ark., 2011)

Şekil 3.1'deki soruların çözüme kavuşması için, mobil robot donanımsal ve yazılımsal bileşenler içermektedir. Şekil 3.2, yapılandırılmamış bir ortamda otonom hareket edebilen herhangi bir robotun bileşenlerini göstermektedir. Şekil 3.2'deki gibi tasarlanmış bir robottan beklenen şey başarılı bir navigasyondur. Navigasyonda, robotun belirli bir noktaya ulaşmak için bir dizi kararlar (örneğin sola dön, sonra sağa dön gibi) alması beklenir. Bunun için robotun ortamdaki faydalı bilgiyi algılaması gereklidir. Başarılı algılama sayesinde robot ortamın yapısını öğrenir ve buna göre kendini konumlandırır. Ancak bu algılama sadece bir kez değil, görev tamamlanana kadar, çok defa tekrar eder. Algılama için çeşitli sensörler ya da kameralar kullanılabilir. Algılanan bilgiye göre robotun hareketi için, robotun mekanik yapısına (motorlar, sürücüler, vb.) bilgi gönderilmelidir. Bu işlem için de bir mikro denetleyici kullanılabilir. Başarılı bir navigasyon için optimum yol seçilmelidir. Bunun için de yol planlaması şarttır. Zaten iyi bir planlama, iyi bir lokalizasyon ve haritalamanın sonucudur. Bunun için de başarılı bir algılama şarttır.



Şekil 3.2. Bilinmeyen bir ortamda otonom bir mobil robot için gerekli bileşenler

Şekil 3.1'deki problem çok uzun bir süredir araştırma konusudur. Ancak gerekliliği ve zorluğu nedeniyle hala aktif bir araştırma konusudur (Stachniss ve ark., 2016). Zor olmasının sebebi, hakkında herhangi bir önbilgi bulunmayan bir ortamda, robotun ortam haritasına göre kendini konumlandırmasıdır. Ayrıca haritalama ve lokalizasyon arasında sıkı bir ilişki vardır. Çünkü sınır ya da köşe noktaları ortamın haritası hakkında bilgi verirken, aynı zamanda robotun içinde bulunduğu ortama göre konumunu da belirtir. Aslında her iki problemin de ayrı olarak ele alınması karmaşıklığı gidererek, zorluğu oldukça azaltır. Çünkü bu durumda bir diğer problemin bilindiği varsayılır. Ancak bilinmeyen bir ortam için, robot hareket ettikçe, robot tarafından yapılan algılamalar sonucunda konumlandırma hesabı harita bilgisine göre iteratif bir şekilde oluşturulmalıdır. Lokalizasyon ve haritalamanın birbiri ile sıkı ilişkili olması ve eş zamanlı olarak her iki bilgiye de ihtiyacın olması sebebiyle, bu problem literatürde SLAM olarak ayrı bir başlık altında irdelenmiştir.

3.2. İnsansız Hava Araçları için Otonomi

Mobil robotik alanında hem uygulamasının daha kolay olması hem de uygulama alanının geniş olması sebebiyle, otonom uygulamalar için şimdiye kadar en çok tekerlekli robotlar tercih edilmiştir. Ancak, insan kaynağına ihtiyaç duymadan zorlu ortamlara girmesi ve ekonomik olması nedeniyle navigasyon alanında İHA'lar artık daha çok talep edilmektedir. Özellikle arama ve kurtarma görevleri robotik sistemler için özel gereksinimlere sebep olur. Küçük ve kullanışlı bir İHA, durum değerlendirmesi ve gözetim uygulamalarında insan güçlerine temel destek sağlar. Şimdiye kadarki İHA uygulamaları umut verici olsa da, bu alan nispeten yenidir ve daha az keşfedilmiştir. İstikrarlı ve güvenilir uygulamaların geliştirilmesi için İHA'ların etkin kullanımı yapılmadan önce çözülmesi gereken birçok sorun vardır. Bu sorunlardan en önemlisi hassas konumlandırmadır (Gupta ve ark., 2016).

İHA'lar ile yapılması gereken görevlerin (örn. arama kurtarma) büyük bir kısmı, iç ortamlarda hassas bir konumlandırmayı gerektirir. Dolayısıyla İHA'lar GPS gibi bileşenler haricinde konumlandırma sağlamalıdır. Ayrıca GPS sinyalleri hava faktörlerine, elektromanyetik gürültülere, yüksek ve yoğun binalara bağlı olarak kolaylıkla bozulabilir. İHA ile yapılan görevlerin büyük bir kısmı ortam haritası, konumlandırma ve takip gibi görevler içerdiğinden İHA, bütün bu görevleri otonom bir şekilde yapabilmelidir. Özellikle bilinmeyen bir ortamda gezinen ve veri toplayan bir İHA için SLAM ve odometri çözümleri, robotik alanında önemli bir yer tutmaktadır.

Dolayısıyla çoğu uygulama İHA'ların, içinde gezindikleri ortamı anlamak ve görevlerini verimli bir şekilde tamamlamak için tamamen yerleşik sensörlere güvenmelerini gerektirir. Birçok araştırmacı, otonom navigasyon, gerçek zamanlı yol planlama ve nesne tanımaya dayalı keşif, teslimat veya gözetleme gibi sivil veya askeri görevler için ideal bir platform olarak İHA'lar üzerinde çalışmaktadır (Choi ve Cha, 2019).

Otonom navigasyon problemine sahip bir İHA, genel olarak, insan etkileşimi olmaksızın çevre ile çarpışmalardan kaçınırken bir hedef konuma ulaşma yeteneğine sahip olmalıdır. Hasar veya yaralanmalardan kaçınmak için güvenli ve doğru bir navigasyon sağlamak çok önemlidir. Ancak güvenli navigasyon sağlamak için İHA mevcut sınırlamaları dikkate alınmalıdır. İHA'larla ilgili mevcut teknolojilerdeki sınırlamalar, insansız kara araçları ve otonom sualtı araçları ile karşılaştırıldığında, otonom navigasyon yöntemlerinin geliştirilmesi çalışmalarına güvenilirlik ve sağlamlık açısından daha fazla zorluk katmaktadır. Algılama yetenekleri, izin verilen yük kapasitesi, uçuş süresi, enerji tüketimi, iletişim ve kontrolü ile ilgili sınırlamalar bunlara örnektir. Etkili ve gelişmiş bir otonom özellik sağlayacak hareket kontrol yöntemlerinin uygulanması, bu sınırlamalar göz önüne alınarak yapıldığı sürece geliştirilebilir. İHA'lara otonom yetenekler kazandırmak için SLAM tabanlı çözümler uygulanmalıdır. Sahip olduğu sınırlamalar nedeniyle, hafiflik ve maliyet açısından kamera kullanan VSLAM yöntemleri İHA'lar için daha elverişlidir (Elmokadem ve Savkin, 2021).

3.3. SLAM

Bir robotun konumlandırılması ve ortam hakkında bilgi edinilmesi için çeşitli yöntemler daha önce kullanılmıştır. Konumlandırmanın en basit olarak yapıldığı ilk uygulamalarda, kat edilen mesafeyi belirlemek için tekerlek enkoderleri kullanılmıştır. Ancak özellikle engebeli arazi veya kaygan zeminlerde, tekerlek kayması sebebiyle tekerlek odometrisinden elde edilen pozisyon tahmini hızlı bir şekilde sapmakta ve bu tahmin birkaç metre sonra kullanılamaz hale gelmektedir. Bu dezavantajlardan ötürü, IMU, LASER odometri, GPS, VO ve SLAM tabanlı diğer konumlandırma stratejileri önerilmiştir. Bunlar arasında VO ve SLAM yöntemleri sağladığı avantajlar sebebiyle son zamanlarda daha sık uygulanmaktadır (Scaramuzza ve Fraundorfer, 2011; Yousif ve ark., 2015; Cadena ve ark., 2016).

SLAM, bir mobil robotun bilinmeyen bir ortamda kendisini konumlandırmasını ve aynı anda üzerindeki sensör(ler) yardımıyla herhangi bir ön bilgi olmadan bu ortamın bir haritasını oluşturmasını gerektiren bir süreçtir. Hassas konumlandırmaya yönelik

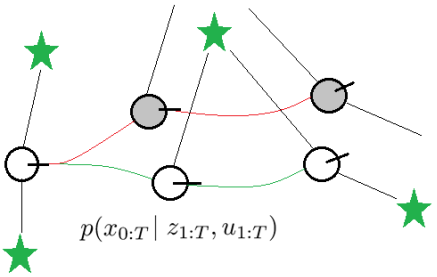
SLAM, çok sayıda uygulama alanı içerir. GPS bilgisinin yetersiz olduğu ortamlarda, konumlandırma bu yöntemler kullanılarak gerçekleştirilir. SLAM, iç – dış ortamlarda, deniz altında (Trabes ve Jordan, 2017), uzayda (Chen ve ark., 2017), yer altı (Ren ve ark., 2019) ile ilgili çalışmalarda ve artırılmış gerçeklik uygulamalarında (Piao ve Kim, 2017) sıklıkla tercih edilir. Ayrıca, mobil robotlar kullanılarak gerçekleştirilen otonom iç mekân haritalama, alan şartları veya güvenlik nedenlerinden dolayı insan erişiminin kısıtlanabileceği yerlerde gerekli bir araçtır. Uygulama alanları dikkate alındığında, otonom mobil robotlar hassas konumlandırmaya oldukça fazla ihtiyaç duyar. Ayrıca günümüzde gittikçe önemli hale gelen servis robotları ve İHA'lar için gerekli olan otonom görevleri sağlamak için güçlü bir SLAM algoritmalarına ihtiyaç vardır (Zhang ve ark., 2016; Trujillo ve ark., 2018).

3.3.1. SLAM Probleminin Tanımı

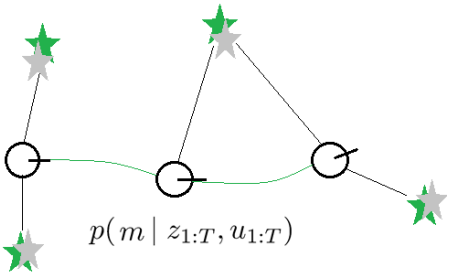
Lokalizasyon ve haritalama geçmişte ayrı problemler olarak çalışılmıştır. Fakat bir harita ancak robotun pozu bilindiğinde oluşturulabilir. Benzer şekilde, konumlandırma yapabilmek için doğru bir kesin bir harita bilgisine ihtiyaç duyulur. Ancak otonom mobil robot uygulamalarında çoğunlukla her iki problemle de aynı anda baş etmek gerekir. Dolayısıyla bu iki problem, her ikisinin de bilinmediği ama her ikisine de ihtiyaç duyulduğu durumlarda birleştirilir. Sonuç olarak ortada iki adet bilinmeyen bir durum vardır ve birindeki hata diğerini etkiler. Üstelik bu hata bir sonraki durumu da etkiler. Bu durum SLAM problemini zorlaştırır (Strasdat ve ark., 2012). Şekil 3.3 konumlandırma, haritalama ve bu ikisinin birleşiminden oluşan SLAM problemini göstermektedir. Şekil 3.3'ün iyi anlaşılabilmesi için bazı terimlerin anlamı bilinmelidir. Bu terimler aşağıdaki gibi özetlenebilir.

- İşaretçiler (Landmarks): Ortamda bulunan, kolayca gözlenebilen ve çevreden ayırt edilebilen özelliklerdir.
- Durum Tahmini: Durum, uygulama ile ilgili ölçülmesi gereken bir büyüklüğü temsil eder. Bu tez çalışması baz alınırsa, robotun pozisyonu veya bir işaretçinin pozisyonu, durum olarak değerlendirilebilir. Amaç bu durumlarla alakalı bir tahmin üretmektir. Aslında genelde durumlar çeşitli sensörler kullanılarak ölçülebilir. Ancak mükemmel olarak ölçülemediği için, durum tahmini yapılır. Çünkü sensörden ölçülen veriler daima gürültü içerir. Bu gürültü çok küçük de olsa zamanla kümülatif bir şekilde birikerek, çok büyük hatalar üretir. Ardışık olarak alınan ölçümlerin belirsizliği veya gürültüsü giderilmediği müddetçe, alınan

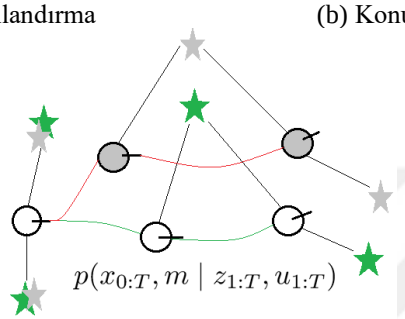
ölçümler gittikçe gerçek değerden uzaklaşır. Dolayısıyla her bir ölçümden sonra, belirsizlik dikkate alınmalı ya da ölçümler sürekli optimize edilmelidir.



(a) Harita tabanlı konumlandırma



(b) Konum tabanlı haritalama



(c) Haritalama ve konumlandırma (SLAM)

Şekil 3.3. SLAM, konumlandırma ve haritalama problemlerinin yapısı (Durrant-Whyte ve Bailey, 2006)

Şekil 3.3'te robotun belirli zaman aralıklarındaki hareketi ve algılamaları gösterilmektedir. Yıldızlar ortamdaki işaretçileri temsil eder. Yani robot ortamda bulunan bu işaretçileri tanıyabilir ve buna göre kendi konumu hakkında bir tahmin yapabilir. Yapılan tahminler gri renkli olarak gösterilmiştir. Buna göre Şekil 3.3 (a)'da amaç robotun veya sensörün hangi konumda olduğunun tespit edilmesidir. Bir durum tahmin problemidir. Konumlandırmada yapılan çevrenin haritasının tam olarak bilindiği varsayılır. Çünkü bir ortamdaki robot, haritaya göre konumlandırılabilir. Şekil 3.3 (b)'de konum tabanlı haritalama gerçekleştirilir. Mobil robot üzerindeki sensörler aracılığıyla alınan veriler kullanılarak ortamın modeli tahmin edilir. Gürültülü ölçümler nedeniyle, haritalama da SLAM için bir durum tahmini problemidir. Sadece haritalama yapılan bir uygulamada, lokalizasyonun çok iyi bilindiği varsayılır. Örneğin konum bilgisi çok iyi bilinen bir robot, Lazer menzil bulucu (laser range finder) kullanarak yakın nesnelerin kendi konumundan uzaklığını ölçebilir. Şekil 3.3 (c) ise, Şekil 3.3 (a) ve Şekil 3.3 (b)'den farklı olarak, ön bilgi olarak harita ve konum bilgisi içermez. Dolayısıyla her iki bilgiden de yoksun olan bir uygulamada bu iki durum tahmini de eş zamanlı olarak hesaplanmalıdır.

Şekil 3.3 (a)'da robot hareket halinde iken kendini konumlandırmaya çalışır. Fakat ikinci durumda robot gerçekte sağa doğru ilerlemiş olmasına rağmen, robot sola

doğru ilerlediğini düşünmüştür. Bu hatanın sebebi teker veya zemin şartları olabilir. Daha sonra, robotla yapılan gözlem sonucunda robotun işaretçiden belirli bir mesafe uzakta olduğu ölçülür. Ancak hatalı olarak ölçülmüş robot konumu dikkate alındığında, bu mesafede bir işaretçiye rastlanmaz. Bu noktada robotun konumunun yanlış ölçüldüğü anlaşılarak, işaretçinin konumuna göre bir düzeltme yapılır. Yani harita (işaretçi konumları) bilindiği için, yanlış olan ölçüm düzeltilerek daha doğru bir lokalizasyon tahmini yapılabilir. Bu durumun tam tersi de geçerlidir. Yani, Şekil 3.3 (b)'de gösterilen haritalama probleminde robotun konumu iyi bilindiği için, hatalı olarak ölçülen işaretçi konumları, robot konumuna göre düzeltilir. Buradan anlaşılacağı üzere konumlandırma ve haritalama birbiri ile sıkı ilişkilidir. Bu sebeple, ölçümdeki hata, o ölçümle ilişkili bilinen bir değerle düzeltilebilir. Ancak SLAM'de her iki problem de bilinmemektedir ve bir değerin tahmini için diğerine ihtiyaç duyulur. Bu sebeple Şekil 3.3 (c)'de görüldüğü üzere, SLAM problemi daha zordur ve tahminler daha hatalıdır.

3.3.2. SLAM Probleminin Olasılıksal Temsili

İlk önerildiği günden bugüne kadar SLAM için çok sayıda yöntem önerilmiştir. Önemli gelişmelere rağmen hala büyük zorluklar vardır, dolayısıyla geliştirilmeye ihtiyaç duymaktadır (Stachniss ve ark., 2016). Şimdiye kadar önerilen geleneksel SLAM, V(I)SLAM, V(I)O, derin öğrenme tabanlı SLAM gibi çok sayıdaki farklı yöntemi açıklayan bir çalışma Huang ve ark. (2019) tarafından sunulmuştur.

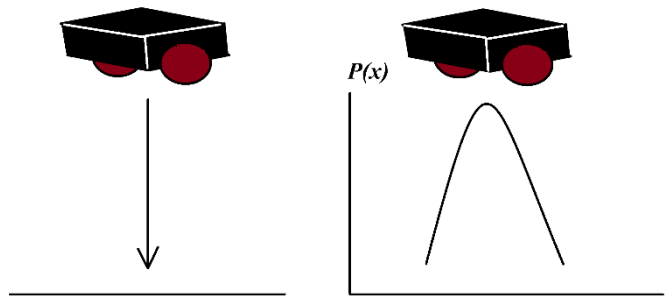
Tablo 3.1. SLAM ve Odometri problemi için verilen ve istenen değerler

Verilen	İstenen
Robotun kontrol komutları ➢ $\boldsymbol{u_{1:T} = \{u_1, u_2, u_3, \dots, u_T\}}$	Ortamın haritası ➢ $\boldsymbol{m}$
Sensörlerle algılanan gözlemler ➢ $\boldsymbol{z_{1:T} = \{z_1, z_2, z_3, \dots, z_T\}}$	Robotun konumu ➢ $\boldsymbol{x_{0:T} = \{x_0, x_1, x_2, \dots, x_T\}}$

Şekil 3.3 aracılığıyla, anlatılan temel bilgiler baz alınarak, SLAM problemi ile ilgili verilenler ve istenenler Tablo 3.1'de gösterilmiştir. Tablo 1 de bulunan $\boldsymbol{u}$, kontrol komutunu temsil eder. Kontrol komutları, robota gönderilen yönelim bilgileridir. Örneğin $\boldsymbol{u_t}$, "bir metre ileri git", "sağa dön", "dur" gibi komutlar olabilir. Bu komutlar robota uzaktan gönderilebilir veya robottan alınan bilgi ile elde edilebilir. Genelde robottan alınan bilgi ile robotun hareketi hakkında bilgi alınır. Çünkü robota gönderilen bir komut, robot tarafından tam olarak gerçekleştirilemez. Örneğin robota uzaktan gönderilen "2 metre ilerle" komutu sonucunda robot 1.98 metre ilerleyebilir. Ancak tekerin devir

sayısını ölçen bir sistem, tekerin çevresine bağlı olarak robotun ne kadar ilerlediğini daha doğru hesaplayabilir. Bu sistem odometri olarak adlandırılır. Dolayısıyla odometri bilgisi, robota gönderilen komut olarak kullanılabilir. $\boldsymbol{z}$, robota bağlı sensörlerden alınan ölçümleri simgeler. Bu değer de doğrudan sensörler aracılığıyla ölçüldüğü için verilen bir değerdir. Örneğin bu $\boldsymbol{z}$ değerleri, en yakın engelin uzaklık bilgisini veren bir lazer sensörden elde edilebilir. Veya sensör yerine kullanılan bir kamera sonucunda, $\boldsymbol{z}$ değerleri görsel olarak hesapalanabilir. Sonuç olarak $\boldsymbol{z}$ değerleri işaretçilerin konumunun ölçülmesine yönelik değerleri ifade eder. $\boldsymbol{m}$ çevrenin haritasını temsil eder. Bu haritalar çevrede bulunan işaretçiler, nesneler, vb. objelere göre oluşturulur. Dolayısıyla $\boldsymbol{m}$ bu işaretçilerin veya nesnelerin konumlarını temsil eder. Bu konumların tamamı bulunursa, robotun konumuna göre ortamın bir haritası elde edilir. Son olarak $\boldsymbol{x}$ robotun belirli zaman aralıklarındaki koordinatlarını ifade eder. Robot bu $\boldsymbol{x}$ lokasyonlarına, $\boldsymbol{u}$ komutları ile gider. Bulunduğu konuma göre $\boldsymbol{z}$ ölçümlerini yaparak $\boldsymbol{m}$ değerlerini tahmin eder.

$\boldsymbol{m}$ ve $\boldsymbol{x}$ değerlerinin tahmin edilmesinin sebebi, daha önce de bahsedildiği gibi, $\boldsymbol{u}$ ve $\boldsymbol{z}$ değerlerinin belirsizlik veya gürültü içermesidir. Doğru bir tahmin bu belirsizliğin azaltılması veya yok edilmesi ile mümkündür. SLAM için temel problem olan bu bellirsizlik, çoğunlukla istatistiksel yaklaşımlarla temsil edilir. Şekil 3.4, bir robotun konumunun nasıl temsil edildiğini göstermektedir. Şekil 3.4(a)'da belirsizlik ihmal edilmiştir ve deterministik olarak robot konumu belirlenmiştir. Şekil 3.4(b) ise belirsizliği, olasılıksal olarak ifade etmektedir. Gerçek dünya ölçümleri baz alındığında, bütün sensörler az ya da çok hata içerir ve hiçbir sensör robotun nerede olduğunu mükemmel bir şekilde ölçemez. Sensör hatası ihmal edilebilecek kadar çok küçük olsa dahi, bu hata zamanla kümülatif olarak birikir. Dolayısıyla sensörden alınan bilgi tek bir değeri değil, bir değer aralığını kapsamalıdır. Bu kapsamda Şekil 3.4 (b), gerçek dünya uygulamalarında daha doğru bir temsil sağlar.



(a) Deterministik yaklaşım (b) Olasılıksal yaklaşım

Şekil 3.4. Mobil robot lokalizasyonu için olasılıksal ve determisitik yaklaşım

Konumlandırma veya haritalama amacıyla kullanılan çeşitli sensörler mevcut olmasına rağmen, SLAM problemini zor olmasına neden olan sensör ölçümlerindeki belirsizliktir. Özellikle mobil robotlar için ortam boyunca ardışık olarak yapılan her bir ölçümde, gürültüler sürekli üst üste biner ve zamanla çok yüksek hatalar meydana gelir. Bu sebeple, gerçek zamanlı, yapılandırılmamış bir çevrede bulunan robotik uygulamalarda, belirsizlik, çözülmesi gereken bir problemdir (Thrun ve Leonard, 2008). Bu problemin çözülmesi için matematiksel tanımı yapılmalıdır. Belirsizliğin uygun bir şekilde modellenmesi için genelde olasılıksal yaklaşımlar dikkate alınır. Bu sebeple, lokalizasyon ve haritalama gibi gerçek dünya problemleri için kullanılan robotik uygulamaların çoğu olasılıksal hesaplamalar içerir. SLAM problemi Tablo 3.1'e göre aşağıdaki gibi ifade edilebilir.

$$p(x_{0:T}, m \mid z_{1:T}, u_{1:T}) \tag{3.1}$$

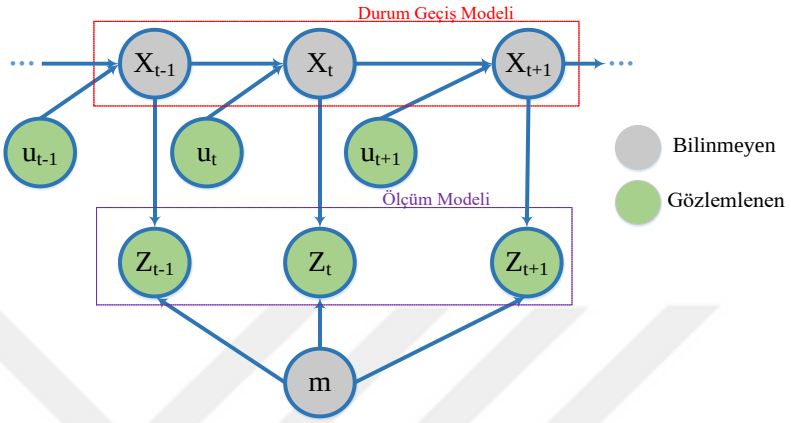
Olasılık Dağılımı — Robotun konumları — Harita noktalar — Şart — Gözlemler — Kontrol Komutları (Odometri bilgisi)

Denklem 3.1 SLAM probleminin olasılıksal olarak temsilidir. Aslında Tablo 3.1'in formüle edilmiş halidir. Bu eşitlik SLAM araştırmacılarına şöyle bir soru yöneltir: "Gözlem ve kontrol girdileri dikkate alındığında, robotun konumunun ve çevre haritasının olasılıksal dağılımı nedir?" Tüm olasılıksal tabanlı SLAM yöntemlerinin amacı bu denklemi verimli bir şekilde çözebilmektir.

SLAM uygulamalarında, robot önceden bilinmeyen bir ortamda bilinmeyen bir konumda göreve başlar. Robotun dış ortamı algılayabilmesi için kamera, lazer, kızıl ötesi, GPS, vb. sensörler kullanılır. Robot, işaretçileri gözlemlemek için üzerindeki sensörlerini kullanır ve bu bilgilerden, yer işaretlerinin yerlerini tahmin ederken aynı anda kendi pozisyonunu hesaplar. Bunun için robot, ortamda kontrol girdileri (veya odometri sensörü) ile gezinmelidir. Robot çevreye doğru ilerledikçe, robotun değişen gözlemsel bakış açısı, robotun mevcut konumunu ilk konumuna göre takip etmek için sürekli olarak kullanılan tam bir işaretçi haritasının artımlı bir şekilde oluşturulmasını sağlar (Spero ve Jarvis, 2007). Eş zamanlı olarak hem harita hem de konumun tespit edilebilmesi bir kestirim problemidir. Dolayısıyla uzun yıllar boyunca, Bayes filtresine bağlı yöntemler SLAM literatürüne hakim olmuştur.

3.3.3. SLAM Probleminin Grafiksel Modeli

Genellikle durumlar arası ilişkiler matematiksel gösterimin yanında grafiksel modelleme ile de gösterilir. SLAM için özellikle grafiksel model yaygın şekilde kullanılır. Ölçülen veya tahmin edilen durumlar arasında zaman bakımından bir ilişki olduğunda, grafiksel modelleme sistemin genel yapısını daha iyi izah eder.



Şekil 3.5. SLAM probleminin grafiksel modeli

Şekil 3.5 SLAM problemindeki ölçülen ve bilinmeyen durumlar arasındaki ilişkiyi gösterir. Her bir düğüm durumları, durumlar arası oklar da etkileşimi gösterir. Örneğin, bir önceki robotun pozu X_{t-1} ve şuanda gerçekleştirilen kontrol girdisi ya da odometri bilgisi u_t, robotun yeni pozu X_t'yi doğrudan etkiler. Bu sebeple eğer robotun, mevcut konumu ve verilen komut biliniyorsa, robotun bir sonraki konumu başarılı bir şekilde tahmin edilebilir. Buna hareket modeli ya da durum geçiş matrisi denir. Ayrıca Şekil 3.5 incelendiğinde, mevcut gözlem Z_t, mevcut durum X_t'ye ve harita noktası m'ye bağlıdır. Yani eğer robotun şuanki durumu ve karşılık gelen mevcut harita noktası biliniyorsa, mevcut ölçüm bilgisi başarılı olarak tahmin edilebilir. Bu da sensör modeli ya da ölçüm modeli olarak adlandırılır (Thrun ve Leonard, 2008).

3.3.4. Tam vs. Çevrimiçi SLAM (Full vs. Online SLAM)

SLAM'de esas olan robotun yolunu tahmin etmektir. Ancak bu iki şekilde de mümkündür: Tam SLAM vs. Çevrimiçi SLAM. Eğer mevcut ve geçmişteki tüm pozlar ile ilgileniliyorsa, bu Tam SLAM'dir. Yani robotun tüm yolu tahmin edilir. Bu durum Denklem 3.2'de gösterilmiştir.

$$\boldsymbol{Tam\ SLAM} \rightarrow \boldsymbol{p(x_{0:T}, m \mid z_{1:T}, u_{1:T})} \tag{3.2}$$

$$\boldsymbol{Çevrimiçi\ SLAM} \rightarrow \boldsymbol{p(x_t, m \mid z_{1:T}, u_{1:T})} \tag{3.3}$$

Eğer sadece mevcut poz tahmin edilmeye çalışılıyorsa ve geçmişteki pozlarla ilgilenilmiyorsa, Çevrimiçi SLAM gerçekleştirilmiş olur. Denklem 3.3'te gösterildiği gibi, robotun sadece mevcut konumu tahmin edilmeye çalışılır. Robotik SLAM uygulamaları genelde gerçek zamanlı uygulamalarda kullanıldığından Çevrimiçi SLAM daha sık kullanılır.

3.3.5. SLAM Problemi için İlk Çözümler

Tanıtılan SLAM problemlerini çözmek için araştırmacılar ilk olarak üç farklı yöntem kullanmışlardır. Bunlardan ilk ikisi Bayes teoremine dayanır. Bayes teoremi durum tahminine yönelik çözüm üretir. Kesinlik içermeyen bilgileri, gözlemler ve önsel bilgi kullanarak tahmin eder. Bayes yöntemi kullanılarak gürültülü veriden en iyi tahmin değerinin bulunması işlemleri gerçekleştirildiği için, bu yöntem aslında bir filtreleme işlemidir. SLAM de bir durum tahmin problemi olduğundan, bu problemin ilk çözümü için, Bayes filtresine dayanan Kalman Filtresi (KF) ve Parçacık Filtresi (PF) tabanlı yöntemler kullanılmıştır (Cadena ve ark., 2016). Ancak mobil robot konumu ve harita bilgisi doğrusal ve Gaussian çözümlerle sınırlı olmadığından KF yetersiz kalır. Dolayısıyla, SLAM sorununu ele almak için KF elverişsiz çözüm sağlar. Bu tür durumlar için, doğrusal olmayan durumlara karşı da çözüm üretebilen KF tabanlı EKF yöntemi kullanılır. Bu bağlamda KF tabanlı EKF SLAM ve PF tabanlı FastSLAM yöntemleri ilk SLAM çözüm yöntemleridir. Filtre tabanlı SLAM yaklaşımları önceki durumları marjinalleştirdiklerinden, çevrimiçi SLAM sorunu genellikle filtreleme olarak adlandırılır. EKF hızlı bir çözüm sağlamasına rağmen, zamanla çözüm karmaşıklaşır, tutarsızlık ve doğrusallaştırma hataları çözümün doğruluğunu olumsuz etkiler. FastSLAM ile durum tahmini dağılımı, bir dizi parçacık ile temsil edilir. Her bir parçacık bir durum tahmini üretir. Çok sayıdaki tahmin, bir dizi parçacıklar kümesinde toplanarak, sonsal dağılıma yaklaşılır. Parçacık sayısının artması çözümü başarılı bir tahmine yaklaştırır. Ancak parçacık sayısının çok artması çok fazla hesap yapılmasına neden olur.

Üçüncü yöntem grafiksel temsillere dayanan ve SLAM problemleri çözümü için doğrusal olmayan optimizasyon yöntemlerini uygulayan grafik tabanlı SLAM (Graph-Based SLAM) yöntemleridir. Yani filtre tabanlı yöntemlerdeki gibi giriş verilerini filtrelemek yerine, matematiksel kısıtlamalar kullanarak mevcut tüm bilgileri içeren bir grafik oluşturur. Yer işaretleri ve robot konumları, bir grafikteki düğümler olarak düşünülür. Bu, algoritmanın tam SLAM problemini çözmesine, yani tüm yer işaretleri ve tüm araç pozları için bir tahmin bulmasına izin verir. Yalnızca en sonuncusu yerine,

tahminde geçmiş robot pozları baz alınarak yörünge oluşturulur. Aslında grafik- tabanlı SLAM yönteminde, durum uzayı daha büyük hale gelse bile, problem yapısı seyrek kalır ve problem düzgünleştirme (yumuşatma (smoothing)) ile çok verimli bir şekilde çözülebilir. Bu çözümü ilk uygulayan çalışma Lu ve Milios (1997) olmuştur. Robotun tüm yörüngesini tam ölçüm setinden tahmin ettiğinden dolayı, tam SLAM sorunu da yumuşatma olarak bilinir (Grisetti ve ark., 2010; Kaess ve ark., 2012).

Grafik tabanlı SLAM yöntemlerinde, güncelleme sırasında EKF SLAM yöntemlerindeki gibi kovaryans matrisine dayalı bir sınırlayıcı bir faktör yoktur. Grafiğin güncelleme süresi sabittir. Ayrıca EKF grafiğin seyrekliğinden yararlanarak EKF SLAM'dan çok daha yüksek boyutlu haritalar için uygulanabilir. Ancak sürekli olarak optimizasyon işlemi gerçek zamanlı uygulama için işlem maliyeti oluşturabilir. Optimizasyon sürecini basitleştirmek için, başlatma süreci de önemli bir adımdır. Öyle ki güçlü bir başlatma süreci optimizasyon süresini azaltabilir (Stachniss ve ark., 2016).

3.4. Görsel SLAM (VSLAM) ve Görsel Odometri (VO)

SLAM için yapılan çoğu çalışmada, sonar sensörler, LIDAR, IR sensörler ve lazer tarayıcılar kullanılarak çeşitli çözümler sunulmuştur (Bueno ve ark., 2016). Son zamanlarda, GPS, lazer, ultrasonik ve sonar sensörlere kıyasla, kameralardan ya da RGB-D sensörlerden elde edilen renk, doku ve diğer görsel unsurlar daha zengin görsel bilgiler içermesi nedeniyle görsel tabanlı VSLAM öne çıkmıştır. Ayrıca lazer, sonar, enkoder ve odometri cihazları ağırdır ve çalışma koşulları kameraya göre daha sınırlıdır. Kameralar ağırlık, maliyet ve güç tüketimi açısından diğer sensörlere nazaran daha avantajlıdır. Sensör, sonar, lazer gibi aygıtlar küçük İHA'lar için, platformun tasarımı, mobilite ve taşıma kapasitesi ile ilgili önemli kısıtlamalara sebep olur. Ayrıca bu cihazların bir kısmı sınırlı bir çalışma aralığına sahiptir ve bazıları 2B haritalarla sınırlıdır. VSLAM, konum ve haritalama yapmak için yalnızca görsel girişleri kullanır. Yani gerekli olan tek sensör, cihazın üzerine monte edilmesi gereken bir kameradır. Başka harici sensör gerekmez.

VSLAM, geleneksel SLAM yöntemlerine avantaj sağlamasına rağmen, yapılan çözümlemeler hesaplama maliyeti, görüntü işleme ve özellik çıkarma gibi karmaşık algoritmaları beraberinde getirmiştir. Monoküler, stereo ve RGB-D kameralarla SLAM'ı uygulamak çok daha zengin ama karışık bir bilgi sağlayacaktır. Bu durumda veri ilişkilendirme ve hesaplama süresi gibi zorluklar devreye girecektir. İlk zamanlardan beri görsel bilgi kullanılarak SLAM uygulamaları geliştirmek bilinen bir yöntemdi. İlk önerilen yöntemler EKF tabanlıydı (Ayache ve Faugeras, 1988). Fakat yeterli hesap

gücüne sahip işlemcilerin bulunmaması VSLAM yöntemlerinin gelişimini engelledi. Bilgisayar teknolojisindeki gelişmeler sayesinde VSLAM çözümleri mümkün hale geldi. İlk çözümler derinlik bilgisi içeren RGB-D sensörlerle yapıldı (Davison, 1999; Se ve ark., 2002).

Maliyet ve kalibrasyon zorlukları göz önüne alındığında monoküler kamera sistemleri daha kullanışlıydı. Ancak bu durum daha zordu ve sadece yansıtma görevi gören bir kamera için de ölçek problemi mevcuttu. Yani ortamın geometrik bilgisi monoküler kamera ile elde edilemez. Dolayısıyla güncel monoküler kamera içeren çalışmalar ölçek tahmini, yani ilgili pikselin derinliğini, ardışık kareler arasındaki hareket ile elde etti. Çoğunlukla üçgenleme (triangulation) tekniği kullanılarak ardışık karelerde gözlenen bir özellik noktası ile bilinmeyen derinlik bilgisi elde edildi. Ayrıca, derinliği zamanla üçgenleme gerekliliği SLAM için önemli bir etkiye sahiptir. Ancak monoküler ölçümler ile elde edilen görsel özellik noktaları için üçgenleme tekniği uygulamak, gerçek zamanlı kısıtlamalar düşünüldüğünde büyük bir zorluktur. Davison ve ark. (2007) tarafından 2007 yılında tanıtılan MonoSLAM, ilk gerçek zamanlı monoküler VSLAM uygulamasıdır. EKF kullanan filtreleme tabanlı bir uygulamadır. Görsel öznitelik noktaları tespit edilir, hareket boyunca eşleştirilir ve ardından üçgenleme gerçekleştirilir. Kameranın 6 DoF hareketi ve özellik noktalarının 3B konumu, EKF için bir durum vektörü olarak temsil edilir. Ancak büyük bir ortamda, çok sayıda özellik noktası nedeniyle durum vektörünün boyutu artar ve hesaplama maliyeti yükselir.

VSLAM araştırması, fotoğraflardan geometrik bilgiler elde etmeye çalışan Fotogrametri (Photogrammetry) ve Hareketten Yapı (Structure From Motion (SFM)) bilimiyle yakından ilişkilidir. Her iki bilim dalında kamera ile çekilen fotoğrafları haritacılık alanına yönelik yorumlamak için, bu fotoğraflar üst üste bindirilir ve hizalanır. Çekilen fotoğraflarla üç boyutlu gerçek dünya yüzeyinin, üç boyutlu nokta bulutu oluşturulur. Bunun için fotoğraflar arasındaki özellik noktalarının arasındaki mesafe hesaplanır (Fraser, 2018). SFM ve Fotogrametri çalışmalarında kullanılan ortak yöntem genellikle Demet Ayarlamadır (Bundle Adjustment (BA)) (Triggs ve ark., 1999). BA, 3B modelin yansıması (reprojection) ile görüntüdeki ilişkili noktalar arasındaki mesafeyi en aza indirmeyi amaçlayan yinelemeli bir optimizasyon tekniğidir. Hem VSLAM hem de SFM yönteminde amaç büyük ölçüde benzerdir, yansıma hatalarını minimize etmektir. Temel fark VSLAM sistemlerin çevrimiçi çözümleme gereksinimleridir. Kamera kareleri ardışık olarak gelir ve her bir kare için konum ve harita güncellemesi yapılır. Ancak SFM yığın/toplu (batch) yaklaşımdır ve bir dizi görüntüyü kullanarak çevrimdışı olarak 3B bir

temsil oluşturur. Klein ve Murray (2007) BA yapısını gerçek zamanlı SLAM problemine dahil eden Parallel Tracking and Mapping (PTAM) çalışması ile, BA tabanlı SLAM çalışmalarının kapısını açtı. Ayrıca PTAM izleme ve eşleme iş parçacıklarını ayrı bir iş parçacığı olarak bölerek, haritalamanın izleme üzerindeki gecikmesini engelledi. Ayrıca işlemler kamera kareleri arasından seçilen ana kareler üzerinde uygulandı. Ana Kare seçimi ayrıca karmaşıklığı azaltma açısından önemliydi. Zaten PTAM çalışmasından sonra, ORB-SLAM ve LSD-SLAM dahil olmak üzere, çoğu popüler VSLAM algoritması bu tür çoklu iş parçacığı yöntemini ve ana kare uygulamasını benimsedi (Strasdat ve ark., 2010; Strasdat ve ark., 2012; Duan ve ark., 2019).

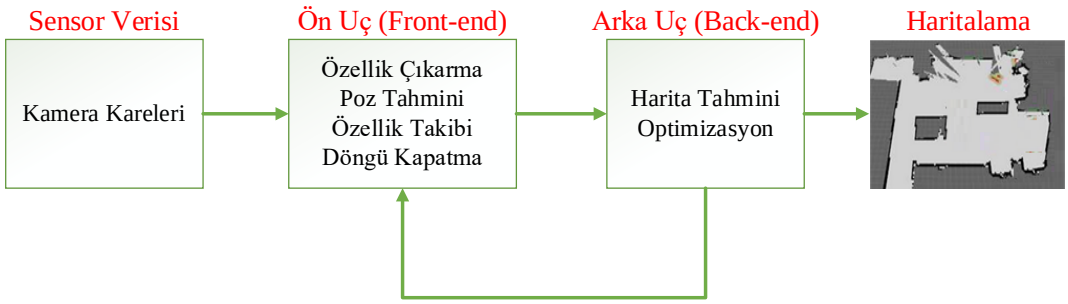
SLAM yöntemlerinin yanında VO çalışmaları da gelişen yöntemlerle araştırmacılar tarafından sıklıkla uygulanmaya başlamıştır. VO ortamının görüntülerini gözlemleyerek robotun hareketini, yönelimini referans kareye göre tahmin etme süreci olarak tanımlanır. VO maliyeti açısından Lazer, LIDAR, vb. kullanan yöntemlere göre daha avantajlıdır. VO yöntemlerinde robot pozisyonuna bağlı olarak robot yolu kademeli olarak tahmin edilir. Yalnızca robot yörüngesinin yerel (local) tutarlılığıyla ilgilenilir ve yerel yörüngenin daha doğru bir tahminini elde etmek için yerel harita kullanılır. SLAM ise robot yolunun küresel (global) ve tutarlı bir tahminini elde etmektir. Bu, robotun daha önce ziyaret edilen bir alana ne zaman döndüğünü anlaması için döngü kapatma (loop closing) adımlarını gerektirir. Bu sayede SLAM'de robotun daha önce haritalanmış bir alana tekrar geldiği tespit edilerek döngü kapatılabilir. Bu bilgi tahminlerdeki sapma miktarını azaltır. Döngü kapatma durumunun tespiti ile robot çevrenin gerçek topolojisini anlar ve konumlar arasındaki kısa yolları bulabilir (Scaramuzza ve Fraundorfer, 2011; Yousif ve ark., 2015; Cadena ve ark., 2016).

VO kameranın yörüngesini aşamalı olarak yeniden oluşturur, aslında uygulama olarak VO, VSLAM içindeki bir modül olarak görülür. Bu nedenle, bazı araştırmacılara göre VSLAM, VO'dan daha gelişmiş bir araştırma alanı olarak görülmektedir. Uygulama senaryoları açısından füze güdüm uçuşları, İHA ve artırılmış gerçeklik gibi gerçek zamanlı lokalizasyonun gerekli olduğu durumlarda VO yeterlidir. Konumlandırmaya ek olarak bir de eş zamanlı bir harita bilgisinin oluşturulması ek işlem gücü gerektirdiğinden, harita bilgisinin gerekli olmadığı uygulamalar için VSLAM uygulamak gereksiz olacaktır (He ve ark., 2020). Ayrıca döngü kapatma olasılığının olmaması durumunda (örneğin geniş ortamlarda), çözüm hızlı bir şekilde bozulabilir ve VO'ya göre üstünlüklerini kaybedebilir (Couturier ve Akhloufi, 2021). Bu nedenle sadece hassas konumlandırmaya ihtiyacın olduğu uygulamalarda, özellikle İHA'larda, VO için yapılan çalışmalar

çoğalmıştır (Mansur ve ark., 2017; Depaola ve ark., 2018; Varshosaz ve ark., 2020; Gurturk ve ark., 2021).

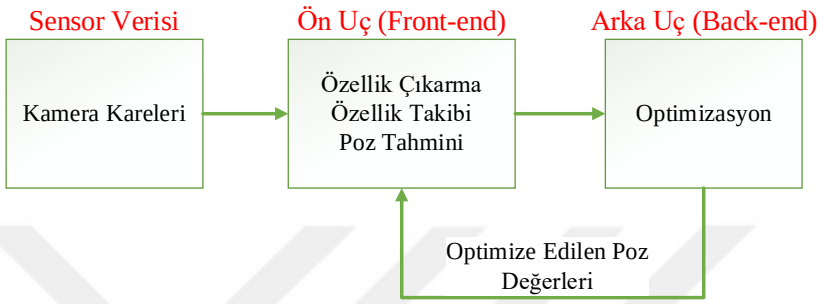
3.4.1. VSLAM ve VO yapısı

VSLAM ya da VO yapısı genel olarak ifade edildi. Ancak çok sayıda yöntem olduğundan, bu yöntemleri blok şema olarak göstemek konunun anlaşılması açısından yaygın olarak kullanılır. VSLAM temel olarak dört modül içerir: sensör verisi, ön uç (front-end), arka uç (back-end) ve haritalama (bkz. Şekil 3.6). Ön uç, ana kare (keyframe) noktaları algılamak, takip etmek, eşleştirmek için algoritmalar ve veri yapılarından oluşur. Tabi bu durum ana kare noktası tabanlı bir sistem olduğu varsayılırsa gerçerlidir, alternatif olarak bazı yeni çalışmalar kenar pikselleri veya hatta tüm pikselleri kullanmaktadır. Ön uçta kamera ile çekilen kareler (ölçüm bilgilerini) kullanılarak, ego-hareket (kamera veya sensör hareketi) tahmin edilir ve arka uç için iyi bir başlangıç değeri sağlanır. Döngü kapatma, görüntüler arasındaki benzerliği hesaplayarak kameranın daha önce aynı yerden geçip geçmediğini belirler. Eğer döngü kapanması tespit edilirse, arka uç için kısıtlama bilgisi sağlanır. Arka uç, haritanın optimizasyon problemini çözer. Optimizasyon modülü gürültülü verilerden yörünge ve harita durumunu tahmin eder. VSLAM, arka uçtaki optimizasyon yöntemine göre filtre tabanlı ya da optimizasyon (veya BA) tabanlı olabilir. Arka uç, ön uç ve döngü kapama tespiti tarafından sağlanan değere dayanarak global en uygun tahmini elde eder. Ön uç esas olarak robotun sensör verisi ile konumunu ve pozisyonunu tahmin etmek için kullanılır. Bununla birlikte, ölçüm bilgileri farklı derecelerde gürültü içerir. Bu gürültüler, poz tahmininde kümülatif hatalara yol açacak ve bu hatalar zamanla artacaktır. Filtreleme veya optimizasyon algoritmaları sayesinde arka uç, kümülatif hataları ortadan kaldırabilir ve konumlandırma ve harita doğruluğunu iyileştirebilir (Jiang ve ark., 2019; Li ve ark., 2019).



Şekil 3.6. VSLAM probleminin yapısı

VO için haritalama görevi olmadığından, Şekil 3.6'daki VSLAM yapısındaki haritalama modülü uygulamaya dahil edilmez. Optimizasyon işlemi sadece poz için yapılır (Poz Grafik Optimizasyonu (Pose-Graph Optimization) (Das ve Dubbelman, 2018)). Şekil 3.7 VO için blok şemasını gösterir. Ön uçta, ardışık kamera karelerindeki özellikler çıkarılır, bu özellikler arasında eşleşme gerçekleştirilerek, hareket ve poz tahmini sağlanır. Arka uç poz doğruluğunu artımak için poz değerlerini optimize eder.



Şekil 3.7. VO probleminin yapısı

SLAM probleminin tanıtımında işaretçiler konumlandırma ve haritalama için önemli bir ipucu sağlamaktaydı. Sensör verileri ile bu işaretçilerin mesafe bilgisi elde edilebiliyordu. Monoküler VSLAM ve VO uygulamalarında görsel işaretçilerin oluşturulması için görüntülerden özellik çıkarılır. Bunun için SIFT (Lowe, 2004), SURF (Bay ve ark., 2006), FAST (Rosten ve Drummond, 2006) ve ORB (Rublee ve ark., 2011) gibi farklı özellik tanımlayıcıları (feature descriptors) kullanılır. Hareket tahmini için bu ardışık kareler arasındaki özellikler eşleştirilir. Eşleşme oranı yüksek seviyede olan özellikler kullanılarak hareket tahmin edilir. Bunun için genellikle Rastgele Örnek Uyuşumu (RANdom Sample Consensus (RANSAC)) algoritması kullanılır. Sonraki optimizasyon süreci bu tahmini geliştirmeye yönelik iyileştirmeler yapar. VSLAM için döngü kapatma da zamanla kümülatif olarak artan pozisyon tahmini için küresel bir sınırlama sağlar.

3.4.2. Arka Uç Yaklaşımına Göre VSLAM ve VO

Geleneksel VSLAM uygulamalarında görüntüler ile yapılan tahminleri iyileştirmeye dayalı iki yaklaşım benimsenmiştir, bunlar filtreleme ve BA tabanlı metotlardır. Filtreleme tabanlı yaklaşımlar, geçmiş durumları sürekli marjinalleştirir. Yani mevcut poz dışındaki tüm pozlar her yeni kareden sonra marjinalleştirilir. Bu durum hesabın gereksiz ve fazla karışmasını önleyecektir, ancak bu durum zamanla tüm karelerin birbirine bağımlı olmasına neden olacaktır. Yeni özelliklerin eklenmesi durum

vektörünün boyutunu artıracaktır ve hesaplama gücü azalacaktır. Harita büyüdükçe filtre tabanlı yöntemler gerçek zaman için uyumsuz hale gelecektir (Munguia ve ark., 2022). Arka uçta BA tabanlı optimizasyon kullanan çalışmalar belirli sayıdaki geçmiş bazı pozları kullanarak mevcut tahmini iyileştirir. BA, özellik konumları, kamera konumları ve gerçek kamera parametrelerini kullanan doğrusal olmayan bir toplu (batch) optimizasyon yöntemidir. Optimizasyon tabanlı yöntemler BA'yı grafik büyüdükçe yeniden hesaplar, ancak hesaba geçmiş pozların tamamını değil, seçilen bir kısmını dahil eder. Bu seçim için belirli şartları sağlayan ana kareler belirlenir. Bu nedenle optimizasyon yaklaşımları ana kare tabanlı olarak da adlandırılabilir. Optimizasyon yöntemlerinde ana kareler haricindekiler tahmin için kullanılmaz, yani bir marjinalleştirme yoktur. Ana kare seçimi tüm pozların seyrek bir temsilini oluşturacaktır. Bu durum özellik ya da ortam büyüdükçe bir hesap birikmesini engelleyecektir (Strasdat ve ark., 2010). Strasdat ve ark. (2012), BA tabanlı yaklaşımların filtreleme yaklaşımlarından daha iyi doğruluk sağladığını ve daha verimli çözümler ürettiğini göstermiştir. Ancak filtreleme tabanlı yöntemlerden en öne çıkan EKF-SLAM'in doğruluğunu ve verimini artırmaya yönelik çeşitli çalışmalar da halen yapılmaktadır (Quan ve ark., 2019a).

3.4.2.1 Filtreleme Tabanlı VSLAM/VO

VSLAM ve VO'da ardışık olarak ölçümler söz konusudur. Filtre tabanlı yöntemler, gürültülü sensör ölçümlerinden ve modellerden poz, işaretçi konumu gibi uzamsal değerleri tahmin eder. Filtreleme yöntemleri ile yapılan konum ya da haritalama için yapılan durum tahmininde, geçmiş pozlar marjinalize edilir, tahmin işlemi sistemin en son durumuna kısıtlanır ve zaman içinde elde edilen bilgiler olasılık dağılımı kullanılarak temsil edilir. Filtreleme tabanlı SLAM iteratif bir durum tahmini gerektirir. Bunun için benimsenen yöntem Bayes tabanlı filtreleme yöntemleridir. Bayes tabanlı yaklaşımlardaki ana amaç, o zamana kadar olan gözlemleri (sensör bilgilerini) kullanarak sistemin o andaki durumunu olasılıksal olarak tahmin etmektir (Aslan ve ark., 2021a).

Bayes filtresi ile x_t rastgele değişkenlerinin t zamanındaki durumları tahmin edilir. Her bir zaman aralığında hesaplanan sonsal (posterior) olasılık dağılımı ($Bel(x_t)$) belirsizliği de içermektedir. Ardışık olarak gelen ölçüm ve durumların tahmini için Bayes filtresi yinelemeli olarak uygulanır. Sonsal dağılımın özyinelemeli olarak hesaplanabilmesi için Markov kabulleri ve toplam olasılık kuralına bağlı olarak elde edilen özyinelemeli Bayes filtresi denklemi Denklem 3.4'te verilmiştir. Denklem 3.4'te

x, durum değişkenlerini, z sensör (ölçüm) verilerini ya da gözlem değişkenlerini, u kontrol girişini, α ise normalizasyon faktörünü temsil eder. $Bel(x_{t-1})$ bir önceki hesaplamadaki sonsal dağılımdır, yineleme terimidir. Ön bilgi olarak da bilinir. Yani yeni durum önceki durumlara bağlıdır. $P(z_t|x_t)$; ölçüm veya sensör modeli güncellemesi olarak adlandırılırken, $P(x_t|u_t, x_{t-1})$ ise hareket modeli güncellemesi olarak adlandırılmaktadır. Diğer bilinen adlandırma sırasıyla güncelleme ve tahmin etmedir. Denklem 3.4'te de görüldüğü üzere filtreleme tabanlı durum (poz, harita) tahminlerinde, geçmiş durumlar marjinalize edilir.

$$Bel(x_t) = \alpha\, P(z_t|x_t) \int P(x_t|u_t, x_{t-1})\; Bel(x_{t-1})\, dx_{t-1} \tag{3.4}$$

VSLAM uygulaması olarak kullanılan EKF-SLAM çalışması en çok bilinen filtreleme yöntemidir. MonoSLAM ile EKF-SLAM'in VSLAM uygulaması ilk kez sağlanmıştır. Ancak EKF-SLAM'da işaretçilerin artması matematiksel karmaşıklığa neden olur. Ayrıca modein doğrusallaştırılması model için tutarsız tahminler oluşturur. Bu problemlerle mücadele etmek için Değişmez (invariant) EKF (IEKF SLAM) (Barrau ve Bonnabel, 2015), genişletilmiş bilgi filtresi (EIF) (Thrun ve ark., 2004; Cheein ve ark., 2011) gibi teknikler geliştirildi. Ancak Filtre tabanlı yöntemler için ön önemli uygulamalardan biri parçacık filtresine dayalı FastSLAM (Montemerlo ve ark., 2002) yöntemidir. EKF-SLAM'e göre doğruluk ve gerçek zaman gereksinimi açısından daha avantajlıdır. Ancak zamanla örneklerin fakirleşmesi (sample impoverishment) FastSLAM'ın kusurudur.

3.4.2.2 Optimizasyon (BA) Tabanlı VSLAM/VO

Optimizasyon tabanlı çalışmalar doğrusal olmayan en küçük kareler tekniğini kullanarak yansıma hatalarını minimize etmeyi amaçlar. Diğer adı, grafik optimizasyonu SLAM'dir (graph-optimization SLAM). İlk başlarda optimizasyonun yoğun uygulanması hesaplama karmaşıklığına neden olduğundan filtre tabanlı çözümler daha çok tercih edilmiştir. Ancak zamanla problemin seyreltilmesi ve bilgisayar teknolojisindeki gelişmeler sayesinde optimizasyon tabanlı yöntemler tercih edilebilir ve gerçek zamanlı uygulanabilir hale gelmiştir.

VSLAM yapısında bulunan arka uç, kamera pozunu ve 3B anahtar noktası konumlarını tahmin etmek için önceki karelerdeki anahtar noktası eşleşmelerini kullanır.

Filtrelemeden hariç olarak bu görevi yerine getirebilecek algoritmalardan biri optimizasyon tabanlı BA algoritmasıdır. BA, özellik konumlarının, kamera konumlarının ve gerçek kamera parametrelerinin optimize edildiği doğrusal olmayan en küçük kareler problemleri olarak ifade edilen bir toplu optimizasyon (batch optimization) yöntemidir (Triggs ve ark., 1999). BA-tipi optimizasyon yöntemlerine robotik topluluğunda genellikle yumuşatma (smoothing) adı da verilir. BA algoritmasında maliyet (cost) fonksiyonu, özellik yeniden yansıma veya yeniden izdüşüm olarak adlandırılan hata değerleri (feature reprojection errors) ile oluşturulur. BA, 3B özellik koordinatları, kamera pozları ve kalibrasyon parametrelerini kullanarak geometrik parametre tahmini yapar. Bu sayede ortamın görsel olarak yeniden yapılandırılmasının (visual reconstruction) iyileştirilmesi amaçlanır. Bir sahnenin 3B geometrisinin bir dizi resimden yeniden oluşturulması problemini ele alan SFM, BA'yı sıklıkla kullanır ve BA bu anlamda yeni bir yöntem değildir. Filtreleme tabanlı yöntemlerden farklı olarak önceki pozlar ve bunlara bağlı tüm ölçümler, filtrede olduğu gibi marjinalize edilmez ve kaldırılarak tahminlere katkıda bulunmazlar. Bu nedenle filtrelemeye kıyasla, bu yaklaşımda birçok geçmiş poz korunduğu için daha fazla unsur içerecektir. Ancak marjinalleşme eksikliği, pozların seyrek olarak birbirine bağlı kalacağı anlamına gelir. Sonuçta, özellik sayısı çok yüksek olsa bile, BA filtrelemeye nispeten daha verimlidir. (Dellaert ve Kaess, 2006; Strasdat ve ark., 2012; Chen, 2021).

Parallel Tracking and Mapping (PTAM), BA'yı gerçek zamanlı görsel SLAM algoritmasına entegre eden, izleme ve haritalamayı iş parçacıklarına bölen ve ana kare yöntemi tekniğini başlatan ilk yöntemdir. PTAM'ın ortaya çıkmasından sonra, çoğu ORB-SLAM ve LSD-SLAM gibi en iyi VSLAM algoritmaları, PTAM mimarisi üzerine inşa edilerek VSLAM optimizasyon tabanlı yöntemleri daha da geliştirdiler.

3.5. Görsel-Atalet SLAM (VISLAM) ve Görsel - Atalet Odometri (VIO)

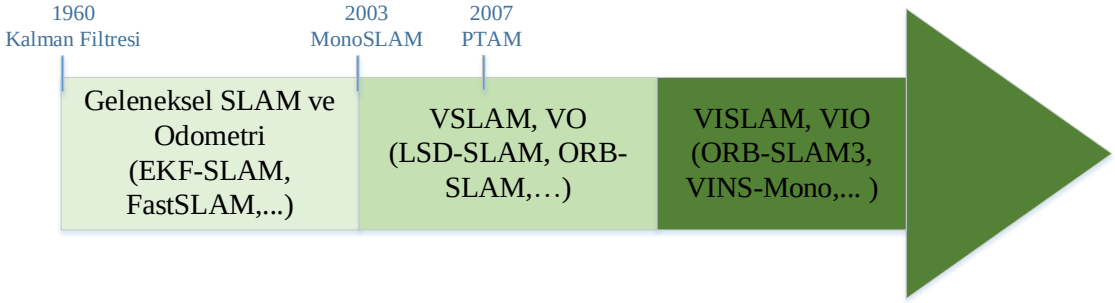
Gelişen bilgisayar teknolojisi ve diğer sensörlere kıyasla düşük maliyetli kameralardan elde edilen zengin görsel bilgiler, VSLAM ve VO yöntemlerine olan ilgiyi artırdı. Bu nedenle son on yıldır genelde araştırmacılar yoğun olarak monoküler VSLAM ve VO yöntemlerinin geliştirilmesine yönelik bir araştırma yürütmektedir. Bu sayede ORB-SLAM, LSD-SLAM, SVO gibi güçlü yöntemler ortaya çıktı. Ancak bu tür yöntemlerde de ölçek ve kayma problemi hala mevcuttur.

VISLAM, lokalizasyon ve haritalama görevleri için kameradan alınan kare bilgileriyle IMU sensöründen alınan bilgileri birleştiren tekniktir. Son yıllarda, görsel

odometri sistemleri, düşük maliyetli donanımı ve zengin görsel sunumu nedeniyle sıklıkla konumlandırma uygulamalarında tercih edilmiştir. Ayrıca, monoküler VO ve VSLAM, ölçek belirsizliği, matematiksel özelliklere duyulan ihtiyaç (örneğin, ORB, SURF, FAST), ani kamera hareketinin neden olduğu görüntü bulanıklığı, düşük dokulu alanlar ve değişken ortam aydınlatma koşulları gibi birçok zorlukla karşı karşıyadır. VSLAM ve VO yavaş hareketlerde zengin bilgi içeriğine sahiptir, ancak kamera hızlı hareket ettiğinde görüntü bulanıklaşır ve özellik noktaları çıkarılamaz veya kamera konumu izlenemez. Üstelik kameraların frekansı yüksek değildir (maks. ~100 Hz). Ataletsel bilgi doğrusal ivme ve açısal hız bileşenlerinden oluşur. Genellikle, IMU sensörleri, kameralardan çok daha yüksek frekansa (~100-1000 Hz) sahiptir. Buna karşılık, kamera hızlı hareket ettiğinde IMU sensörü son derece hassastır. Ancak IMU sensörleri de gürültüden çok çabuk etkilenirler. Bununla birlikte, düşük ivmelerde ve düşük açısal hızlarda zayıf sinyal-gürültü oranı meydana gelir. Sensör gürültüsü nedeniyle, tek başına bir IMU'dan tahmin edilen hareket, hızlı bir şekilde kümülatif olarak kayma eğilimindedir. Hem kameranın hem de IMU'nun avantaj ve dezavantajları incelendiğinde, bu iki bilginin birleştirilmesi dezavantajları ortadan kaldırabilir (Scaramuzza ve Zhang, 2019). VISLAM ve VIO yöntemlerinde, hareket izleme performansını iyileştirmek için bir IMU sensöründen gelen bilgileri birleştirilerek VO ve VSLAM sistemlerinin sağlamlığı artırılır. Bu sayede monoküler kamera kullanılmasından kaynaklı ölçek belirsizliği ve kayma problemleri IMU ile en aza indirilmiş olur. Görsel ve ataletsel bilginin birleştirilmesiyle, VISLAM ve VIO, VO ve VSLAM'e göre daha güçlü sonuçlar üretir. Ayrıca kameralar ve IMU sensörlerinin, mobil cihazlarda yaygın olarak kullanılması da bu uygulamaların geliştirilebilirligi açısından avantajdır (Chen ve ark., 2018; Hong ve Lim, 2018).

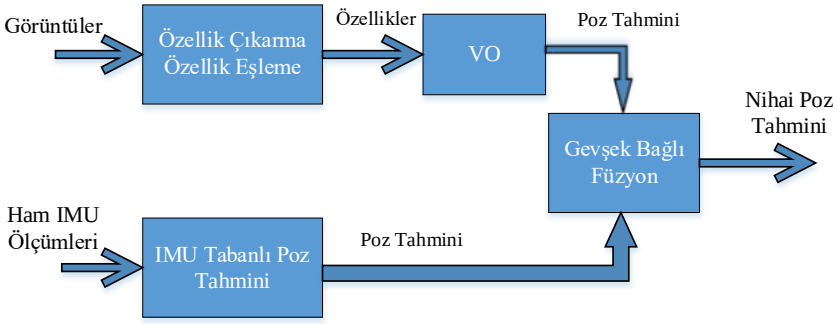
Sağladığı avantajlardan dolayı kameraların ve IMU ölçümlerinin birleştirilmesi önemli bir konu haline geldi. Bu bağlamda MSCKF (Mourikis ve Roumeliotis, 2007), OKVIS (Leutenegger ve ark., 2015), ROVIO (Bloesch ve ark., 2015), VINS-Mono (Qin ve ark., 2018), ORB-SLAM-VI (Mur-Artal ve Tardós, 2017a), ORB-SLAM3 (Campos ve ark., 2021b) gibi IMU sensörlerini kullanan çalışmaların sayısı hızla artmaya başladı. MSCKF, EKF tabanlı olan ve çok sayıda farklı versiyonları olan görsel-ataletsel navigasyon sistemleri için kullanılan bir filtredir. OKVIS, ana kare (keyframe) pozlarının kayan penceresi için doğrusal olmayan optimizasyon kullanan tahmin algoritmasıdır. ROVIO, hem 3B işaretçileri hem de görüntü özelliklerini takip edebilen bir EKF tabanlı bir görsel-ataletsel durum tahmincisidir. VINS-Mono, önceden entegre (optimizasyonda

kullanılmadan önce) IMU değerlerini kullanan doğrusal olmayan optimizasyon tabanlı kayan pencere tahmincisidir. ORB-SLAM-VI ve ORB-SLAM3 mevcut ORB-SLAM yöntemine ilave olarak atalet bilgisini ekler. Şekil 3.8, VSLAM ve VO yöntemlerinin geçmişten günümüze kadar nasıl şekillendiğini göstermektedir (Servières ve ark., 2021).

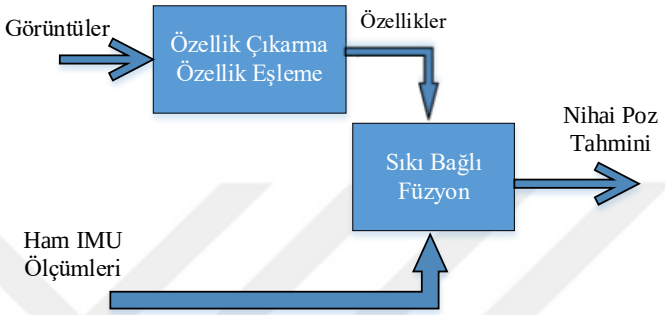


Şekil 3.8. SLAM problemlerinin atalet tabanlı ölçümlere doğru gelişme süreci (Servières ve ark., 2021)

VISLAM ve VIO yöntemleri ile daha doğru sonuçlar sağlamak için, ataletsel ve görsel verinin verimli bir şekilde birleştirilmesi gerekir. Bunun için füzyon tekniğine göre VISLAM ve VIO yaklaşımları, gevşek bağlı (loosely-coupled) ve sıkı bağlı (tightly-coupled) olarak iki gruba ayrılabilir. Her iki tekniğe ait füzyon şekli bir VIO algoritması için Şekil 3.9'daki gibidir. Gevşek bağlanmış yaklaşımlar, yönü ve olası konum değişikliğini tahmin etmek için görsel ve eylemsizlik ölçümlerini ayrı ayrı işler. Yani bağıl hareket ayrı ayrı tahmin edilir, daha sonra nihai tahmin sonucu için atalet ve görsel tahminler işlenir. Bu nedenle, görsel modülde biriken sapma, eylemsizlik ölçümlerinin kullanılması ile ortadan kaldırılamaz. Sonuçta ortaya çıkan tahmin yetersizdir. Sıkı bağlı yaklaşımlar, her iki sensördeki algılama sonucunu en iyi şekilde kullanarak görsel ve eylemsizlik ölçümleri arasındaki sıkı etkileşimi göz önünde bulundururlar. Kamera ve IMU'dan alınan ham ölçümler doğrudan birleştirildikten sonra nihai tahmini bulur. Böylece fazla hesaplama karmaşıklığı olsa da daha yüksek hassasiyetli bir birleşim sağlanır. Gevşek bağlı yaklaşımlarla karşılaştırıldığında, sıkı bağlı yaklaşımlar genellikle daha doğru ve sağlam tahminler üretir. Bu nedenle sıkı bağlı yöntemler daha sık kullanılırlar. VISLAM ya da VIO teknikleri için filtreleme tabanlı ve BA tabanlı yöntemler olmak üzere iki yöntem yaygın olarak kullanılır (Servières ve ark., 2021).



(a) Gevşek Bağlı Füzyon Tekniği



(b) Sıkı Bağlı Füzyon Tekniği

Şekil 3.9. VIO ve VISLAM için füzyon teknikleri

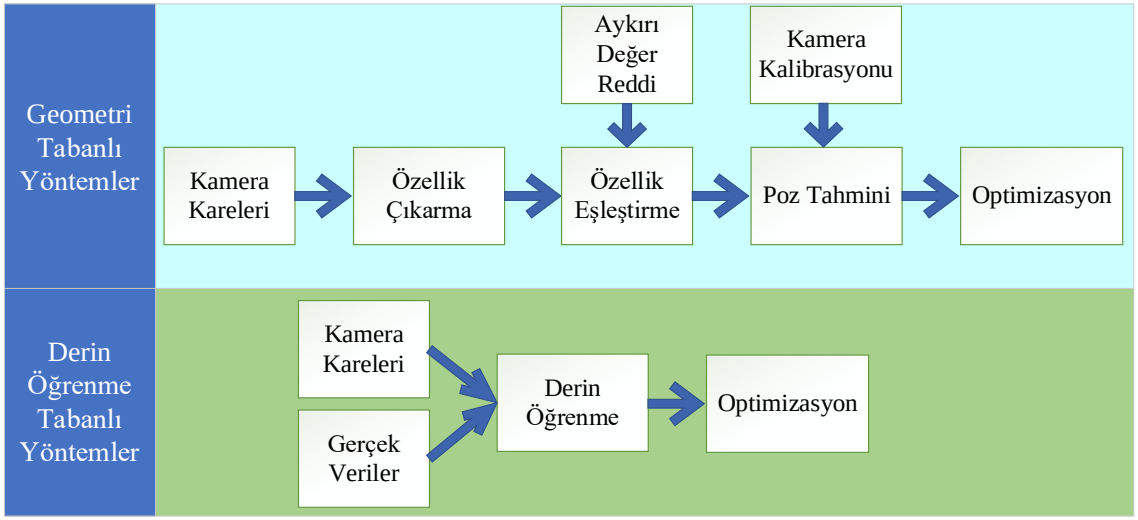
3.6. Derin Öğrenme Tabanlı V(I)SLAM ve V(I)O

Geleneksel olarak VSLAM ve VO problemi geometrik teoriye odaklanır, bu nedenle şimdiye kadar yapılan birçok çalışma, hareketleri tahmin etmek için görüntülerden kısıtlamalar oluşturur. Genel olarak kullanılan görsel yöntemler, görüntülerden çıkarılan geometrik kısıtlamalara dayalı özellik tabanlı (dolaylı) yöntemler ya da tüm görüntünün veya yerel alanın fotometrik hatasına dayanan direk (doğrudan) metotlar olarak sınıflandırılabilir. Özellik tabanlı yöntemlerde, ardışık kamera karelerinden seyrek (sparse) özellikler çıkarılır, bu özellikler takip edilir ve geometrik hesaplamalarla (örn. üçgenleme) özellikler analiz edilerek hareket ya da poz hesaplanır. Özellik tabanlı yöntemlere örnek olarak popüler ORB-SLAM gösterilebilir. Direk metotlarda ise özellik çıkarılmadan doğrudan görüntünün piksellerine göre bir gradyan hesabı gerçekleştirir. Tüm görüntüye ait piksellerin gradyan hesabını yapması, özellik tabanlı yöntemlere göre daha doğru sonuçlar sağlar. Ancak direk yaklaşımlar özellik tabanlılara göre zaman açısından daha maliyetlidir (Engel ve ark., 2013). Direk yaklaşıma örnek olarak popüler LSD-SLAM gösterilebilir. ORB-SLAM ve LSD-SLAM gibi yöntemler gerçek zamanlı performans sağladıkları için mobil robotik için yaygın kullanılmışlardır. Özellik tabanlı ve direk yaklaşımlar kullanılarak şimdiye kadar başarı oranı yüksek çalışmalar gerçekleştirilse de, bu yöntemlerin uygulanması oldukça

zahmetlidir. Bu çalışmalar geometrik sınırlamalara dayandığından, genel olarak geometrik tabanlı olarak adlandırılır (Wang ve ark., 2020c; Zhao ve ark., 2021).

Geometrik tabanlı yaklaşımların sağladığı yüksek performans, genel olarak çalışmaları bu alana yöneltti. Tipik olarak bir geometrik tabanlı VO uygulaması özellik tespiti, özellik eşleme, aykırı değer reddi, hareket tahmini, optimizasyon gibi adımlardan oluşur. Tüm bu işlemler için ayrıca kamera hassas bir şekilde kalibre edilmelidir. VSLAM için ayrıca bir de döngü kapatma ve haritalama işlemi uygulamaya dahil edilmelidir. Görüldüğü gibi geometrik tabanlı gerçekleştirilen bir çalışma, çok sayıda paralel işlem blokları, çok sayıda parametre ayarı ve karmaşık hesaplamalar içerir. Ayrıca aydınlatmanın değişmesi, dokusuz ortamlar, bulanık görüntüler gibi durumlarda geometrik tabanlı yöntemler güvenilir sonuçlar üretmez. Ayrıca elle ayarlanmış çok sayıda parametre ile gerçek dünyanın tam olarak modellenmesi imkânsızdır. Bu durumda bir ortamda iyi çalışan VO/VSLAM algoritmasına ait parametreler, başka bir ortamda zayıf kalır (Wang ve ark., 2017). Bunun için sorun manuel olarak çözülmelidir. Bu insan tarafından belirlenen ayarlar ya da parametreler sonucunda görüntülerden faydalanma oranı değişebilir ve ön yargılı sonuçlar oluşabilir. Bu ve benzer durumlar geometrik tabanlı yöntemlere olan güveni azaltmaktadır (Wang ve ark., 2018).

Geometrik tabanlı çalışmaların bu dezavantajlarını azaltmak/gidermek için son zamanlarda bilgisayarlı görü konularında yüksek başarılar gösteren derin öğrenme ön plana çıktı. Derin öğrenme özellik temsilinin güçlü olmasından dolayı popülerdir. Eğer veri seti yeterliyse, derin öğrenme ile çok daha etkili ve yüksek seviyeli özellikler çıkarılır. Üstelik bu özellikler otomatik olarak çıkarılır ve dokusuz, değişken ortamlar için daha güçlü bir temsil sağlar. Üstelik öğrenmenin en önemli avantajı algıdır. Algı ile yeni ortamlara otomatik olarak uyum sağlanabilir ya da ortamdaki dinamik/statik nesne ayrımı yapılabilir. Dolayısıyla insansı bir algılamayla V(I)O ya da V(I)SLAM çalışmaları daha uygulanabilir hale getirilebilir. Üstelik derin öğrenme ile özelliklerin temsil edilmesi durumunda, klasik VO blok şemasına göre, iş yükü büyük ölçüde azaltılır. Bu durum Şekil 3.10'da gösterilmiştir (Wang ve ark., 2017; Wang ve ark., 2020c). Görüldüğü üzere Derin Öğrenme, önemli görsel işlemler içeren birçok bloğa gerek duymadan VO uygulaması gerçekleştirebilir.



Şekil 3.10. Geometri tabanlı ve derin öğrenme tabanlı VO yöntemleri adımları

Son yıllarda derin öğrenmenin ayırt edilmesi zor görsel özellikler arasındaki ayrımı yapabilmesi ve güçlü tahmin yeteneği sağlaması, durum tahmini içeren görsel problemlerde de kullanılmasını yaygınlaştırmıştır. Ancak oldukça yeni bir alandır ve yeni çalışmalara ihtiyaç duyulmaktadır. Bu bağlamda, V(I)O ve V(I)SLAM çalışmaları yaklaşık son beş yıldır öğrenme tabanlı olarak geliştirilmeye başladı. Görüntüdeki gürültülere karşı sağlamlığı ve kamera kalibrasyonundan bağımsızlığı, veri seti oluşturulmasını kolaylaştırdı. Öğrenme tabanlı yaklaşımların en çok kullandığı V(I)O veri setleri otonom İHA'lar için EuRoC (Burri ve ark., 2016) iken otonom yer araçları için KITTI (Geiger ve ark., 2013)'dir.

Derin öğrenme, doğrudan VSLAM/VO yöntemlerdeki gibi ham görüntüleri doğrudan işleme tabi tutar, ancak kendi içinde bulunan özellik çıkarma adımı sayesinde yoğun SLAM tekniklerindeki karmaşık matematik modellemesini ve başlangıç adımını içermez. Ayrıca SIFT, ORB gibi özellikler kullanan özellik tabanlı yöntemlerden daha güçlü özellik temsili sağlarlar (Wang ve Gao, 2021). Ayrıca derin öğrenmedeki tanıma ve etiketleme sayesinde konumlandırmayı daha akıllı hale dönüştüren semantik SLAM uygulamaları (Xiao ve ark., 2019; Bavle ve ark., 2020) da geliştirilmeye başlanmıştır.

Derin Öğrenme ile karmaşık geometrik dönüşümler ve matematiksel hesaplamalar oldukça az kullanılarak daha başarılı sonuçların elde edilmesi amaçlanır. Şimdiye kadar yapılan, VO/VSLAM yöntemlerini uygulamak için kullanılan derin öğrenme çalışmaları ümit verici bir performans gerçekleştirerek ölçek kayması, özellik çıkarma ihtiyacı gibi problemleri hafifletmiştir. Bu kapsamda genel olarak benimsenen yol, CNN ve RNN tabanlı mimariler kullanılarak poz tahmini geliştirmektir. Kullanılan yöntemlerin çoğu bağıl poz tahmini yöntemini kullanır. Bu yöntem iki veya daha fazla

ardışık görüntüden özellikler çıkararak eğitilir. Çoğunlukla iki görüntü tercih edilir. Ardışık karelerden aynı anda çıkarılan özellikler sayesinde bu iki kare arasındaki hareket değişiminin tahmin edilmesi sağlanır. Genel olarak geometrik özelliklerin çıkarılması CNN mimarileri, sıralı özelliklerin temsili de RNN mimarileri aracılığıyla gerçekleştirilir. VO yöntemi ilk olarak DeepVO (Wang ve ark., 2017) olarak adlandırılan yöntemle sıralı öğrenme algoritması olarak sunuldu. Ardışık iki kare görüntüsünü önce CNN ve daha sonra RNN tabanlı LSTM mimarisine besledi. Sonuçta 6DoF poz bilgisini KITTI veri seti için başarılı bir şekilde tahmin etmeyi başardı.

DeepVO'nun başarısı MagicVO (Jiao ve ark., 2018), ESP-VO (Wang ve ark., 2018), UndeepVO (Li ve ark., 2018), D3VO (Yang ve ark., 2020) gibi yeni VO çalışmalarının önünü açtı. Ayrıca atalet bilgisinin görsel bilgiye sağladığı katkı sonucu, derin öğrenme ağını IMU bilgisiyle de besleyen çalışmalar hızla arttı. Bunlara örnek olarak DeepVIO (Han ve ark., 2019), VINet (Clark ve ark., 2017), SelfVIO (Almalioglu ve ark., 2019) gibi yeni poz tahmin çalışmaları geliştirildi. Ayrıca derin öğrenme, monoküler kamera görüntülerden derinlik tahmini gerçekleştirmek amacıyla da kullanılmıştır (Kim ve ark., 2018; Zhang ve ark., 2018c).

Derin öğrenme tabanlı çalışmalar kendi içinde danışmanlı (supervised) ve danışmansız (unsupervised) olarak ikiye ayrılır. Danışmanlı derin öğrenme yöntemleri, büyük miktarlarda etiketli veri kullanarak çeşitli bilgisayarlı görü problemlerinde sıklıkla kullanılmaktadır. VIO uygulamalarının danışmanlı uygulanmasında girişler kareler ve IMU bilgileri olarak, çıktılar ise poz bilgisi olarak düşünülebilir. Bu girdi ve çıktılarla tasarlanan mimari sayesinde, giriş verisini temsil eden verimli özellik vektörlerinin oluşturulması sağlanır. Bu tür uygulamalar zorlu ortamlarda umut verici bir performans göstermiştir ve üstelik ölçek kayması, parametre ayarı gibi önemli sorunları başarıyla hafifletmiştir. Paylaşılan veri setlerinin artması ve bu veri setleri üzerindeki doğruluk rekabetinin artması sonucunda literatürdeki V(I)O ve V(I)SLAM uygulamalarının çoğu için danışmanlı öğrenme yapısı ile uygulamalar geliştirilmiştir. Ancak son zamanlarda veri seti oluşturmanın zorluğu göz ününde bulundurularak, bir uzman bilgisi gerektirmeyen danışmansız öğrenme yapıları da (örn. SelfVIO (Almalioglu ve ark. (2019)) uygulanmaya başladı. Üstelik ölçüm bilgilerindeki hatalar danışmanlı öğrenme yapıları için bir dezavantajdır.

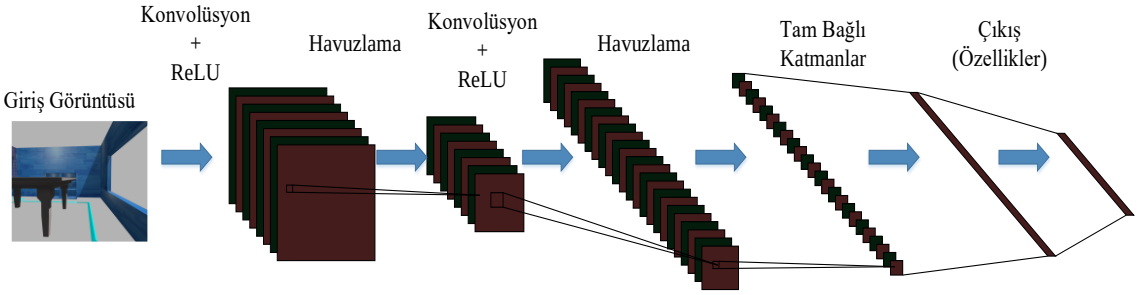
3.6.1. V(I)O ve V(I)SLAM Uygulamalarına Yönelik Derin Öğrenme Yapıları

V(I)O ve V(I)SLAM uygulamaları için kullanılan veriler belirli bir zamanda sıralı olarak oluşturulur ve kaydedilir. Zamanda sıralı bilgilerden, birbirine bağlı özelliklerin çıkarılması için genel olarak CNN tanında RNN mimarileri kullanılır. Bu tür mimarilerden elde edilen özellikler sayesinde kamera ego hareketinin başarılı bir şekilde tahmini hedeflenir. Bu bölüm V(I)O ve V(I)SLAM için sıklıkla tercih edilen CNN ve RNN yapılarını genel ifadelerle özetleyecektir.

3.6.1.1 Konvolüsyonel Sinir Ağları (CNN)

Bilgisayar teknolojisinin gelişmesi, ileri beslemeli Yapay Sinir Ağları (YSA) yapılarının daha da derinleştirebilmesine imkân sağladı. Geleneksel bir YSA ve CNN arasındaki en büyük fark, YSA'da her nöron diğer tüm nöronlara bağlı iken, bir CNN'nin yalnızca son katmanı tamamen bağlı katmanlar içerir. CNN genel olarak bilgisayarlı görü problemlerine çözüm sağlamak için elverişlidir. Buna bağlı olarak obje tanıma, örüntü tanıma, sınıflandırma gibi çok sayıda uygulamaları başarı ile gerçekleştirir. YSA ise bir görüntü ile işlem yapmak istediğinde çok fazla işlem yoğunluğu gerektirir. Çünkü piksel boyutuna göre nöron sayısı artırılmalıdır ve görüntü boyutu yükseldikçe sistemin hızı ve öğrenmesi olumsuz etkilenir (Gogul ve Kumar, 2017). CNN mimarileri, derin yapısı ile giriş olarak verilen görüntülerden katmanları aracılığıyla otomatik olarak özellikler çıkarabilir ve daha sonra bu özellikleri sınıflandırma ya da regresyon amacıyla değerlendirebilir.

Temel CNN mimarilerinde özellik çıkarımı ardışık katmanlar aracılığıyla gerçekleştirilir. Bu ardışık katmanlar, konvolüsyon, havuzlama ve tamamen bağlantılı katmanların belirli bir düzende sıralı yerleştirilmesi ile oluşur. Bir konvolüsyon katmanı, tipik olarak doğrusal ve doğrusal olmayan işlemleri içerir, yani konvolüsyon işlemi ve aktivasyon fonksiyonundan oluşur. Bu sayede konvolüsyon çıkışında doğrusal olmayan özellikler çıkarılır. Konvolüsyon ve havuzlama katmanları giriş görüntüsünün özelliklerinin çıkarılması ve yapılandırılması ile ilgili işlem gerçekleştirirken, son katmanda bulunan tam bağlantılı yapı çıkarılan özelliklerin sınıflandırılmasını sağlar (Yamashita ve ark., 2018).



Şekil 3.11. Klasik CNN yapısı (LeNail, 2019)

Klasik bir CNN yapısı Şekil 3.11'de olduğu gibidir. CNN tabanlı özellik çıkarma adımında, ardışık konvolüsyon katmanları eklemek, yüksek seviyeli özellikleri öğrenme yeteneğini önemli ölçüde geliştirir. Her konvolüsyon katmanı, her biri farklı özellikleri çıkaran birçok farklı filtreden oluşur. Bu filtrelerin görüntüye uygulanması ile özellik haritaları elde edilir. Konvolüsyon katmanının matematiksel tanımı Denklem 3.5'de verilmiştir.

$$O(i,j) = (\mathrm{I} * \mathrm{F})(i,j) = \sum_{m}\sum_{n} F(m,n)I(i-m, j-n) \tag{3.5}$$

Denklem 3.5'te I giriş görüntüsünü, $O\ ixj$ boyutundaki çıkış matrisini (özellik haritası) ve $F\ mxn$ boyutunda $2D$ filtreyi temsil eder. Konvolüsyon adımında, genellikle birden fazla özelliği tespit etmek için birden fazla filtre uygulanır. Filtre, ham görüntü üzerinde kaydırılarak çarpma ve toplama işlemleri yapılır. Her filtrenin uygulanması sırasında, filtre boyutu, filtrenin kaydırma aralığını belirleyen adım değeri (stride) ve konvolüsyon işleminin özellik harita boyutunu korumak için ham görüntüye eklenen dolgu değeri (padding) belirlenir. Konvolüsyon işleminden sonra adım ve dolgu değerlerine bağlı olarak çıkışta oluşan özellik haritasının boyutu Şekil 3.11'de görüldüğü gibi değişir. Yeni matrisin boyutları Denklem 3.6'ya göre hesaplanır.

$$n_o = \frac{n_i - f + 2p}{s} + 1 \tag{3.6}$$

Denklem 3.6'da n_i ve n_o giriş görüntüsünün ve çıkış görüntüsünün boyutunu, f filtre boyutunu, p dolgu değerini ve s ise adım değerini temsil eder. Ardışık evrişim süreçleri sonucunda öznitelik haritasının boyutu genellikle küçülürken derinliği uygulanan filtre sayısına göre artar. Konvolüsyon gibi doğrusal bir işlemin çıktılarını daha sonra doğrusal olmayan bir matrise dönüştürmek için aktivasyon fonksiyonları

kullanılır. Genellikle derin öğrenmede ReLU aktivasyon fonksiyonu sıklıkla tercih edilir. ReLU'nun matematiksel işlevi Denklem 3.7'de verilmiştir (Yamashita ve ark., 2018). x ReLU'ya girdi olarak verilen değerdir.

$$f(x) = \max(0, x) \tag{3.7}$$

Konvolüsyon işleminden sonra uygulanan havuzlama katmanı, konvolüsyon sonucunda oluşan özellik matrisinin belirli bir alanındaki değerlerin genellikle maksimum veya ortalama değerlerini hesaplar. Bu tipik bir alt örneklemedir. Amaç çok sayıdaki özelliği daha az özellikle temsil etmektir. Bu sayede özellik matrisinin boyutu ve parametre sayısı azaltılır. CNN mimarisinde son konvolüsyon ya da havuzlama katmanından sonra elde edilen özellikler öncelikle tek sütunlu bir matris (1 Boyutlu (1B)) olarak ifade edilir, yani düzleştirilir (flattening). Son olarak tam bağlantılı katmanlar CNN mimarisinin sonuna yerleştirilir. Bunlar 1B verilerin tamamını çıktıya bir ağırlık değeri ile bağlar. Birden fazla sayıda olabilir. Yani bir YSA gibi çalışır (Kumar ve ark., 2020).

Üç temel işlem CNN mimarisini oluşturur. Fakat bunlar dışında ağın düzenlileştirilmesi (Regularization) için bu katmanlar arasına farklı görevlere sahip ek katmanlar eklenebilir. Mesela Seyreltme (Dropout) katmanları aşırı öğrenmeyi engellemek için bazı nöronları işleme dâhil etmezler, yani belirli oranda rastgele olarak nöron değerleri sıfırlanır. Paket Normalizasyonu ise önceki katmanların çıktısını normalleştirir. Böylece her katmanın girdisi normalize edilmiş değerlerle beslenir. Bu, öğrenme verimliliğini arttırır ve aşırı öğrenmeyi önler (Kurt, 2018; Kumar ve ark., 2020).

CNN modellerinin yapısından anlaşıldığı gibi, özellik çıkarma ve sınıflandırma (veya regresyon) otomatik olarak gerçekleştirilir. Ayrıca, çıkarılan öznitelikler küçük detayları ayırt edebilir. Bu güçlü özellik çıkarma adımı, derin öğrenmeyi makine öğrenmesi yöntemlerinden üstün kılar (Toraman ve ark., 2020). CNN modellerinin başarısını artırmak, iyi tasarlanmış bir mimari ile mümkündür. Bu nedenle farklı araştırma gurupları veya araştırmacılar farklı şekillerde CNN modelleri tasarlamışlardır. Bu CNN modellerinin mimarisi birbirinden katman, derinlik, parametre, vb. unsurlar açısından farklılık gösterir. Bu mimarilere örnek olarak GoogleNet (Szegedy ve ark., 2015), ResNet18 (He ve ark., 2016), ResNet50 (He ve ark., 2016) ve MobileNet v2 (Howard ve ark., 2017; Sandler ve ark., 2018), Inception (Szegedy ve ark., 2016), vb. modeller örnek verilebilir. Bu popüler CNN modellerinin bir kısmı şu şekilde kısaca

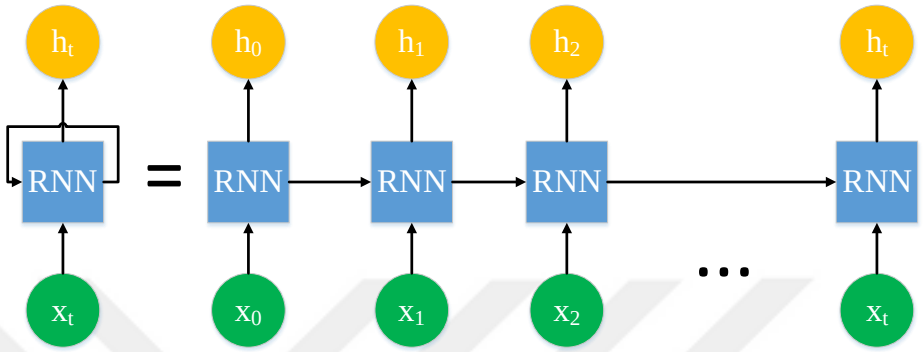
açıklanabilir: GoogLeNet, Google araştırmacıları tarafından geliştirildi ve Inception ağının bir çeşidi olan 22 katmanlı bir modeldir. Çeşitli filtre boyutlarını (1 × 1, 3 × 3 ve 5 × 5) kullanarak özellikleri daha etkin bir şekilde çıkarmak için Inception modülünü kullandı. 1 × 1 konvolüsyon ile ağın derinliği ve hesaplama yükü azaltıldı. ResNet'i farklı ve ayrıcalıklı kılan, gradyanların kaybolması problemini engellemek amacıyla, artık (residual) değerlerin sonraki katmanlara iletilmesi için artık (residual) bloklar kullanmasıdır. Farklı derinliklere sahip ResNet modelleri literatürde mevcuttur. Örneğin ResNet 18 ve ResNet50 sırasıyla 18 ve 50 katman derinliğine sahiptir. Son olarak MobileNet, mobil cihazlarda kullanılmak üzere tasarlanmıştır, bu nedenle düşük hesaplama ve hızlı çalışma gibi avantajlar içerir. Bunun için derinlemesine ayrılabilir (depthwise separable) konvolüsyonlar kullanır. Bu sayede parametre sayısı azalır ve hızı artar. MobileNetV2, 53-katman derinliğine sahiptir, darboğaz özelliklerine sahip ters çevrilmiş artık blokları kullanması ile MobileNet'ten ayrılır. Ayrıca çok daha düşük parametre sayısına sahiptir (Kim ve ark., 2020). Sağladığı farklı üstünlükle sebebiyle, kıyaslama çalışmalarında bu modeller sıklıkla tercih edilir (Ardakani ve ark., 2020) Özellikle Transfer Öğrenme (Transfer Learning (TL)) ile bu önceden eğitilmiş modellerin kullanımı artmıştır. TL sayesinde daha önce eğitilmiş ağ bilgisi kullanılarak az sayıda eğitim verisi ile daha yüksek başarı sağlayan modeller geliştirilebilmektedir (Zhuang ve ark., 2020).

3.6.1.2 Yinelemeli Sinir Ağları (RNN)

Klasik CNN ve YSA yapılarında girişler ve çıkışlar arasında bağımsız bir ilişki vardır. Ancak çoğu zaman veriler arasında bir sıralı ilişki söz konusudur. İnsanlar da benzer şekilde bir karar vermek için önceki girdilerden bağımsız hareket etmez. Girdiler arasındaki bağımsız ilişki varsayımı yapan CNN için bu en büyük kusurdur. Ancak eğer gerçekten bağımsız bir ilişkiye sahip veriler mevcutsa CNN en iyi seçimdir.

RNN, sıralı verilerden özellikler çıkarmak için tasarlanmıştır ve bu sayede dinamik sistemlerin modellenmesi için sıklıkla tercih edilir. Gerçek dünyadaki zamanda sıralı gelen veriler olan metin çevirisi, dil işleme, konuşma tanıma, vb. uygulamalar RNN tabanlı mimariler ile gerçekleştirilir (Sherstinsky, 2020). RNN yapısal olarak içerdiği bir döngü sayesinde geçmiş bilgilerin saklanmasını sağlarlar. RNN'nin iç ağında bir döngü vardır, böylece son zaman adımı bilgisinin nöronda tutulmasına izin verilirken, aynı zamanda bu bilgi önceki adımdaki bilgilerden de etkilenir. Kısaca içerdiği sıralı hafıza ile RNN, CNN'den ayrılır. Bunun için her nöron bir sonrakine bağlanır (Tai ve Liu, 2016).

Şekil 3.12 genel bir RNN mimarisinin bir parçasını gösterir. RNN, x_t vektörünü girdi olarak alır ve çıktı olarak h_t vektörünü üretir. Ancak çıktı olarak üretilen bu vektör, sadece gerçek zamanlı olarak verdiğimiz girdiden değil, daha önce beslenen girdi geçmişinden de etkilenir (Dhruv ve Naskar, 2020). Bu durum RNN yapılarına kısa süreli de olsa bellek özelliği kazandırır.



Şekil 3.12. RNN yapısı

RNN'deki geçmiş durumlara olan bağlantı uzadıkça, geçmiş durumlardan gelen bilgilerin etkisi azalır. Zincir kuralına göre bağlı durumda tekrar tekrar çarpım gradyanların zamanla kaybolmasına neden olur. Bu durumda gradyan değerleri zamanla sıfıra yaklaşır ve bu durumda geçmiş durumlardan öğrenme işlemi yapılamaz. Bu nedenle bağlantı uzadıkça kaybolan gradyanlar sinir ağının öğrenmesini engeller. Bu durumda RNN mimarileri uzun dizi verilerini çözmek için yetersiz kalır (Li, 2020; Şırlancı, 2021).

Kaybolan gradyan problemini çözmek ve daha uzun süreli diziler için çözüm sağlamak için Uzun kısa süreli bellek (Long Short-Term Memory (LSTM)) (Hochreiter ve Schmidhuber, 1997), kapı mekanizmalarını tanıtarak RNN'deki kaybolan gradyan problemini etkili bir şekilde çözmüştür. Ayrıca uzun veri dizilerini öğrenebilir. LSTM, hata gradyanının zamanda geriye akmasına yardımcı olmak için kapıları ve bellek hücrelerini kullanır. Bu şekilde, hata değeri geri yayılım adımında daha iyi korunur ve durumlarını çok daha uzun süre hatırlar (Gers ve ark., 2002). LSTM hücre aktivasyon vektörü (c), giriş kapısı (i), çıkış kapısı (o) ve unutma kapısından (f) oluşur. Hücre aktivasyon vektörü, önceki bellekten (c_{t-1}) ve yeni modüle edilmiş bellekten (c_t) oluşur. Giriş kapısı, belleğe eklenen yeni bellek içeriğinin boyutunu kontrol eder. Yani LSTM yapısına girmek için yeni değerlerin aralığını kontrol eder ve bunu bellek hücresine iletir. Çıkış kapısı, çıkış belleği içeriğini kontrol eder. Unutma kapısı, bilgileri depolamak ve gereksiz bilgileri kaldırmak için (unutulacak hafıza) bellek hücresini kontrol eder. Tüm bu adımların bir sonucu olarak, mevcut gizli durum h_t, mevcut x_t girişi ve önceki x_{t-1}

girdisinin gizli durumu h_{t-1} kullanılarak hesaplanır (Liu ve Guo, 2019; Unlersen ve ark., 2021). LSTM yapısına ait eşitlikler Denklem 3.8 -Denklem 3.13 arasında gösterilmiştir. Burada W gizli katmanın ağırlıklarını ifade eder, b ön bilgi (bias) değerini, $\sigma(\dots)$ sigmoid aktivasyon fonksiyonunu ve $\tanh(\dots)$ hiperbolik tanjant fonksiyonunu gösterir. $\otimes$ işlevi eleman bazında çarpmayı gösterir. Bu eşitliklere göre de LSTM'in blok yapısı Şekil 3.13'de verilmiştir.

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \tag{3.8}$$

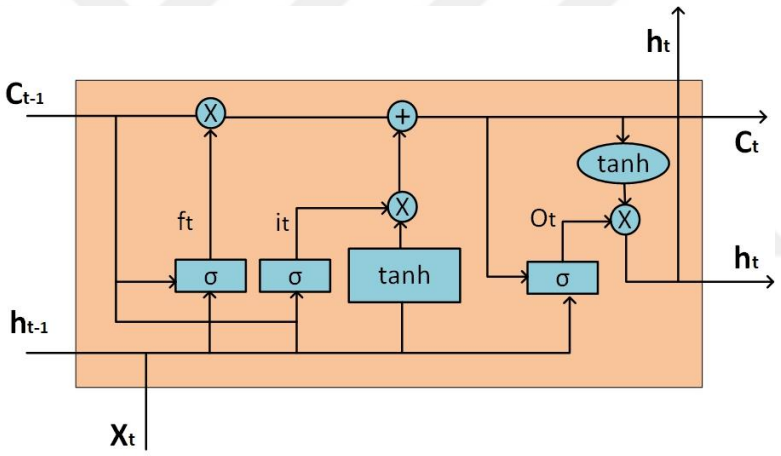
$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \tag{3.9}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \tag{3.10}$$

$$\tilde{c_t} = tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{3.11}$$

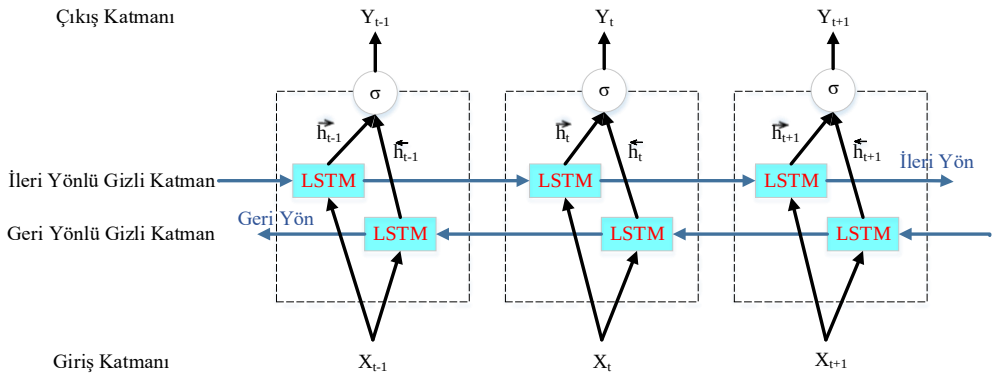
$$c_t = f_t \otimes c_{t-1} + i_t + \otimes \tilde{c_t} \tag{3.12}$$

$$h_t = o_t \otimes \tanh(c_t) \tag{3.13}$$



Şekil 3.13. LSTM yapısı (Unlersen ve ark., 2021)

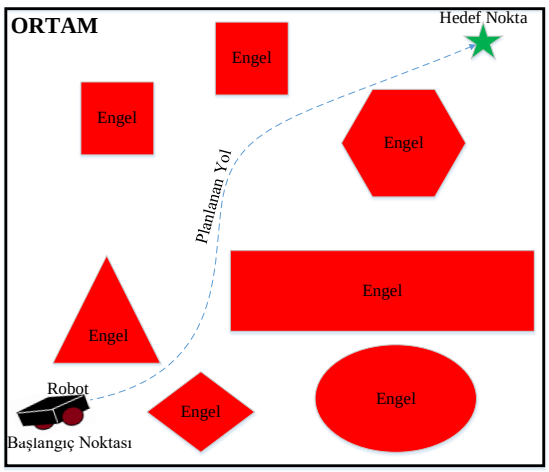
LSTM sayesinde önceki ağın çıktıları hatırlanır ve mevcut verilere bağlanır. Bu, özellikle IMU gibi zaman sıralı verileri işlemek için kullanışlıdır. BiLSTM, LSTM'nin geliştirilmiş bir versiyonudur. BiLSTM, hem önceki hem de sonraki bilgilere erişmek için ileri yönlü gizli katmanı ve geri yönlü gizli katmanı birleştirir. LSTM sadece geçmiş bilgileri kullanırken, BiLSTM geçmiş ve gelecek bilgileri temel alır. Bu nedenle BiLSTM'nin yüksek genelleme kabiliyeti vardır, bu nedenle daha güçlü bir tahmin sağlar. Dolayısıyla, BiLSTM (Graves ve Schmidhuber, 2005) sinyali ileri yöne ek olarak geri yönde de yayarak standart LSTM çerçevesini genişletmiş olur. Yapılan çalışmalar BiLSTM'nin sıralı modelleme görevini LSTM'den daha iyi çözme kabiliyetine sahip olduğunu göstermiştir (Liu ve Guo, 2019). LSTM'ye dayalı BiLSTM'nin blok yapısı Şekil 3.14'te gösterilmektedir.



Şekil 3.14. BiLSTM yapısı

3.7. Yol Planlama

Mobil robotik için uzun zamandır süregelen çaba, otonom özelliğini aktif bir şekilde bu robotlara entegre etmektir. Tam bir otonomi özelliğinin sağlanması için yeterli şart, tamamen insandan bağımsız olarak, robotun bilinmeyen gerçek dünya ortamlarında gezinmesi ve bu ortamda otonom görevler gerçekleştirmesidir (Dissanayake ve ark., 2001). Daha önce de bahsedildiği gibi otonom mobil robot uygulamaları için, üç ana şart sağlanmalıydı. Bu problemler gezinme için lokalizasyon ve haritalama iken, otonom görevler gerçekleştirmek için ise yol planlamadır (Wen ve ark., 2020). Lokalizasyon probleminde, robotun bilinmeyen bir ortamda kendini konumlandırması amaçlanırken, haritalama probleminde ise mobil robot üzerindeki sensörler aracılığıyla alınan veriler kullanılarak ortamın geometrik yapısı oluşturulmaktaydı. Her iki problemin eş zamanlı çözümünü ele alan SLAM probleminin kapsamına ve çözümüne yönelik detaylı anlatım, önceki bölümlerde yapıldı.



Şekil 3.15. Genel bir yol planlama problemi

V(I)O ve/veya V(I)SLAM uygulamaları, bir mobil robotun lokalizasyon ve/veya ortamın harita belirsizliğini en aza indirmeyi amaçlar, bu sayede herhangi bir konumdaki robot, kendini ortam yapısına göre hassas bir şekilde konumlandırır. Ancak SLAM, robotun otonom görevleri gerçekleştirmesi için gerekli olan hareket bilgisini belirlemez (Maurović ve ark., 2017). Hareket bilgisinin oluşturulması için ortam yapısına (yani engellere ve konfigürasyon uzayına (Configuration Space (C-space)'e)) bağlı olarak başlangıç ve hedef noktaları arasındaki yolun planlanması gerekir. C-space, yol planlanması için sistemin sahip olabileceği tüm olası konfigürasyonları temsil eder. Engelle çarpışmayan olası tüm yol planlamaları çarpışmasız (collision-free) C-space olarak adlandırılır (Sansebastiano ve del Pobil, 2021). Şimdiye kadarki yol planlama çalışmaları en optimum çarpışmasız c-space yapısını bulmayı amaçlar (Bak ve ark., 2019).

Yol planlama, bir robotun bir görevi başarıyla gerçekleştirebilmesi için nasıl bir yörünge izlemesi gerektiğini belirlemek amacıyla kullanılır. Genel bir yol planlama görevi için bir ortamda otonom hareket edecek bir robot, engelleri aşarak hedef noktaya ulaşmayı amaçlar. Bunu temsil eden bir görüntü 2B bir ortam için Şekil 3.15'te gösterilmiştir. Otonom mobil robot tasarımının önemli bir parçası olan bu alan, SLAM'a benzer şekilde uzun süredir robotik araştırmacılarının ilgi odağıdır ve 1960'ların sonlarından itibaren yeni yöntemler önerilmektedir (Ajeil ve ark., 2020). Bu yöntemlerden en popüler olanları Artificial Potential Field (APF) (Khatib, 1986), düğüm tabanlı yöntemler (örn. Dijkstra (Dijkstra, 1959), A* (Hart ve ark., 1968) , D* (Stentz, 1997)) ve örnekleme-tabanlı (örn. Hızlı Keşfeden Rastgele Ağaç (Rapidly Random-Exploring Tree (RRT)) (LaValle, 1998), Hızlı Keşfeden Rastgele Ağaç-Yıldız (Rapidly Random-Exploring Tree Star (RRT*))(Karaman ve Frazzoli, 2011), Olasılıksal Yol Haritası (Probabilistic Roadmap (PRM)) (Kavraki ve ark., 1996)) yöntemlerdir (Flores-Caballero ve ark., 2020). Yol planlama yöntemleri ile ilgili geniş teorik bilgi içeren çalışma Zhang ve ark. (2018a) ve Karur ve ark. (2021) tarafından hazırlanmıştır. APF yönteminde tüm olası konfigürasyonlar için bir potansiyel fonksiyon kullanılarak hareket koordinatları belirlenir, ancak yerel minimuma takılmaya çok eğilimlidir. Düğüm tabanlı yöntemler, önceki çalışmalarda sıklıkla tercih edilmiştir. En büyük avantajları, en uygun yolu bulmayı garanti etmeleridir. Bunun için oluşturulan düğümlerin sık ve düzenli dağılması gereklidir. Yani ne kadar sık düğüm oluşturulursa o oranda daha kısa yol bulunabilir. Kısacası, oluşturulan optimum yolun uzunluğu düğüm çözünürlüğüne bağlıdır. Ancak çözünürlüğün artması işlem yoğunluğu ve zaman maliyeti oluşturur.

Özellikle ortamın boyutu arttıkça işlem süresi katlanarak artar, bu nedenle genellikle küçük haritalarda kullanılabilirler (Chaari ve ark., 2017). Örnekleme tabanlı algoritmalar ise daha yeni ve gittikçe popülerliği artan algoritmalardır. Önceki yöntemlere göre en büyük avantajları hesaplama karmaşıklığı azdır, çok daha az düğüm ile hedefe ulaşma imkânı sağlarlar. Bu nedenle geniş haritalarda daha hızlı çalışırlar. Ancak bu yöntemlerin de en büyük dezavantajı oluşturulan yollar rastgele örneklemeye bağlı olduğu için, bu yollar genellikle optimal olmaz (Nasir ve ark., 2013; Wang ve ark., 2020a).

Anlatılan yol planlama yöntemleri ortamın ya da çözümün matematiksel olarak modellenmesini amaçlar, ancak modelleme her zaman iyi sonuç vermez. Arama uzayı genişledikçe karmaşıklık artar. Bu nedenle son zamanlarda, yukarıda bahsedilen yöntemlerden farklı olarak, doğal olaylardan ilham alan doğa esinli teknikler ile, en kısa yol kısıtlamasına dayanan optimizasyon tabanlı yol planlama çalışmaları çoğalmıştır. Yol planlama için kullanılan optimizasyon yöntemlerine örnek olarak PSO (Kennedy ve Eberhart, 1995), DE (Price ve ark., 2006), GA (Davis, 1991), Karınca Kolonisi Optimizasyonu (Dorigo ve ark., 1996) verilebilir (Zhao ve ark., 2018).

Geçmiş çalışmalarda, mobil robot ile yol planlama uygulamalarının büyük bir kısmı kinematik açıdan kolaylık sağladığı için genellikle sadece iki boyuta (2B) odaklanmışlardır. Bu çalışmalarda yükseklik (z ekseni) sabit kabul edilmiştir (Yang ve ark., 2016). Haritanın boyutunun artması, yol arama uzayının genişlemesi sebebiyle karmaşıklığı artıracaktır. Özellikle A* gibi ızgara (grid) tabanlı algoritmaların 3B çözümleri için ızgara sayısı n^2'den n^3'e yükselecektir (Han, 2019). Bu nedenlerle yol planlama için çözümlerin çoğu 2B haritalarla sınırlıdır. 2B ortamlar için yol planlama önerisi sunan yöntemlerden bazılarına örnek olarak şu çalışmalar verilebilir: (Nasir ve ark., 2013; Maurović ve ark., 2017; Williams ve ark., 2017; Noreen ve ark., 2018; Li ve ark., 2020; Wang ve ark., 2020a; Wen ve ark., 2020; Wang ve ark., 2021; Zhang ve Wang, 2021). Bu tür çalışmalar harita üzerinde tekerlekli robotlarla yapılacak görevler için uygundur. Ancak son zamanlarda İHA'ların yükselen trendi, yol planlama için 3B çözümlerin gerekliliğini ve önemini ortaya çıkarmıştır. İHA'lar çevre izleme, arama-kurtarma, iklim izleme, tarım, savunma, uzay, vb. çok sayıda alanda yaygın olarak kullanılmaktadır. İHA'ların en çok tercih edilme sebebi görüntüleme ve hareket esnekliği açısından sağladığı avantajlardan dolayıdır. Ancak İHA'ların en büyük dezavantajı, batarya kapasitelinden kaynaklanan enerji problemidir. İHA'ların minimum enerji harcaması için hedefe en kısa ve basit yoldan ulaşmaları gerekmektedir. Hem İHA'ların hareket esnekliğini verimli kullanmak, hem minimum enerji ile otonom görevler

gerçekleştirmek, hem de gerçek zamanlı uygulamalara uyum sağlamak için günümüzde güçlü ve hızlı 3B yol planlama algoritmaları gereklidir (Aggarwal ve Kumar, 2020; Flores-Caballero ve ark., 2020).

Mevcut yol planlama yöntemleri ele alındığında, düğüm tabanlı yöntemlerin geniş arama uzaylarında maliyetli olması, doğadan ilham alan optimizasyon yöntemlerinin ise hesaplama maliyetlerinin yüksek olması ve yerel minimumda sıkışıp kalması araştırmacıları örnekleme tabanlı yöntemlere yöneltmiştir. Ayrıca şimdiye kadarki önerilen yöntemler içinde, İHA için geniş alanlarda ve 3B ortamlarda hızlı çözüm sağlayacak yol planlama teknikleri örnekleme tabanlı yöntemlerle sağlanabilir. Çünkü örnekleme tabanlı yöntemler düşük hesaplama maliyetine sahiptir ve yüksek boyutlu problemlere uyarlanabilir. Örnekleme tabanlı yöntemler arasında hem zaman hem de uzay karmaşıklığı açısından daha avantajlı yöntem RRT* (Karaman ve Frazzoli, 2011) olmuştur. RRT*'ın geliştirilmeye açık olması ve diğer avantajlarına rağmen, RRT*'ın arama alanı genişledikçe yüksek bellek tüketimi ve hedefe yavaş yakınsama hızı gibi eksiklikleri nedeniyle geliştirilmesi gereklidir (Nasir ve ark., 2013; Noreen ve ark., 2016; Liao ve ark., 2021). Bu nedenle RRT*'ı geliştirmek amaçlı RRT*-SMART(Nasir ve ark., 2013), RRT*FN (Adiyatov ve Varol, 2013), Informed-RRT*(Gammell ve ark., 2014), A*-RRT* (Brunner ve ark., 2013), Adapted-RRT (Kiani ve ark., 2021), P-RRT* (Qureshi ve Ayaz, 2016), PQ-RRT*(Li ve ark., 2020), Informed-RRT*-Connect (Mashayekhi ve ark., 2020), Quick-RRT*(Jeong ve ark., 2019), Cloud-RRT*(Kim ve ark., 2014), F-RRT*(Liao ve ark., 2021), vb. yeni yaklaşımlar ileri sürülmüştür.

3.7.1. RRT*

Bu tez çalışması uygulamasında da RRT* tabanlı yeni bir yol planlama tekniği geliştirilmiştir. Bu nedenle RRT*'ın çalışma yapısına bu bölümde değinilmiştir.

RRT* genel olarak başlangıç noktasından (x_{init}) başlayarak hedef noktaya (x_{goal}) ulaşıncaya kadar obstacle-free uzayda (X_{free}) örneklemeye dayalı random örnekler oluşturur. RRT* algoritması giriş olarak başlangıç noktası (x_{init}), hedef noktası (x_{goal}) ve engelleri içeren ortam bilgisini alır, çıkış olarak $T = (V, E)$ olarak ifade edilen bir ağaç yapısını oluşturur. Bu ağaç x_{init} noktasını (ilk düğüm), x_{goal} (son düğüm) noktasına engelsiz uzay (obstacle-free space) (X_{free}) üzerinden bağlar. Ağaç yapısını oluşturan iki eleman, köşeler (V) ve bu köşeler arasındaki bağlantı kenarlarıdır (E). Ağaç oluşturma döngüsü başladıktan sonra, ilk olarak x_{init} kökü ile başlayan ağaca, daha sonra yinelemeli

olarak yeni dallar (kenarlar) ve yeni köşeler eklenir. Her yinelemede X_{free}'den rastgele örnek (x_{rand}) oluşturulur, daha sonra bu örneğe en yakın ağaç üzerindeki mevcut düğüm ($x_{nearest}$) bulunur. $x_{nearest}$'den x_{rand}'da uzanan doğrusal yol üzerinde, ağacın maksimum uzatma mesafesi (δ) baz alınarak yeni bir örnek (x_{new}) oluşturulur. Eğer x_{new}'e erişim çarpışmasız olarak sağlanabiliyorsa, x_{new} baz alınarak yeni aramalar ve kenarlar oluşturulur. İlk olarak R_{near} yarıçaplı ve x_{new} merkezli bir hiper küre içindeki köşeler (x_{near}) bulunur. Daha sonra x_{near} içindeki köşelerin (vertices) maliyetinin x_{new} aracılığıyla düşürülmesi mümkünse ve bu çarpışmasız olarak sağlanabiliyorsa, bu köşelerin ebeveyni x_{new} olarak değiştirilir. Ayrıca maliyeti düşüren yeni düğüm (x_{min}) kaydedilir. Daha sonra yeni x_{min} ve x_{new} nodeları ağaca (T) eklenir ve son olarak yeni düğümler bağlanır. Tüm bu anlatılanlar RRT* çalışmasını gösteren Algoritma 1 sözde kod (pseudocode) olarak aşağıda gösterilmiştir. Daha kolay anlaşılması için, Algoritma 1'e göre her bir satırın anlatımı aşağıdaki gibidir.

Algoritma 3.1. RRT*

1. T ← $InitializeTree()$;
2. T ← $InsertNode(\emptyset, x_{init}, \text{T})$;
3. **for** $i = 0$ **to** $i = N$ **do**
4. $x_{rand} = sample(i)$;
5. $x_{nearest} = Nearest(T, x_{rand})$;
6. $x_{new} = Steer(x_{nearest}, x_{rand}, \delta)$;
7. **if** $Collisionfree\ (x_{new})$ **then**
8. $x_{near} \leftarrow Near(\text{T}, x_{new}, |V|)$;
9. $x_{min} \leftarrow Chooseparent(x_{near}, x_{nearest}, x_{new})$;
10. T ← $InsertNode(x_{min}, x_{new}, \text{T})$;
11. T ← $Rewire(\text{T}, x_{near}, x_{min}, x_{new})$;
12. **return** T

Satır 1: Başlangıç ayarları, engelsiz alanda $T = (V, E)$ ağaç yapısını oluşturmak için etkinleştirilir. Burada V, örneklenen köşeleri temsil eder ve E, bu köşeleri birleştiren kenarları temsil eder.

Satır 2: x_{init} ağacın ilk kenarsız düğümü olarak atanır.

Satır 3: Döngü başlar.

Satır 4: $Sample$ fonksiyonu ile uzayda random bir örnek (x_{rand}) oluşturulur.

Satır 5: $Nearest$ fonksiyonu ile hedef noktaya ulaşılmadığı müddetçe üretilen örneğe en yakın düğüm noktası ($x_{nearest}$) bulunur.

Satır 6: $Steer$ fonksiyonu ile en yakın düğümden ($x_{nearest}$) örneğe (x_{rand}) doğru maksimum uzatma mesafesi (δ) baz alınarak yeni bir düğüm noktası (x_{new}) oluşturulur.

Satır 7: $Collisionfree$ fonksiyonu $x_{nearest}$ ile x_{rand} arasındaki yolun engele çarpıp çarpmadığını denetler.

Satır 8: $Near$ fonksiyonu ile x_{new} merkezli belirli bir R_{near} yarıçapına sahip bir küre sınırları içindeki bir dizi köşe noktası bulur.

Satır 9: $Chooseparent$, optimum x_{new} yolunu oluşturan tepe noktasını (vertex) bulmak için x_{near} noktalarında arama yapar. Herhangi bir x_{near} noktasının ana düğümünü (parent node) x_{rand} ile değiştirmek, x_{init}'den x_{near}'a giden yolun maliyetinin düşmesine bağlıdır. Eğer daha optimum yolu sağlayan bulunursa, bu yeni düğüm x_{min}'dir.

Satır 10: $InsertNode$ fonksiyonu ile yeni düğümler (x_{min}, x_{new}) mevcut düğümlere eklenir

Satır 11: *Rewire* fonksiyonu yeni düğümlere göre kablolamayı günceller.

Satır 12: RRT*'ın çıktısı olarak ağaç yapısı (T) döndürülür.

4. OTONOM İHA TABANLI UYGULAMALAR

Bu bölümde bu tez kapsamında uygulanan üç farklı uygulama, bu uygulamalarda kullanılan veri setleri ve uygulama sonuçlarını anlatılmıştır.

4.1. Uygulamalarda Kullanılan Veri Setleri

İç ortamlarda otonom İHA sistemlerinin geliştirilmesi için geliştirilen yöntemlerin tamamında danışmanlı derin öğrenme yöntemleri aktif olarak kullanılmıştır. Bu nedenle her bir uygulama sonuçları belirli bir veri seti üzerinden değerlendirilmiştir. İlk iki uygulama sırasıyla İHA için pozisyon ve poz tahmini gerçekleştirir. Bu ilk iki uygulama için kapalı bir ortamda oluşturulan iki adet İHA veri setleri kullanılır. Bunlardan ilki açık (public) bir veri seti olan EuRoC veri setidir. Diğeri ise ROS simülasyon ortamında ürettiğimiz veri setidir. Son uygulamada da yol planlama için çok sayıda 3B harita üretilmiştir. İlk iki uygulama için kullanılan veri seti aynı olduğundan, bu bölümde EuRoC ve ROS veri seti tanıtımları yapılmıştır. Son uygulama için kullanılan veri seti üçüncü uygulama içinde ele alınacaktır.

Önerilen yöntemin halka açık bir veri seti ile test edilmesi, çalışmanın performansını gösterir ve aynı veri setini kullanan önceki çalışmalarla karşılaştırma yapılmasını sağlar. Farklı bir simülasyon ortamı ile gerçekleştirilen deney, önerilen algoritmanın çeşitli ortamlardaki performansını göstererek algoritmanın sağlamlığını kanıtlamaktadır.

4.1.1. EuRoC Veri Seti

European Robotics Challenge (EuRoC) veri seti (Burri ve ark., 2016), VIO/ VISLAM uygulamalarında sıklıkla kullanılan bir kaynak veridir ve ETH üniversitesi araştırmacıları tarafından herkese açık olarak paylaşılmıştır. Gri uzaylı stereo kameradan alınan görüntüler Mikro Hava Aracı ile kaydedilmiştir. Ek olarak, veri seti, yerleşik IMU sensörü ile elde edilen eşzamanlı ivmeölçer ve jiroskop verilerini içerir. EuRoC veri setinde kullanılan IMU sensörünün frekansı 200 Hz, kamera frekansı ise 20 Hz'dir. VIO çalışmaları, kamera ve IMU sensörü için kalibrasyon, gürültü seviyeleri gerektirdiğinden bu bilgi de bu veri setinde yer almaktadır. İHA konumuna it gerçek değerler, bir Leica MS50 lazer takip cihazı ve Vicon hareket yakalama sistemi ile ölçülmüştür. Veri seti iki farklı ortamda kayıt altına alınmış ve ortamın geometrik yapılandırılmasını sağlamak için ortama farklı engeller yerleştirilmiştir. Ayrıca veri seti aynı ortamlar için oluşturulmuş

kolay, orta ve zor kategorilerine sahiptir. Bu kategoriler, İHA'ın hızı, ortamın parlaklığı, görüntü bulanıklığına göre belirlenmiştir.

Bu tez çalışmasında Vicon Room veri seti V1-01, V1-02, V1-03, V2-01, V2-02 ve V2-03 kullanılmıştır. “V1” ve “V2” farklı odaları belirtirken, “01”, “02” ve “03” sırasıyla kolay, orta ve zor kategorilerini temsil eder. Ayrıca bu tezdeki tüm uygulamalar monoküler kamera için geliştirildiğinden, sadece kamera-0 (cam-0) kareleri kullanılmıştır. Şekil 4.1, V1-01 veri setine ait İHA ile elde edilen bazı kareleri göstermektedir.

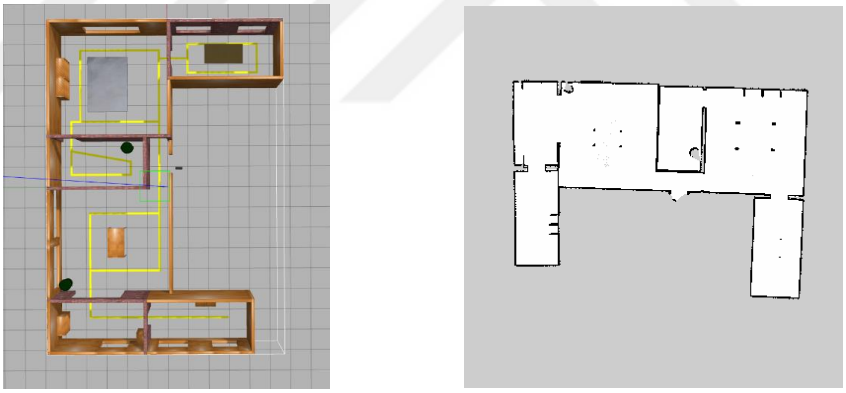
Şekil 4.1. EuRoC veri setine ait bazı kareler (V1-01)

4.1.2. Robot İşletim Sistemi (ROS) ve Tasarlanmış Simülasyon Ortamı

ROS, robotun gerçek dünyadan aldığı bilgileri sensörler aracılığıyla işleyen ve ardından robota komutlar gönderen bir yazılım platformudur. Sensör ve robot arasındaki iletişim sistemi, ROS ara yüzündeki konular (topics) ve mesajlar (messages) yardımıyla yapılmaktadır. Veri seti oluşturmak için kullanılan yazılım platformu, ROS Kinetic ile donatılmış Ubuntu 16.04 işletim sistemidir. ROS, robotik araştırmacıların kendi programlanmış algoritmalarını veya bir insan operatörün algoritmalarını takip eden otonom robotlar geliştirmesini ve uygulamasını kolaylaştırır. Robotların davranışını yönetmek için karmaşık, yüksek düzeyde ölçeklenebilir ve modüler paketlerin kullanılmasını sağlar. Ayrıca hem C++ hem de Python programlama dillerini destekler (ROS, 2020). Ayrıca, algoritmaları hızlı bir şekilde test etmeye, robot tasarlamaya, regresyon testleri gerçekleştirmeye ve gerçekçi senaryolar kullanarak yapay zekayı eğitmeye olanak tanıyan Gazebo simülasyon yazılımına sahiptir. Gazebo, karmaşık iç ve dış ortamlarda çeşitli robotların doğru ve verimli simülasyonunu sağlar (Koenig ve Howard, 2004).

Eğitim ve test verilerinin oluşturulması için simüle edilmiş bir iç mekan ortamında İHA türü olarak DJI Tello[1] kullanılmıştır. DJI Tello, düşük maliyetli bir İHA'dır ve 8 m/s'ye kadar maksimum hıza ve 30 m'lik maksimum yüksekliğe çıkabilir. Ayrıca, yaklaşık 80 g ağırlığındadır, 720 piksellik HD görüntü aktarımı yapabilir. Video çekebilen 5 MP kamera ile donatılmıştır ve saniyede 30 kare hızına sahiptir (30fps) (RYZE, 2020). Ucuz ve kullanışlı olmasından dolayı Tello robotik araştırmacılarının görsel tabanlı algoritmalarını test etmeleri için uygun bir platform olmuştur.

Simülasyon için oluşturulan ortam 16 × 10 m boyutundaydı ve Gazebo 7'de tasarlanmıştır. Şekil 4.2 veri toplama için kullanılan simülasyon ortamını gösterir. Veri toplama sırasında İHA, üç farklı doğrusal ve açısal hızla havalanır ve uçar ve kamera çerçevelerini ve IMU verilerini ve bunlara karşılık gelen 6DoF gerçek pozlarını kaydeder. Eğitim hesaplama maliyetini azaltmak için RGB kare çözünürlüğü 640 x 480 x 3'e düşürülmüştür ve kare hızı 30 olarak ayarlanmıştır. IMU ölçümünün açısal hızı ve doğrusal ivmesi 300 Hz'de kaydedilir. Şekil 4.3, simülasyon anında İHA'nın bir görüntüsünü ve anlık kamera görüntüsünü göstermektedir.



a)Simülasyon Ortamı b) Ortamın 2B haritası

Şekil 4.2. İHA ile veri toplamak için oluşturulan simülasyon ortamı

a) Tello İHA'nın simülasyon anında görüntüsü b) Tello ile elde edilen anlık kare

Şekil 4.3. Tello İHA ile simülayon ortamında veri toplanması

[1] https://www.dji.com/

4.2. UYGULAMA 1: HVIOnet: İHA Konum Tahmini için Derin Öğrenme Tabanlı Hibrit VIO Yaklaşımı

Sensör füzyonu, otonom sistemlerde lokalizasyon problemini çözmek için yaygın olarak kullanılmaktadır. Bu uygulamada, İHA konum tahmini için kamera ve IMU arasında derin öğrenme tabanlı sensör füzyonu gerçekleştirilmiştir. Yaklaşımımızda, ilk olarak, herhangi iki kare arasındaki ham IMU verileri bir BiLSTM katmanına girdi olarak verilir ve çıkışta üç eksenli konum tahmini elde edilir (atalet konum tahmini). Bu iki kare daha sonra tasarlanan konvolüsyonel ağa girdi olarak uygulanır ve konum tahmini tekrar yapılır (görsel konum tahmini). Son aşamada, her iki tahmin adımında çıkarılan görsel-atalet özellikleri birleştirilir ve BiLSTM katmanına girdi olarak verilir. İHA'nın 3 eksenli hareketi, füzyon özellikleri kullanılarak tahmin edilir. Önerilen yöntem, EuRoC ve ROS simülasyon ortamında üretilen İHA verileri kullanılarak test edilmiştir. Sonuçlar önerilen yöntemin uygulanabilirliğini göstermekte ve çalışmanın metodolojik katkısını kanıtlamaktadır. Bu çalışmanın başlıca katkıları şunlardır:

(a) İHA'yı bilinmeyen kapalı ortamlarda konumlandırmak için CNN ve BiLSTM katmanlarından oluşan bir hibrit VIO mimarisi önerilmiştir.

(b) Önerilen uçtan uca algoritma, girdi olarak ham görüntüleri ve ham IMU verilerini kullanır, özellik çıkarımı gerektirmez.

(c) Giriş görüntüleri için kamera kalibrasyon ayarlarına gerek yoktur. Bu nedenle kalibrasyon hatası yoktur.

(d) Çevrenin 3B yapısına oluşturmadan konumlandırma yapılır; dolayısıyla hesaplama süresi hızlıdır.

(e) Bu algoritma, hızlı bir şekilde İHA'nın konumunu tahmin edebilir ve uygulanması optimizasyon tabanlı VIO ve VISLAM algoritmalarına göre daha kolaydır.

Bu uygulamanın anlatılması için bu bölümde şu şekilde yapılandırılmıştır. İlk olarak derin öğrenme için önerilen mimari ana hatlarıyla tanıtılmıştır. Bölüm 4.2.2 önerilen yöntemin EuRoC ile uygulanmasına ilişkin ayrıntılı bilgi ve sonuçları paylaşmaktadır. Bölüm 4.2.3, ROS ortamında oluşturulan veri setimiz ile benzer deneyleri ve sonuçlarını paylaşmaktadır.

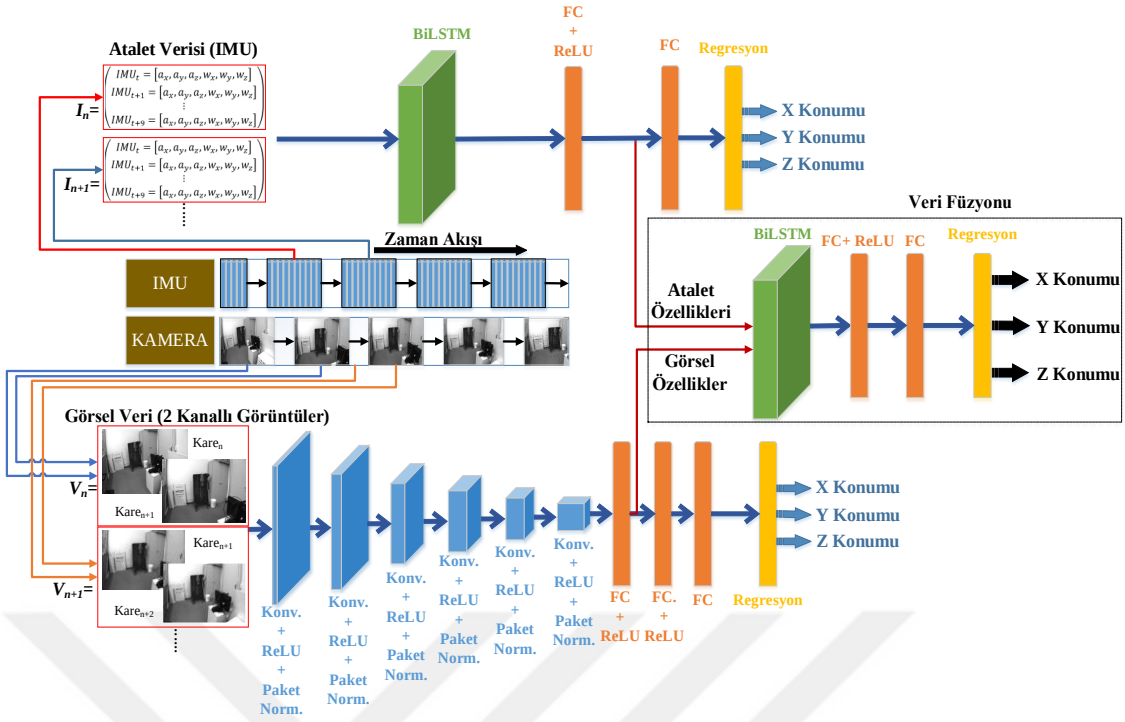
4.2.1. Önerilen Derin Mimari

Sensör füzyonu genellikle aynı büyüklüğü ölçen birbirini tamamlayıcı sensör verileriyle gerçekleştirilir. Bu çalışmada, İHA'nın konumlandırılması için görüntüler ve IMU ölçümleri birleştirilmiştir. Atalet ve görsel verileri, CNN ve BiLSTM katmanlarının

bir kombinasyonunu içeren bir derin öğrenme mimarisi kullanılarak birleştirilir. Tasarlanan derin mimari ilk olarak EuRoC veri seti ile test edilir; ve daha sonra ROS ile üretilen simülasyon verileri kullanılarak önerilen yöntemin etkisi kanıtlanır.

Önerilen hibrit mimari, girdi olarak ardışık kareler, ivmeölçer ve jiroskop verilerini kullanan ve çıktı olarak platformun konumunu tahmin eden uçtan uca bir yapıya sahiptir. Geleneksel yöntemlerdeki elle belirlenerek çıkarılan özelliklerin aksine, sistem doğrudan tahmin için ham sensör verilerini kullanmaktadır. Bu tasarım, sistem doğruluğunu artırmak için ağ mimarisinin tasarımını kritik hale getirmektedir. Şekil 4.4, İHA konum tahmini için önerilen CNN-BiLSTM uçtan uca hibrit ağ mimarisini göstermektedir. Önerilen yaklaşım içinde üç mimari barındırır: görsel tabanlı mimari, atalet tabanlı mimari ve füzyon mimarisi. Görsel tabanlı CNN mimarisi, toplam 10 konvolüsyon katmanı, ReLU katmanı, Paket Normalizasyon Katmanı, Tam Bağlantılı (FC) katmanlar ve son olarak regresyon (tahmin) katmanı içermektedir. Atalet tabanlı mimari ve füzyon mimarisinin her ikisi de BiLSTM, FC ve regresyondan oluşan 4 katman içermektedir.

Önerilen model ile, hem atalet hem de görsel mimarilerin eğitimi ayrı ayrı gerçekleştirilmiştir. Şekil 4.4'te görüldüğü gibi, her mimari regresyon katmanı ile sonlandırılmıştır. Bu şekilde her bir ağın eğitimi gerçekleştirilmiştir. Füzyon mimarisi için kullanılacak görsel-atalet özellikler, diğer iki mimari eğitimi tamamlandıktan sonra elde edilir. Eğitim adımında, özellik vektörleri, hem kamera hem de IMU ham verilerinden oluşturulur. Görsel özellik çıkarma aşamasında, n ve $n+1$ zamanındaki ardışık iki kamera çerçevesi, V_n, V_{n+1}, ... (bkz. Şekil 4.4), birleştirilmiştir ve bu karelerden CNN katmanları aracılığıyla özellikler çıkarılmıştır. Atalet özniteliklerini çıkarmak için, BiLSTM katmanı, iki kamera karesi arasındaki her bir sıralı IMU veri dizisine, I_n, I_{n+1}, ... (bkz. Şekil 4.4) uygulanır. EuRoC'de ve kendi ürettiğimiz veri setimizde, IMU frekansı kamera frekansının 10 katıdır. Dolayısıyla Şekil 4.4'te görüldüğü gibi iki kare arasında 10 adet IMU verisi bulunmaktadır. Görsel, atalet ve füzyon mimarileri ile çıkarılan özniteliklerin eğitimi FC ve regresyon katmanları ile sağlanmaktadır. Tasarlanan mimarinin atalet, görsel ve füzyon kısımlarına ait ayrıntılı bilgi aşağıdaki başlıklarda izah edilmiştir.



Şekil 4.4. VIO için tasarlanmış uçtan uca derin öğrenme mimarisi

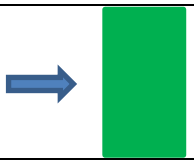
4.2.1.1 Atalet Verisinin İşlenmesi

Ataletsel özellik çıkarma adımını içeren atalet tabanlı mimari, Şekil 4.4'in üst kısmında yer almaktadır. t zamanındaki bir IMU verisi, Denklem 4.1'deki gibi ivmeölçer (a_x, a_y, a_z) ve jiroskop (w_x, w_y, w_z) verilerinden oluşmaktadır. Bu aşamada gerçekleştirilen işlem, her iki kamera karesi arasındaki 10 adet IMU verisinin zamana bağlı özelliklerinin çıkarılmasıdır. Bunun için iki kare arasındaki 10 adet jiroskop ve ivmeölçer verisinden oluşan 10×6'lık veri BiLSTM ağına giriş olarak uygulanmıştır (bkz. Tablo 4.1).

$$IMU_t = [a_x, a_y, a_z, w_x, w_y, w_z] \tag{4.1}$$

BiLSTM katmanı çıktısında, zamana bağlı IMU verilerinden öznitelik vektörleri elde edilmektedir. Şekil 3.14'teki BiLSTM yapısı göz önüne alındığında, $X_{t-1}, X_t, X_{t+1}, \ldots$ IMU verilerine karşılık gelir. Çıktı katmanındaki özellikler $Y_{t-1}, Y_t, Y_{t+1}, \ldots$ sırasıyla FC, ReLU ve regresyon katmanlarından geçer. Ağın eğitimi için çıktı olarak x, y ve z gerçek konum değerleri hedef olarak verilmiştir. Tablo 4.1'de görüldüğü gibi çıkış olarak verilen konum değerleri, son IMU giriş verisi zamanından bir sonraki zamandaki gerçek konum değerleridir. Tablo 4.1, atalet verilerinin eğitimi ve özellik çıkarımı için n. giriş ve n. çıkış verilerini göstermektedir.

Tablo 4.1. Atalet özellik çıkarma ağının yapısı

Atalet Giriş Verisi	BiLSTM Katmanı	Gerçek Konum Değerleri
$I_n = \begin{pmatrix} IMU_t = [a_x, a_y, a_z, w_x, w_y, w_z] \\ IMU_{t+1} = [a_x, a_y, a_z, w_x, w_y, w_z] \\ \vdots \\ IMU_{t+9} = [a_x, a_y, a_z, w_x, w_y, w_z] \end{pmatrix}$		$Y_n = (x_{t+10}, y_{t+10}, z_{t+10})$

Bu adımdaki eğitimin parametreleri deneme yanılma yöntemi ile belirlenmiştir. Eğitim aşamasında, ağırlıkları hata değerine göre güncellemek için Adam (Kingma ve Ba, 2014) optimizasyon algoritması kullanılmıştır. Adam optimizasyon yöntemi, derin sinir ağlarını eğitmek için özel olarak tasarlanmış uyarlanabilir bir öğrenme oranı algoritmasıdır. Adını uyarlamalı moment tahmininden alan Adam algoritması, her bir farklı parametre için bireysel öğrenme oranlarını hesaplar. Adam algoritması, momentum ve RMSprop algoritmaları ile gradyan azaltma yönteminin bir kombinasyonudur (Tieleman ve Hinton, 2012). Nispeten düşük bellek gereksinimi nedeniyle diğer optimizasyon algoritmalarına üstünlük sağlar (Kingma ve Ba, 2014). RMSProp'te olduğu gibi bir parametre güncellemesine sahiptir, ancak farklı olarak bir de momentum terimi içermektedir. Adam algoritması için parametre güncelleme Denklem (4.2) - Denklem (4.4) aracılığıyla gerçekleştirilmiştir. Bu denklemlerde görüldüğü gibi, algoritma, sinir ağının her ağırlığı için öğrenme oranını ayarlamak için eğimin birinci ve ikinci momentlerini tahmin eder. Yani güncellenecek terimin birinci ve ikinci kuvvetinin beklenen değeri kullanılır. Denklemlerde kullanılan harflerin temsil ettiği değerler şöyledir: m: Gradyan hareketli ortalaması, v: Gradyan karesinin hareketli ortalaması, θ : Güncellenecek ağ parametre değeri, $E(\theta)$:Kayıp (loss) Fonksiyonu, $\nabla E(\theta)$: Kayıp (loss) Fonksiyonunun Gradyanı, , β_1: Gradyan Bozulma Faktörü, , β_2: Kare Gradyan Bozulma Faktörü, ϵ: Epsilon(ϵ sıfıra bölünme hatasını engellemek için eklenen bir sabittir.), α: Öğrenme Hızı, l: Yineleme Sayısı (Kingma ve Ba, 2014; Yurttakal, 2019).

$$m_l = B_1 m_{l-1} + (1 - \beta_1)\nabla E(\theta_l) \tag{4.2}$$

$$v_l = B_2 v_{l-1} + (1 - \beta_2)[\nabla E(\theta_l)]^2 \tag{4.3}$$

$$\theta_{l+1} = \theta_l - \frac{\alpha m_l}{\sqrt{v_l} + \epsilon} \tag{4.4}$$

4.2.1.2 Görsel Verisinin İşlenmesi

Görsel öznitelik çıkarımını içeren görsel tabanlı mimari, Şekil 4.4'ün alt kısmında yer almaktadır. Şekil 4.4'te, ardışık iki karenin birleştirildiği ve ardından CNN ağına girdi

olarak verildiği görülmektedir. Yalnızca giriş görüntülerinin boyutu değiştirilmiştir. Eğitimin daha hızlı olması için her görüntünün boyutu 0.25 ile çarpılmış yani her görüntünün yeni piksel boyutları 120×188 olarak belirlenmiştir. Amaç, konumu tahmin etmek olduğundan, mevcut karedeki konum, bir önceki kare ile doğrudan ilişkilidir. Bu nedenle ağın güçlü bir öğrenme sağlaması için CNN ağına iki ardışık kare verilmiştir. Bu durumda tek bir görsel veri için giriş boyutu 120×188×2'dir. Çıkış olarak verilen konum değerleri Tablo 4.1'deki çıkış değerleri ile aynıdır. Çünkü Tablo 4.1'deki IMU verisi bu bölümde giriş olarak verilen kareler arasında gerçekleşir. Bu ayarlama veri setlerindeki zaman damgası yardımıyla gerçekleştirilmiştir. Sonuçta görsel ve atalet verisi paralel olarak aynı zaman sınırındaki verileri ele alır, yani hedef konum değerleri aynı olmalıdır. Görsel özellik çıkarımı ve eğitim adımları için kullanılan ağ yapısı Tablo 4.2'de gösterilmiştir.

Tablo 4.2. Görsel özellik çıkarma ağının yapısı

Görsel Giriş Verileri	CNN	Gerçek Konum Değerleri
		$Y_n = (x_{t+10}, y_{t+10}, z_{t+10})$

Bu adımda, hedef olarak belirlenen gerçek konum değerlerine göre eğitim adımında ağı optimize etmek için Momentumlu Stokastik Gradyan İnişi (Stochastic Gradient Descent With Momentum (SGDM)) optimizasyon algoritması kullanılmıştır. SGDM algoritması için kullanılan parametre güncelleme formülü Denklem 4.5'te verilmiştir. Amaç, Kayıp fonksiyonunu $E(\theta_l)$ minimize etmektir. Bunun için kayıp fonksiyonunun negatif gradyanı yönünde küçük adımlarla ilerlenerek minimum hata değeri bulunmaya çalışılır. Bu adımların boyutu öğrenme oranı (α) ile belirlenir. Momentum değeri (γ), önceki yinelemedeki parametre değerinin mevcut yinelemedeki parametre değerine katkısını belirler.

$$\theta_{l+1} = \theta_l - \alpha \nabla E(\theta_l) + \gamma(\theta_l - \theta_{l-1}) \tag{4.5}$$

4.2.1.3 Füzyon Verisinin İşlenmesi

Tablo 4.3, ağ yapısını ve bu uygulamada görsel-ataletsel füzyon olarak hazırlanan verileri göstermektedir. Amaç Kare$_{n+1}$ adlı görüntüyü kaydeden kameranın anlık konumunu belirlemektir. Geçerli konum, bir önceki konum ve iki kare arasındaki hareket

(atalet) bilgisi ile doğrudan ilişkilidir. Bu nedenle, bu üç veri bilgisi, önceki adımlarda çıkarılan atalet ve görsel özelliklerin birleştirilmesiyle tek bir vektörde elde edilmiştir. Elde edilen vektör sıralı ve zamana bağlı olduğundan, bu adımda en uygun eğitimi gerçekleştirmek için BiLSTM kullanılmıştır. Hedef olarak gösterilen konum değerleri, önceki özellik çıkarma adımlarıyla aynıdır. Zaten Kare_{n+1} görüntüsü, son IMU verisinden (IMU_{t+9}) bir sonraki zamanda $(t+10)$ elde edilir. Bu nedenle, $t+10$ zamanı, Kare_{n+1}'den elde edilen gerçek konum değerlerini göstermektedir. Bu durum Tablo 4.3'ten de anlaşılabilir.

Tablo 4.3. Görsel-atalet özellik çıkarma ağının yapısı

Görsel-Atalet Giriş Özellikleri	BiLSTM	Gerçek Konum Değerleri
		

BiLSTM katmanından çıkarılan öznitelikler, FC ve regresyon katmanlarında yapılan tahmin sonucunda son eğitim süreci tamamlanır. Artık, önerilen yöntemin performansı, test verilerinin eğitilmiş ağlara doğrudan uygulanmasıyla belirlenebilir. Test verilerini kullanan konum tahmin algoritmasının sözde kodu (pseudo) Algoritma 4.1'de gösterilmektedir.

Algoritma 4.1. : İHA Konum Tahmini

While İHA hareket halinde
  for n=1,..,number of test data
    // Atalet Özellik Çıkarma
    Giriş: Atalet verisi,
    BiLSTM ile özellik çıkar
    FC katmanı ile çıkarılan özellikleri al
    Çıkış: Atalet Özellikleri
    // Görsel Özellik Çıkarma
    Giriş: Görsel veri,
    CNN ile özellik çıkar
    FC katmanı ile çıkarılan özellikleri al
    Çıkış: Görsel Özellikler
    // Özellikleri Füzyon Etme
    Görsel-atalet özellikleri birleştir
    Giriş: Füzyon verisi,
    BiLSTM ile özellik çıkar
    Konum Tahmini gerçekleştir
    Çıkış: Tahmin edilen İHA konumları
  end
end

4.2.1.4 Hatanın Ölçülmesi

Önerilen yöntemin konumlandırma hatasını ölçmek için tercih edilen metrik, ortalama karekök hatası ((Root Mean Square Error) (RMSE)'dır. RMSE, öğrenme algoritmalarında regresyon uygulamalarında performans metriği olarak yaygın olarak tercih edilir. Tahmin edilen veri ile gerçek veri arasındaki farkı hesaplayan bir yöntemdir. RMSE formülü Denklem (4.6)'da gösterilmiştir. Denklem (4.6)'da, $\widehat{y_1}, \widehat{y_2}, \widehat{y_3}$, ... tahmini değerleri temsil eder, y_1, y_2, y_3, ... hedef değerleri ve n toplam veri sayısını belirtir.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\widehat{y_i} - y_i)^2}{n}} \tag{4.6}$$

4.2.2. EuRoC Veri Seti İle Deneyler Ve Sonuçlar

Bu bölümde önerilen mimarinin EuRoC veri seti ile gerçekleştirilmesi için parametre ayarları ve sonuçlarından bahsedilmiştir.

4.2.2.1 Parametre Ayarları

Atalet öznitelik çıkarma bölümünde, uygulamanın tam olarak anlaşılabilmesi için derin öğrenme mimarisinin hiperparametre değerlerinin bilinmesi gereklidir. Şekil 4.4, Atalet özellik çıkarımı için tasarlanmış ağın mimarisini (BiLSTM →FC + ReLU →FC) zaten göstermektedir. Bu mimaride kullanılan her katman için önemli hiperparametreler ve eğitim parametreleri Tablo 4.4'te gösterilmiştir. FC1 ve FC2 sırasıyla birinci ve ikinci FC katmanlarını temsil etmektedir.

Tablo 4.4. Atalet özellik çıkarma için kullanılan katmanların hiper parametreleri ve eğitim parametreleri

BiLSTM			**FC1**		**FC2**	
Gizli Katman Sayısı	**Aktivasyon Fonksiyonu**	**Kapı Aktivasyon Fonksiyonu**	**Çıkış Boyutu**	**Aktivasyon Fonksiyonu**	**Çıkış Boyutu**	**Aktivasyon Fonksiyonu**
100	$tanh$	$sigmoid$	100	$tanh$	3	$tanh$
Eğitim Parametreleri						
Optimizasyon Algoritması	$\boldsymbol{\beta_1}$	$\boldsymbol{\beta_2}$	**Maks. Epok**	**Küme (Mini Batch) Boyutu**	$\boldsymbol{\alpha}$	ϵ
Adam	0.9	0.999	250	256	0.001	10^{-8}

Tablo 4.4'teki katmanların özellikleri deneme yanılma yöntemiyle belirlenir. Eğitim için IMU verilerinin %85'i BiLSTM katmanına girdi olarak verilmiştir. Regresyon katmanında İHA'nın x, y ve z konumları hedef olarak gösterilmiştir. Bu nedenle FC2

katmanının çıktısı üç değişken içermektedir. Eğitim sonucunda belirlenen ağırlıklara göre öznitelikleri çıkarmak için FC1 katmanı eklenir. FC1 katmanı ayrıca atalet ve görsel özelliklerin birleştirilmesi için kullanılır. Tablo 4'te belirlenen parametreler Denklem 4.2 - Denklem 4.6'da gösterilen Adam optimizasyonu algoritması için kullanılır ve eğitim aşaması, ağırlıkların güncellemesi bitince sonlandırılır.

Tablo 4.5. Görsel özellik çıkarımı için kullanılan CNN ve FC katmanlarının hiperparametreleri ve eğitim parametreleri

CNN Layers						
Katman Adı	**Filtre Boyutu**	**Aktivasyon Fonksiyonu**	**Dolgu (Padding)**	**Adım (Stride)**	**Paket Norm.**	**Filtre sayısı**
Conv-1	3	ReLU	0	1	Var	64
Conv-2	3	ReLU	0	1	Var	128
Conv-3	3	ReLU	0	1	Var	256
Conv-4	3	ReLU	0	1	Var	512
Conv-5	3	ReLU	0	1	Var	1024
Conv-6	3	ReLU	0	1	Var	6
FC Katmanları						
Katman Adı	**Çıkış boyutu**	**Aktivasyon Fonksiyonu**	**ReLU**	**Seyreltme Değeri (Dropout)**		
FC1	100	$tanh$	Var	Yok		
FC2	50	$tanh$	Var	Var		
FC3	3	$tanh$	Yok	Yok		
Eğitim Parametreleri						
Optimizasyon Algoritması	**Maksimum Epok**	**Küme (Mini Batch) Boyutu**	$\boldsymbol{\alpha}$	$\boldsymbol{\gamma}$		
SGDM	15	256	0.001	0.95		

Görsel özellik çıkarımı bölümünde, uygulamanın tam olarak anlaşılabilmesi için ağın eğitim aşamasında kullanılan hiperparametreler bilinmelidir. Çünkü bu değerler tahmin edilen sonucu ve çıkarılan öznitelikleri doğrudan etkiler. Tablo 4.5, tasarlanan CNN mimarisindeki farklı katmanların hiperparametrelerini göstermektedir. Şekil 4.4'te görüldüğü gibi, görsel öznitelik çıkarımı için 6 adet Konvolüsyon-ReLU-Paket Normalizasyonu yapısı arka arkaya bağlanmıştır. Bu işlemden sonra çıkarılan özniteliklerden konum tahminleri yapabilmek için bu öznitelikler Tam Bağlantılı Katmanlara (sırasıyla FC1, FC2 ve FC3) girdi olarak verilmiştir. Tablo 4.5, her katmanın hiperparametrelerini ve eğitim parametrelerini göstermektedir.

Tablo 4.5'te gösterilen FC1, FC2 ve FC3 katmanları, belirli sayıda öznitelik elde etmek ve bunlara göre konum değerlerini tahmin etmek için kullanılır. FC1 ve FC2, CNN katmanları aracılığıyla çıkarılan özelliklerin işlenmesine izin verir, bu nedenle çıktı boyutu olarak belirli sayılar (100, 50) belirlenir. x, y ve z konumlarının tahmin edildiği regresyon katmanından önceki katman olduğundan, FC3'ün çıkış boyutu üçtür. Ayrıca

hem FC katmanları hem de CNN katmanları için belirlenen hiperparametre değerleri deneme yanılma yoluyla belirlenir. FC1, görsel verilerden çıkarılan öznitelik vektörlerini atalet verilerinden çıkarılan öznitelik vektörleriyle birleştirmek için kullanılır. Bu katmandan alınan 100 görsel öznitelik, aynı anda elde edilen 100 atalet özniteliği ile birleştirilir. Derin ağ, Tablo 4.5'teki parametre değerleri kullanılarak SGDM algoritması ile eğitilmiştir.

Görsel-atalet (V-I) veri birleştirme bölümünde, hem atalet hem de görsel özellik çıkarma adımlarında bulunan FC1 katmanları, IMU sensöründen ve kamera karelerinden çıkarılan özellikleri birleştirmek için kullanılmıştır. Her bir kareden ve IMU dizisinden 100 özellik çıkarılmış, bu sayede her iki farklı sensör bilgisinden de eşit oranda faydalanılmıştır. Her iki FC1 katmanından elde edilen toplam 200 görsel-atalet (VI) verisi birleştirilmiş ve Şekil 4.4'te gösterildiği gibi bir sonraki BiLSTM katmanına girdi olarak verilmiştir. Füzyon edilen özellikleri işleyen mimariye ait hiperparametreler ve eğitim parametreleri Tablo 4.6'da gösterilmektedir. Hem katmanların hiperparametreleri hem de eğitim parametreleri Tablo 4.4'te gösterilen değerlere benzerdir. Füzyon edilen özellik vektörlerinin %85'i eğitim ve %15'i test olarak kullanılmıştır.

Tablo 4.6. Görsel-atalet özellik çıkarmada katmanlar için hiperparametreler ve eğitim parametreleri

BiLSTM			**FC1**		**FC2**	
Gizli Katman Sayısı	Aktivasyon Fonksiyonu	Kapı Aktivasyon Fonksiyonu	Çıkış Boyutu	Aktivasyon Fonksiyonu	Çıkış Boyutu	Aktivasyon Fonksiyonu
100	tanh	sigmoid	100	tanh	3	tanh
Training Parameters						
Optimizasyon Algoritması	$\boldsymbol{\beta_1}$	$\boldsymbol{\beta_2}$	Maks. Epok	Küme (Mini Batch) Boyutu	$\boldsymbol{\alpha}$	ϵ
Adam	0.9	0.999	250	100	0.001	10^{-8}

Tablo 4.4, Tablo 4.5 ve Tablo 4.6 incelendiğinde maksimum epok değerleri sırasıyla 250, 15 ve 250'dir. Bunun nedeni görsel özellik çıkarma adımının eğitim süresini azaltmaktır. Diğer BiLSTM kullanan atalet ve görsel-atalet adımlarında, özellikler nümerik verilerden çıkarıldığından ve ağ derinliği az olduğundan eğitim zaten hızlıdır.

4.2.2.2 EuRoC Veri Seti İle Elde Edilen Sonuçlar

EuRoC veri setindeki farklı veriler (V1-01, V1-02, V1-03, V2-01, V2-02 ve V2-03) kullanılarak elde edilen sonuçlar Tablo 4.7, Tablo 4.8 ve Tablo 4.9'da gösterilmiştir. Bu sonuçlar %15'lik test verisi ile elde edilmiştir. Tablo 4.7 sadece atalet verilerine, Tablo 4.8 sadece görsel verilere, Tablo 4.9 ise füzyon edilen görsel-atalet verilerine göre

hesaplanan hata değerlerini göstermektedir. Bu tablolar her bir eksendeki hata değerlerini ve ortalama hata değerlerini belirtmektedir.

Tablo 4.7. Atalet verilerine göre tahmini İHA konumlarının (x, y, z) RMSE değerleri

V1-01			**V1-02**			**V1-03**		
X	**Y**	**Z**	**X**	**Y**	**Z**	**X**	**Y**	**Z**
1.56	1.92	0.28	1.13	1.52	0.36	1.91	0.90	0.58
Ortalama=1.25			Ortalama =1.00			Ortalama =1.13		
V2-01			**V2-02**			**V2-03**		
X	**Y**	**Z**	**X**	**Y**	**Z**	**X**	**Y**	**Z**
2.75	2.46	0.38	2.69	1.21	0.47	1.67	1.24	0.32
Ortalama =1.86			Ortalama =1.45			Ortalama =1.07		

Tablo 4.8. Görsel verilere dayalı tahmin edilen İHA konumlarının $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ RMSE değerleri

V1-01			**V1-02**			**V1-03**		
X	**Y**	**Z**	**X**	**Y**	**Z**	**X**	**Y**	**Z**
1.38	1.46	0.21	1.15	1.12	0.32	0.9	1.36	0.50
Ortalama =1.01			Ortalama =0.86			Ortalama =0.92		
V2-01			**V2-02**			**V2-03**		
X	**Y**	**Z**	**X**	**Y**	**Z**	**X**	**Y**	**Z**
2.86	0.97	0.25	3.59	0.76	0.29	1.26	0.88	0.29
Ortalama =1.36			Ortalama =1.55			Ortalama =0.81		

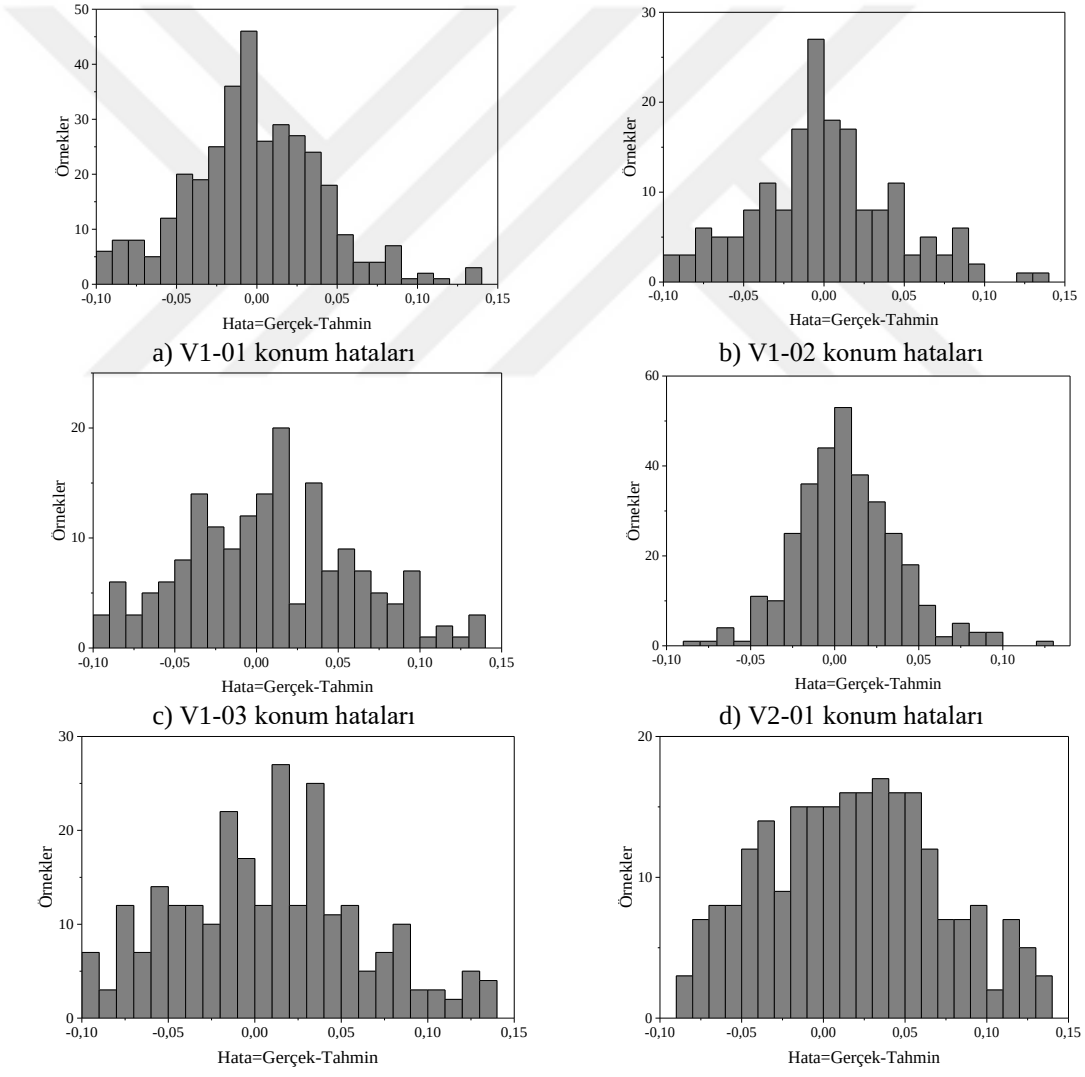
Tablo 4.9. Görsel-atalet verilerine göre tahmin edilen İHA konumlarının (x, y, z) RMSE değerleri

V1-01			**V1-02**			**V1-03**		
X	**Y**	**Z**	**X**	**Y**	**Z**	**X**	**Y**	**Z**
0.13	0.13	0.07	0.10	0.10	0.06	0.21	0.22	0.14
Ortalama = 0.11			Ortalama = 0.087			Ortalama = 0.19		
V2-01			**V2-02**			**V2-03**		
X	**Y**	**Z**	**X**	**Y**	**Z**	**X**	**Y**	**Z**
0.051	0.068	0.039	0.25	0.19	0.10	0.23	0.21	0.11
Ortalama = 0.0527			Ortalama = 0.183			Ortalama = 0.183		

(V1-01, V1-02, V1-03, V2-01, V2-02, V2-03, farklı ortam koşullarındaki verileri temsil eder .)

Tablo 4.9, önerilen VIO algoritmasının EuRoC veri seti üzerindeki sonuçlarını göstermektedir. EuRoC veri setindeki altı farklı verinin hata dağılımını göstermek için ortalama hata histogramları da Şekil 4.5'te verilmiştir. Hata histogramı sayesinde hedef değerler ile tahmin edilen değerler arasındaki fark daha net bir şekilde ifade edilir. Şekil 4.5 incelendiğinde en başarılı tahminin V2-01 verileriyle yapıldığı, V1-03 verilerinde tahmin hatasının daha yüksek olduğu görülmektedir. Ancak genel olarak hata oranları çok düşüktür ve hata değerleri (Gerçek –Tahmin) sürekli sıfır noktasına etrafında toplanır. Ek olarak Tablo 4.10, önerilen hibrit VIO çalışmasının EuRoC veri seti sonuçlarını önceki çalışmalarla karşılaştırmaktadır.

Tablo 4.10’da elde edilen sonuçlar, bu uygulamada geçmiş çalışmalarla karşılaştırılabilir düzeyde başarılı sonuçlar elde edildiğini göstermektedir. Tablo 4.10'dan da görülebileceği gibi, geçmiş çalışmalar genellikle V(I)SLAM ve V(I)O uygulamaları için genellikle ya filtre ya da optimizasyon tabanlı yöntemleri benimsemişlerdir. Ancak, her iki yöntemi karşılaştıran bir çalışmada, Chen ve ark. (2018), optimizasyon tabanlı ve filtre tabanlı yöntemlerde V(I)SLAM ve V(I)O için kullanılan görüntü özelliklerinin anlamsal düzeyinin düşük olduğunu belirtmiş ve derin öğrenme tabanlı yöntemlerin gelecek V(I)SLAM ve V(I)O çalışmalarına yön vereceğini belirtmişlerdir. Bu kapsamda hem uygulama kolaylığı, hem modern olması hem de geleneksel yöntemlerin eksiklerini gidermesi açısından derin öğrenme tabanlı çözümler günümüzde popülerdir.



a) V1-01 konum hataları b) V1-02 konum hataları

c) V1-03 konum hataları d) V2-01 konum hataları

e) V2-02 konum hataları f) V2-03 konum hataları

Şekil 4.5. Tahmin edilen konum değerlerinin hata histogramları

Tablo 4.10. Önerilen çalışmanın önceki EuRoC veri seti kullanan çalışmalarla karşılaştırılması

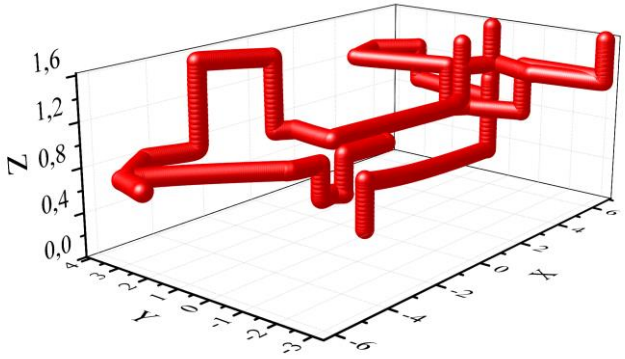
Önceki Çalışma	Algoritma	V1-01	V1-02	V1-03	V2-01	V2-02	V2-03
SVOMSF (Kaiser ve ark., 2016)	VIO (Yarı Doğrudan - Filtre Tabanlı)	0.400	0.630	x	0.200	0.370	x
MSCKF (Mourikis ve Roumeliotis, 2007)	VIO (Filtre Tabanlı)	0.340	0.200	0.670	0.100	0.160	1.130
OKVIS (Leutenegger ve ark., 2015)	VISLAM (Ana Kare Tabanlı)	0.090	0.200	0.240	0.130	0.160	0.290
ROVIO (Bloesch ve ark., 2015)	VIO (Filtre Tabanlı)	0.100	0.100	0.140	0.120	0.140	0.140
VINS-MONO (Qin ve ark., 2018)	VIO (Optimizasyon Tabanlı)	0.070	0.100	0.130	0.080	0.080	0.210
VINS-MONO-LC (Lynen ve ark., 2013)	VIO (Optimizasyon Tabanlı)	0.040	0.060	0.110	0.060	0.060	0.090
SVOGTSAM (Forster ve ark., 2016a)	VIO (Semi Direct - Optimizasyon Tabanlı)	0.070	0.110	x	0.070	x	x
ORB-SLAM2 (Mur-Artal ve Tardós, 2017b)	VSLAM (Özellik Tabanlı)	0.087	0.065	0.092	0.071	0.061	0.184
STCM-SLAM (Chen ve ark., 2019)	VISLAM (Ana Kare Tabanlı)	0.079	0.113	0.120	0.084	0.060	0.099
SELFVIO (Almalioglu ve ark., 2019)	DL	0.080	0.090	0.100	0.110	0.080	0.110
KIMERA (Rosinol ve ark., 2020)	VISLAM (Metrik - Semantik)	0.050	0.080	0.070	0.080	0.100	0.210
Li ve Waslander (2020)	DL	2.070	2.200	2.830	1.490	1.490	2.220
Bizim Metodumuz	**CNN+BiLSTM**	**0.110**	**0.087**	**0.190**	**0.053**	**0.183**	**0.183**

(V1-01, V1-02, V1-03, V2-01, V2-02, V2-03 farklı ortam koşullarındaki verileri temsil eder .)

4.2.3. Simülasyon Sonuçları

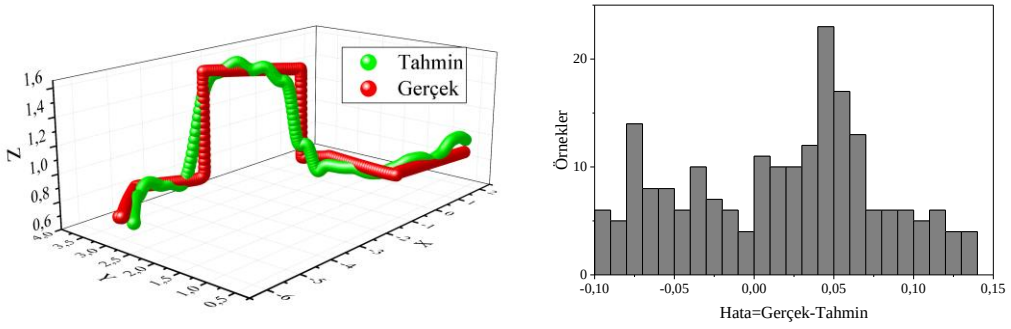
EuRoC ile elde edilen başarılı sonuçlara dayalı olarak tasarlanan mimari, ROS ortamında oluşturulan yeni İHA verilerine uygulanmıştır. Bu bölüm simülasyon veri seti sonuçlarını değerlendirmektedir.

Tello'nun simülasyon ortamında gerçekleştirdiği 3B gezinme hareketi Şekil 4.6'da gösterilmektedir. Tello $z = 0$ noktasında hareket etmeye başlamış ve toplamda 2432 karelik bir video kaydetmiştir. IMU verileri video ile eş zamanlı olarak kaydedilmiştir. EuRoC veri setinde olduğu gibi IMU verileri, gerçek konum değerleri ve karelere ait zaman damgası değerleri ile birlikte kaydedilmiştir. Şekil 4.6'daki İHA konumları, her bir karedeki gerçek konum değerlerini göstermektedir. Hareket yörüngesi incelendiğinde, yön değiştirme anında hareketin keskin (ani) olduğu görülmektedir. Bu tür verilerde yön değişikliği anındaki konumu tahmin etmek nispeten daha zordur.



Şekil 4.6. Simülasyonda İHA'nin hareket yörüngesi

Gazebo simülasyon ortamında İHA ile oluşturulan IMU ve görsel veriler EuRoC verileri ile benzerlik göstermektedir. EuRoC veri seti için uygulanan üç aşamalı hibrit öğrenme stratejisi (BiLSTM-CNN-BiLSTM) (bkz. Şekil 4.4) doğrudan simülasyon veri setimize uygulanmaktadır. İHA konum tahmini için IMU, görsel ve hibrit verilerin %85'i eğitim için ayrılmıştır. Eğitim adımından sonra test verileri ile yapılan tahmin adımında x, y ve z konumlarının RMSE değerleri sırasıyla 0.314, 0.14 ve 0.05 olarak hesaplanmıştır. Sonuç olarak, test verilerindeki ortalama RMSE hata değeri 0.167'dir. Şekil 4.7 simülasyon veri seti ile elde edilen sonuçları göstermektedir. Şekil 4.7 (a) test verilerinin gerçek konumlarını ve önerilen algoritma tarafından tahmin edilen konumları göstermektedir. Gerçek verilerdeki keskin yön değişiklikleri, tahmin verilerinde sapmalara neden olur. Ancak, test verisi tahminindeki hata oldukça kabul edilebilir düzeydedir. Şekil 4.7 (a)'daki x, y ve z koordinatlarındaki ortalama hata histogramı Şekil 4.8'de gösterilmiştir. Buna göre, gerçek değerler ile tahmin edilen değerler arasındaki fark genellikle 0 noktası etrafında dağılmıştır. Bu da önerilen yöntemin veri setimiz üzerinde başarılı tahminler yaptığını kanıtlamaktadır.



(a) Gerçek ve tahmin edilen İHA yörüngesi

(b) Simülasyon ortamında tahmin edilen konum değerlerinin hata histogramı

Şekil 4.7. Önerilen yöntemin simülasyon veri seti üzerindeki performansı

4.3. UYGULAMA 2: Görsel-Atalet Görüntü-Odometri (VIIONet): İHA Poz Tahmini için Gauss Süreci Regresyonu Tabanlı Derin Mimari Önerisi

Şimdiye kadarki derin öğrenme tabanlı çalışmalar genellikle görsel ve atalet bilgilerin füzyonu için CNN-RNN tabanlı mimarileri tercih etti. IMU verileri arasındaki zamansal bağımlılık, yazarları bu bağımlılığı dikkate alan LSTM katmanını kullanmaya yöneltti. Bu nedenle önerilen derin öğrenmeye dayalı önceki VIO/VISLAM çalışmalarına ait mimariler benzer özelliktedir. Bu yöntemler her ne kadar başarılı sonuçlar gösterse de, LSTM'in sınırlı tahmin yeteneği popüler CNN modellerini daha tercih edilebilir kılmaktadır. Weytjens ve De Weerdt (2020) yaptığı bir kıyaslama çalışmasında, CNN'lerin LSTM'lere göre daha sağlam sonuçlar verdiğini ve CNN ile LSTM'e benzer sonuçların daha kısa sürede elde edildiğini belirtti.

Özellikle son zamanlarda çokça kullanılan transfer öğrenme yapıları birçok uygulama için hem kolaylık sağlamış hem de sonuçlara katkı sağlamıştır. Bu sayede araştırmacılar önceden eğitilmiş ya da önceden tasarlanmış AlexNet (Krizhevsky ve ark., 2012), VGGNet (Simonyan ve Zisserman, 2014), GoogleNet (Szegedy ve ark., 2015), Inception (Szegedy ve ark., 2016), Xception (Chollet, 2017), ResNet (He ve ark., 2016), vb. mimarileri doğrudan kendi çalışmaları için kullanmaktadır. Transfer öğrenme, özellikle nesne tanıma, sınıflandırma gibi çalışmalarda günümüzde sıklıkla uygulanmaktadır (Aslan ve ark., 2021b). Ancak IMU gibi 1B veriler popüler CNN modelleri için uygun değildir. Çünkü bu modeller genellikle 2B veya 3B görüntüyü giriş olarak alır. Bu nedenle doğrudan IMU verileri popüler CNN modellerine giriş olarak uygulanamaz. Bizim çalışmamız IMU verilerini görüntülere dönüştürerek Inception-v3 transfer öğrenme içeren bir atalet tabanlı konum tahmini gerçekleştirmektedir. Görüntüye dönüştürme öncesinde IMU verilerinin gürültüsü giderilir. Çünkü gürültü, tahmin performansını olumsuz etkiler. Brossard ve ark. (2020) IMU verilerinin gürültüye ve ön bilgiye (bias) sahip olduklarını ifade etmiştir. Daha sonra IMU jiroskop ölçümlerinin gürültüsünü kaldırarak konum tahmini gerçekleştirmiştir. Cao ve ark. (2020) ise VSLAM uygulamaları için gürültülü görüntülerin SLAM performansını azalttığını kanıtladı. Bizim çalışmamızda da IMU verilerinin gürültüsü Savitzky–Golay (Savitzky ve Golay, 1964; Schafer, 2011) yumuşatma (smoothing) filtresi ile azaltıldı.

Genel olarak açıklamak gerekirse, bu uygulama görsel ve atalet bilgilerini kullanarak bir hava aracının poz tahminini (konum ve yönelim) (VIO) gerçekleştirmeyi amaçlamıştır. Önerilen yöntem EuRoC (Burri ve ark., 2016) ve kendi veri setimiz kullanılarak uygulanmıştır. Önerilen yöntem yeni bir mimariyi tasarlamakta ve, görsel ve

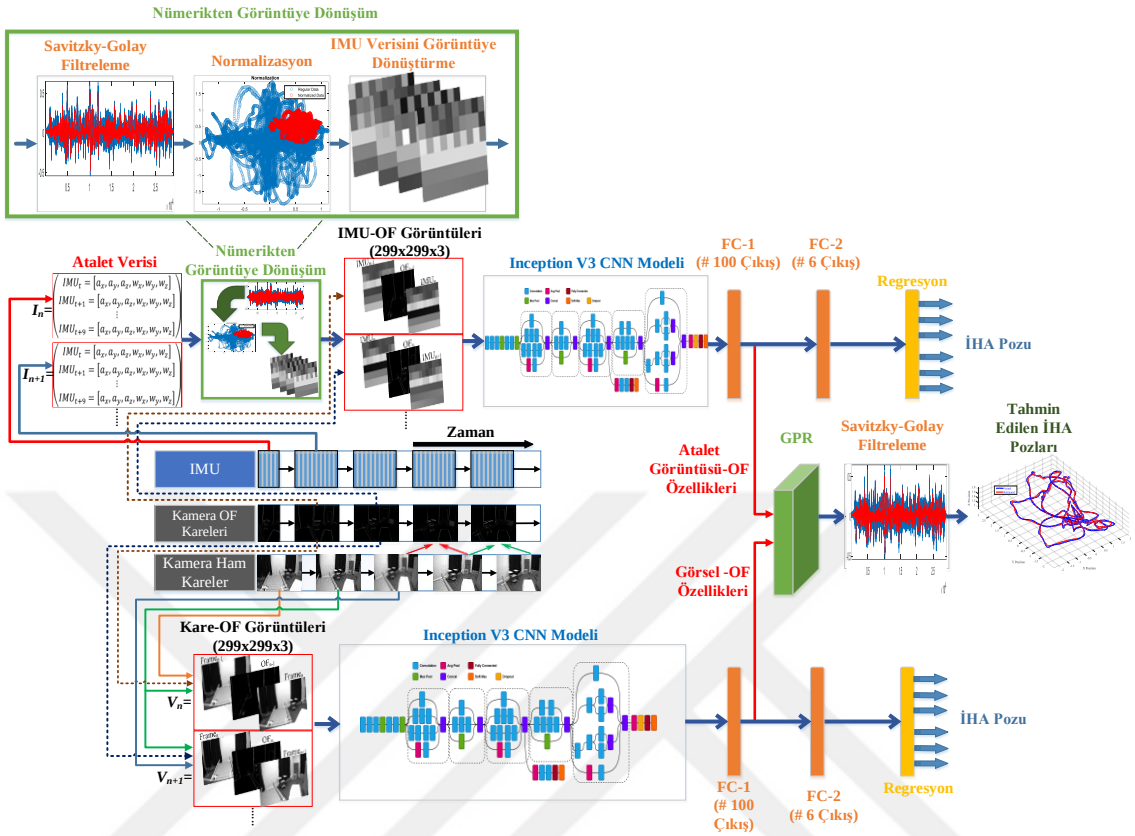
atalet bilgilerini şimdiye kadarki yöntemlerden farklı şekilde işlemektedir. Çalışma üç adımda ele alınabilir: görsel adım, atalet adımı ve füzyon adımı. İlk adımda ardışık iki ham gri uzaylı görüntüler ile OF genlik görüntüleri birleştirilerek, ince ayarlı (fine-tuning) Inception-v3 modeline giriş olarak verilir, sonuçta poz tahmini sağlanır. Atalet adımında ise numerik ham IMU verileri görüntüye dönüştürülür. Bunun için önce ardışık iki kare arasındaki IMU verilerinin gürültüsü Savitzky–Golay filtre ile azaltılır, daha sonra normalizasyon uygulanır. Normalizasyon adımından sonra görüntüye dönüştürülen ardışık iki atalet görüntüsü verisi, ardışık iki karenin OF genlik görüntüsü ile birleştirilir. İlk adımda olduğu gibi ikinci adımda da Inception-v3 ile poz tahmini sağlanır. Görsel ve atalet adımlarındaki özelliklerin birleşimi için, FC katmanlarındaki özellikler kullanılır ve bu füzyon edilen özellikler, son olarak Gauss Süreç Regresyonu (Gaussian Process Regression (GPR)) (Williams ve Rasmussen, 2006; Le Gentil ve ark., 2020) aracılıyla İHA pozlarına dönüştürülür. GPR'nin çıktısındaki kararsızlıkların azaltılması için tekrar bir Savitzky–Golay filtresi uygulanır ve nihai poz tahmini gerçekleştirilir. Bu çalışmanın önceki çalışmalara göre katkıları aşağıdaki gibi özetlenebilir.

a) OF-Atalet-Görsel özelliklere dayalı poz tahmini gerçekleştiren bir derin mimari tasarlanmıştır.
b) Görsel-Atalet Görüntü Ağı (Visual-Inertial Image-Odometry (VIIONet)) adında yeni bir strateji ile İHA poz tahmini gerçekleştirildi.
c) IMU verilerinin görüntüye dönüştürülmesi, popüler CNN modellerinin atalet tabanlı poz tahminlerinde kullanılabilirliğini sağladı.
d) IMU verilerinin gürültüsü Savitzky–Golay filter ile azaltıldı.
e) Inception-v3 ile çıkarılan özellikler GPR'a giriş olarak verilerek nihai poz tahmini sağlandı.
f) Önerilen yöntem önceki çalışmaların çoğuna üstünlük sağladı.

4.3.1. Önerilen Mimari

Bu çalışmada önerilen pratik uygulamada, öncelikle ham karelerden OF alanı görüntüleri elde edilmiştir. Daha sonra İHA ile VIO için, görsel ve atalet bilgileri üç farklı adımda değerlendirilir. Bunlar, kullanılan veri setlerindeki ham görüntüleri kullanan görsel adım, IMU verilerini görsel verilerle birleştiren atalet görüntüsü adımı ve son olarak her iki bilgiyi birleştiren füzyon adımıdır. Şekil 4.8 detaylı olarak önerilen çalışmanın adımlarını gösterir. İlk olarak OF görüntülerinin elde edilmesi anlatıldıktan

sonra, diğer her bir adıma ait detaylı bilgiler Şekil 4.8'e bağlı olarak alt başlıklarda ifade edilmiştir.



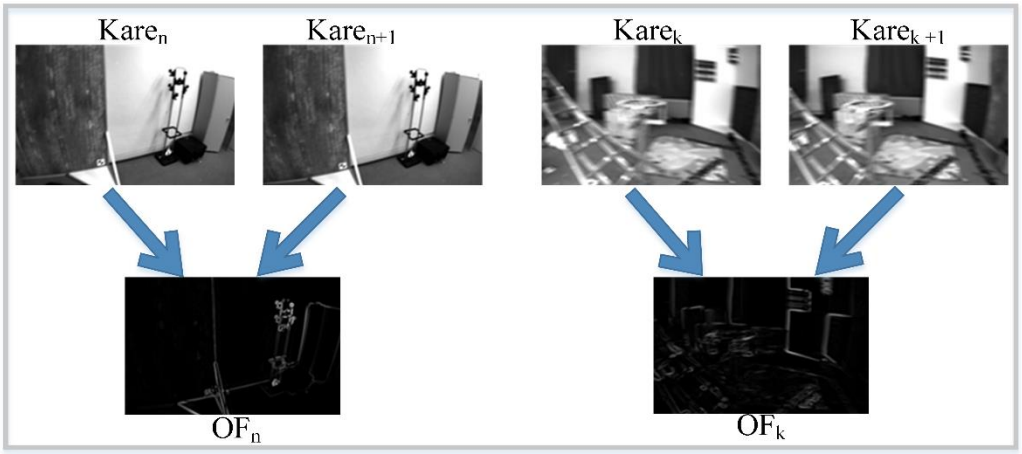
Şekil 4.8. Önerilen yaklaşımın mimarisi

4.3.1.1 Ham Görüntülerden OF Görüntüleri Elde Etme

Derin Öğrenme tabanlı VIO çalışmaları genellikle ham görüntülerden tüm poz bilgisini öğrenmeyi amaçlar. Ancak sadece ham görüntülere bağlı olarak yapılan poz tahmini çalışmaları her ne kadar umut verici sonuçlar üretse de, aynı çözümleri farklı ortamlarda aynı başarı ile uygulamak zordur. Yani eğitim verisinden farklı olan sahne ve hareket dinamiklerinde tahmin performansı çok azalır. Bu nedenle derin ağın görsel ve hareket bilgisini ayırt edebilmesi ve öğrenmesi gerekir (Costante ve Ciarfuglia, 2018). Bu bağlamda, nesne ve kamera arasındaki göreceli harekete bağlı olarak, hareket dinamiklerini ifade eden OF alanları VIO tahminlerine büyük katkı sağlar. Bu nedenle OF ile elde edilen hareket ipuçları otonom amaçlı çalışmalarda çokça tercih edilir (Jiao ve ark., 2021). Ardışık kareler arasındaki harekete göre hesaplanan OF alanı, görüntünün yatay ve düşey hareket bilgisini ve bunların büyüklüğünü içermektedir.

Bu çalışmada da hareket tahminine OF alan görüntülerinin katkısını sağlamak için ardışık iki kare arasındaki OF genlik görüntüleri kaydedilir. Daha sonra bu iki kare arasındaki OF görüntüleri, hem görsel hem de atalet adımında kullanılır. Bu sayede ham

kamera ve IMU verilerinin hareket bilgisi ile beslenmesi sağlanır. Şekil 4.9'da ardışık karelerden elde edilen bazı OF alanları gösterilmiştir.



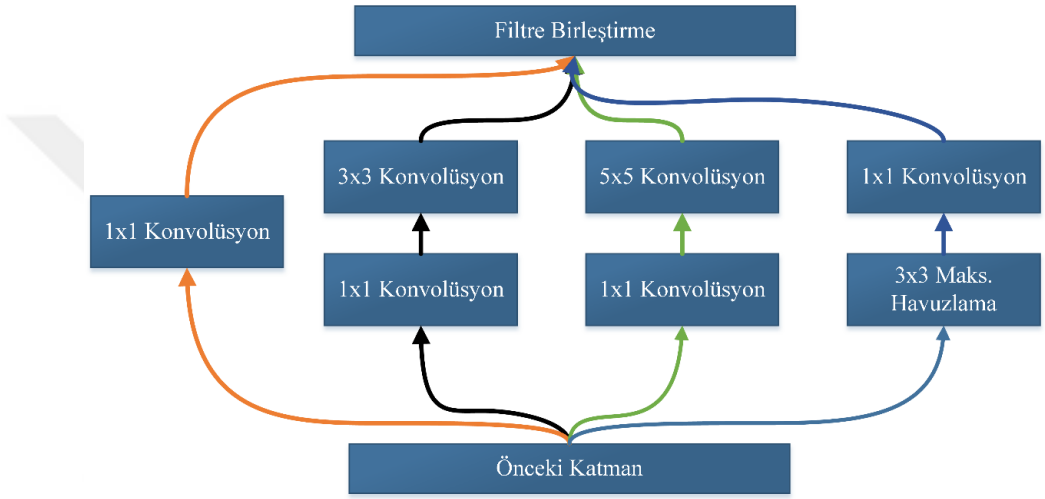
Şekil 4.9. Bazı ardışık ham karelerden OF görüntüsünün oluşturulması

4.3.1.2 Görsel Veri Tabanlı VIO

Görsel adımdaki genel odak, derin ağın genelleştirme ve ölçek kayması gibi unsurları sağlam bir şekilde öğrenebilmesidir. Bu uygulamada kullanılan Inception-v3, ağ parametrelerinin miktarını azaltan ve aynı zamanda ağ derinliğini de artıran bir mimariye sahiptir. Bu nedenle çok ilgi çekmiştir, ve Inception-V1, Inception-V2, Inception-V4 gibi sürümleri de vardır (Dong ve ark., 2020). Inception-v3 hem doğruluk hem de genelleştirme yeteneği açısından diğer CNN modellerinin çoğuna üstünlük sağlar. Inception-v3'ü de içeren farklı CNN modelleri, Canziani ve ark. (2016) tarafından yapılan çalışmada farklı açılardan karşılaştırılmıştır. Karşılaştırmalar dikkate alındığında, Inception-v3'ün çoğu modern CNN modeline farklı açılardan üstünlük sağladığı gösterilmiştir. Bu anlamda, bu çalışmada poz tahmini açısından tatmin edici bir doğruluk sağlanması ve sağlam bir genelleştirme becerisi için Inception-V3 tercih edilmiştir. Ölçek kayma probleminin hafifletilmesi için, Inception-V3'ün aynı zamanda iki kare arasındaki hareket değişimlerini içeren OF görüntülerinden beslenmesi sağlanmıştır. Bu durumda hem genelleştirme yeteneği yüksek hem de ölçek kayması minimum seviyede tahminler elde edileceği beklenmiştir.

Şekil 4.8'in alt kısmında yer alan ve ham İHA kerelerini OF görüntüleri ile birleştirdikten sonra poz tahmini gerçekleştiren kısım görsel tabanlı VIO'dur. Görsel adımdaki amaç, geleneksel yöntemlerden farklı olarak, iki ardışık kare arasına, bu iki kare arasındaki hareket bilgilerini belirten OF bilgilerini ilave etmektir. Bunu senkronize bir şekilde sağlamak için n zamanındaki bir poz tahmini için (V_n), sırasıyla $Kare_{n-1}$, OF_{n-1}, $Kare_n$ görüntüleri birleştirilir. Buradaki OF_{n-1}, $Kare_{n-1}$ ve $Kare_n$ arasındaki hareket

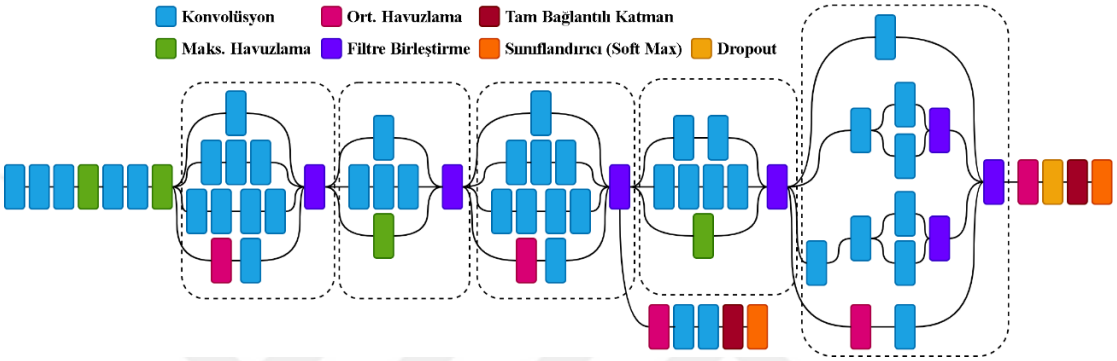
değişiminden kaynaklanan alanlardaki değişimin büyüklüğünü ifade etmektedir. Bu sayede derin öğrenme aracılığıyla hareket değişimine daha duyarlı bir ağ eğitimi sağlanabilir. Üç adet görüntünün bir araya getirilmesi üç kanallı bir görüntü oluşturulur. Bu görüntü boyutları Inception-V3 mimarisi girişi için 299×299 olarak yeniden boyutlandırılır. $299 \times 299 \times 3$ olarak boyutlandırılan her 3B görüntü artık görsel tabanlı VIO için yeni bir veri setini oluşturur. Sonrasında bu 3B görüntüler Inception-v3'e giriş olarak, ve poz değerleri ise Inception-v3'e çıkış olarak verilmek şartıyla eğitim gerçekleştirilir.



Şekil 4.10. Inception modülünün yapısı

Bu çalışmada hem görsel hem de atalet adımında uygulanan Inception mimarisi, yani Inception-v3, derinliği artırıp parametreyi azaltmak için Inception yapılarını ağ içinde kullanır. Bir Inception ağı, Inception modüllerinin birleşiminden oluşur, GoogleNet (Inception-v1) olarak da adlandırılır. Inception modülü üç farklı boyutta konvolüsyon ((1×1), (3×3), (5×5)) ve 3×3 boyutunda bir maksimum havuzlama içermektedir. Tüm bu işlemler paralel olarak uygulanmaktadır. Darboğaz katmanı (Bottleneck layer) olarak adlandırılan 1×1 konvolüsyon katmanları sayesinde işlem karmaşıklığının ve parametre sayısının azaltılması sağlanır. Şekil 4.10'daki bir Inception modülünde, bu 1x1 konvolüsyon katmanlarının kullanımı gösterilir. Her 1×1 konvolüsyon katmanları sonrasında aktivasyon fonksiyonu olarak ReLU kullanılır ve bu sayede ağa daha derin bir doğrusal olmama durumu (nonlineer) sağlanır. Inception-V3 mimarisinde FC katmanlar yerine ortalama havuzlama kullanılmaktadır. Bu parametre sayısını önemli ölçüde azaltmaktadır. Sonuçta Inception-V3 ağının daha derin, VGG modelinden daha hızlı ve AlexNet'ten daha az parametreye sahip olmasını sağlanır (KIZRAK ve BOLAT, 2018; Mahdianpari ve ark., 2018; Dong ve ark., 2020).

İlk Inception (Inception-v1 (GoogleNet)) çalışmasından sonra, sonraki iki versiyon (Inception-v2, Inception-v3) aynı çalışmada (Szegedy ve ark., 2016) tanıtılmıştır. İlk versiyondan sonra önerilen yöntemler optimizasyon ve konvolüsyon ağlarının daha efektif kullanımını sağladı. Örneğin Inception-v2'de, 5x5 konvolüsyonlar iki ardışık 3x3 konvolüsyonlarla değiştirilmiştir. Inception-V3'de ise için yardımcı sınıflayıcı olarak paket normalizasyonu (batch-normalized) ve FC katmanı eklenmiştir. Bu çalışmada kullanılan Inception-v3'e ait mimari Şekil 4.11'de verilmiştir.



Şekil 4.11. Inception V3 yapısı (Ding ve ark., 2019)

Inception-V3'ün orijinali 1000 sınıflı bir çıkış sağlamaktadır. Bu mevcut ağ ile bir poz regresyonu sağlamak için mevcut son katmanlarda bir değişiklik yapılmalıdır. Bunun için Inception-V3'ün son üç katmanı çıkarılır ve yerine iki adet FC ve bir adet regresyon katmanı eklenir. Eklenen FC katmanlarına ait parametre bilgileri Tablo 4.11'de gösterilir. FC-1 katmanı füzyon için gerekli 100 adet özelliği çıkarmaktadır. FC-2 ise regresyon katmanına verilecek 6DOF bilgileri elde etmek için 6 çıkışa sahiptir (bkz. Şekil 4.8). Katmanlar eklendikten sonra artık eğitim parametreleri belirlenerek eğitim başlatılır. Hedef olarak belirlenen poz gerçek değerlerine göre eğitim değerlerini optimize etmek için Denklem 4.5'te parametre güncellemesi eşitliği verilen SGDM algoritması kullanılmıştır.

Tablo 4.11. Görsel özellik çıkarımı için kullanılan CNN ve FC katmanlarının eğitim parametreleri

FC Katmanları				
Katman Adı	**Çıkış Boyutu**	**Aktivasyon Fonksiyonu**	**ReLU**	**Drop Out**
FC-1	100	$tanh$	Var	Yok
FC-2	6	$tanh$	Yok	Yok
Eğitim Parametreleri				
Optimizasyon Alg.	**Maksimum Epok**	**Küme (Mini Batch) Boyutu**	$\boldsymbol{\alpha}$	$\boldsymbol{\gamma}$
SGDM	15	16	0.001	0.95

4.3.1.3 Atalet Verisine Dayalı VIO

Bu adımda görsel adımdaki işlemlere benzer olarak Inception-V3, özellik çıkarma, regresyon, vb. kısımlar bulunur. Şekil 4.8'de de görüldüğü gibi, aynı olan bu kısımlar üzerinde bu bölümde tekrar durulmayacaktır. Aynı olan kısım atalet görüntüsü üretildikten sonraki kısımlardır ve FC katmanları da görsel olan kısımlarla aynı özelliktedir. Sadece, farklı olarak IMU görüntülerin eğitimi için kullanılan eğitim parametrelerinde maksimum epok sayısı 60'tır. Bu adımdaki farklı olan kısım IMU verilerinin görüntüye dönüştürülmesi ve OF görüntülerle birlikte Inception-v3'e verilmesidir.

Bilindiği gibi IMU değerleri tamamen iç hareket dinamiğini (ego-motion) yansıtır. Harekete bağlı olarak, herhangi bir t anında daha önce Denklem 4.1'de belirtildiği gibi 3 boyutlu lineer (a_x, a_y, a_z) ve açısal ivme (w_x, w_y, w_z) değerlerini çıktı olarak verir. Harekete karşı oldukça hassas olan bu sensörler, oldukça yüksek frekansta çalışır. Bizim çalışmamızda kullanılan EuRoC veri seti için IMU frekansı 200Hz idi, ve kendi veri setimiz için bu değer 300Hz idi. Ancak IMU çok daha yüksek frekans değerlerine sahip olabilir. Her iki veri setinde de IMU frekansı kamera frekansının 10 katıdır. Bu nedenle herhangi iki kamera karesi arasında 10 adet IMU bilgisi bulunur.

Bu bölümde amaç kamera kareleri arasındaki numerik değerlere sahip IMU değerlerini poz tahmini için kullanmaktır. Önceki çalışmalar genellikle bu zamansal bağımlılığa sahip IMU'ları doğrudan LSTM katmanlarına verirler ve sonuçta zamansal özellikleri elde ederler. Bu çalışma IMU verilerinin özelliklerini diğer çalışmalardan farklı bir şekilde çıkarmaktadır. Geleneksel yöntemlerdeki gibi RNN tabanlı LSTM kullanmak yerine, CNN modellerinin sağlam tahmin yeteneğine bağlı olarak IMU'lardan daha çok faydalanmak amaçlanır. Bunu yapabilmek için, genellikle 3B görüntü girişini kabul eden CNN modelleri referans alınır. IMU verisi 1B yapıdadır, ve örneğin Inception-v3 ağına doğrudan girdi olarak uygulanamaz. IMU bilgisinin 6 boyutlu olması ve frekansının kameraya göre yüksek olması bir avantajdır. Bu nedenle iki kare arasındaki IMU bilgileri 10×6 boyutunda olmak üzere Denklem 4.7'deki gibi bir araya getirilebilir. Bu durumda iki kare arasındaki atalet bilgisi, I_n olarak 2 boyutlu ifade edilebilir. Bu, numerik IMU bilgilerinin, görüntü olarak ifade edilebileceğini göstermektedir.

$$I_n = \begin{bmatrix} IMU_t \\ IMU_{t+1} \\ IMU_{t+2} \\ \vdots \\ IMU_{t+9} \end{bmatrix} = \begin{bmatrix} a_{x_t}, a_{y_t}, a_{z_t}, w_{x_t}, w_{y_t}, w_{z_t} \\ a_{x_{t+1}}, a_{y_{t+1}}, a_{z_{t+1}}, w_{x_{t+1}}, w_{y_{t+1}}, w_{z_{t+1}} \\ a_{x_{t+2}}, a_{y_{t+2}}, a_{z_{t+2}}, w_{x_{t+2}}, w_{y_{t+2}}, w_{z_{t+2}} \\ \vdots \\ a_{x_{t+9}}, a_{y_{t+9}}, a_{z_{t+9}}, w_{x_{t+9}}, w_{y_{t+9}}, w_{z_{t+9}} \end{bmatrix} \quad (4.7)$$

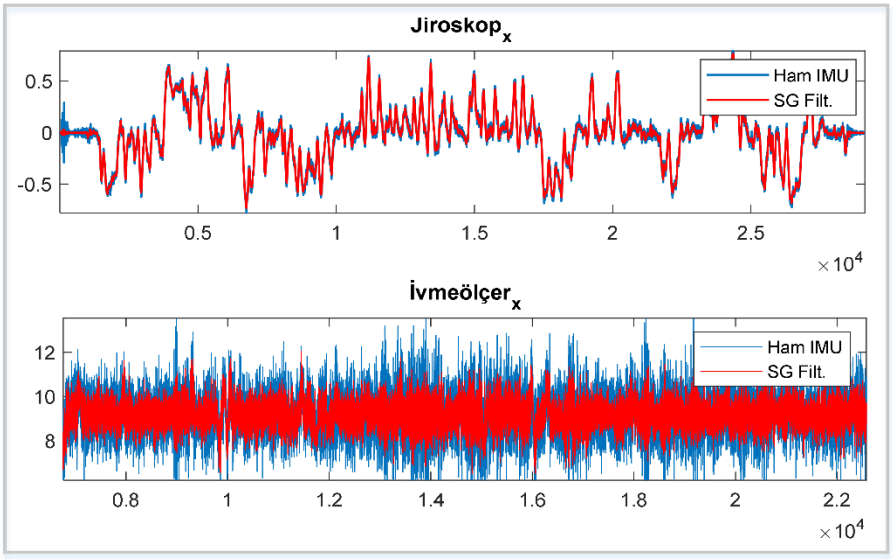
Şimdiye kadarki çalışmalar genel olarak IMU bilgilerini doğrudan kullanmıştır. Ancak harekete çok hassas olma, beraberinde gürültüye karşı hassasiyete de neden olur. IMU'nun çok gürültülü ölçümler aldığı ve bu nedenle öncelikle bir yumuşatma ya da bir gürültü giderme işleminin gereksinimi literatürde bilinen bir durumdur. Dolayısıyla literatürdeki (Chiang ve ark., 2004; Karaim ve ark., 2019; Khaddour ve ark., 2021; Mishra ve ark., 2021) gibi çok sayıdaki çalışma, IMU sensörü için öncelikle gürültüleri azaltmaya ihtiyaç duymuştur. Cao ve ark. (2020) gürültülü görüntülerin VSLAM performansını azalttığını ortaya koydu. Benzer mantıkla gürültülü IMU ölçümleri de VIO çalışmaları için performansı azaltacaktır. Bu nedenle bu çalışma IMU verilerini görüntüye dönüştürmeden önce, gürültü kaldırma işlemini gerçekleştirir. Bunun için genellikle kullanılan yöntemler dalgacık tabanlı yöntemler olmasına rağmen, gerçek zamanlı uygulamalar için hesaplama maliyeti daha az olan yumuşatma filtreleri gereklidir. Karaim ve ark. (2019) Savitzky-Golay (Savitzky ve Golay, 1964; Schafer, 2011) filtreyi IMU sensörü için kullandı ve bu filtrenin geleneksel gürültü giderme yöntemlerinden daha hızlı ve doğru sonuçlar verdiğini belirtti. Bizim çalışmamızda da IMU verisi gürültüleri Savitzky-Golay filtresi ile kaldırılmıştır.

Savitzky-Golay, bir polinomu en küçük kareler yöntemiyle bir dizi girdi örneğine uyduran, en düşük kareler (least-squares) yumuşatma yaklaşımına dayanan veri yumuşatma tekniğidir. Bu filtre ile sinyalin şekli bozulmadan verilerin kesinliğini artırmak amacıyla gürültüler azaltılır. Belirli bir veri penceresi boyutu (örneğin $2L + 1$) için, optimize edilmiş bir eğri uydurma polinomu, en küçük kareler çözümüyle elde edilebilir. $x[n]$, bir dizi veri örneği olsun. Her veri noktasının etrafındaki bir grup ($2L + 1$) örnekle, polinom uydurma gerçekleştirilebilir. Uydurma polinomu ve bu uydurulan değerler ile minimum hatayı sağlamak için Denklem 4.8 ve Denklem 4.9 kullanılır. Burada $(a_0, a_1, a_2, ...)$ uydurma katsayılardır ve N polinom sırasıdır. Denklem 4.9 ile $p(n)$'in ham değerlere göre hatası (e) minimize edilir (Karaim ve ark., 2019).

$$p(n) = \sum_{j=0}^{N} a_j n^j \tag{4.8}$$

$$e = \sum_{n=-L}^{L} (p(n) - x[n])^2 \tag{4.9}$$

Bu çalışmada IMU'ya ait ham ivmeölçer ve jiroskop değerleri Savitzky-Golay fitresinden geçirilir. Bir açısal ve bir doğrusal ivmenin x değerine ait IMU verilerinin Savitzky-Golay filtresi ile yumuşatılması ve gürültülerin azaltılması Şekil 4.12'de gösterilmiştir. Mavi çizgiler ham sinyali gösterirken, kırmızı çizgiler Savitzky-Golay (SG) ile gerçekleştirilen yumuşatılmış sinyali göstermektedir.



Şekil 4.12. Ham IMU verileri üzerine Savitzky-Golay filtresinin uygulanması

Gürültülerin giderilmesi aşamasından sonra, gürültüsü azaltılmış IMU verileri, Şekil 4.8'de görüldüğü gibi, normalize edilmiştir. Bunun sebebi Şekil 4.12'de de görüldüğü gibi IMU içindeki ivmeölçer ve jiroskop değerlerinin farklı alt ve üst limitlere sahip olmasıdır. Zaten normalizasyon, farklı özellik verilerindeki sınır değerlerinin çok farklı olması durumunda, yapay zekâ yöntemlerinde ağ eğitiminde başarıyı artırmak için sıklıkla kullanılır. Bizim çalışmamız hem bu avantajdan faydalanmakta hem de normalizasyonu sayısal verileri görüntüye dönüştürmede kullanmaktadır. Görüntüye dönüştürme işlemi oldukça basittir. Görüntüye dönüştürürken her bir IMU değerinin 0-1 arasında bir değere sahip olması istenmektedir. Normalizasyon aracılığıyla bu sağlanır. Denklem 4.7'deki gürültüsü Savitzky-Golay ile azaltılmış I_n verisi, özellik ölçekleme (feature scaling) metodu aracılığıyla normalize edilir. Literatürde farklı özellik ölçekleme yöntemleri vardır. Bizim uygulamamızda tercih edilen yöntem min-maks

normalleştirmedir. Bu normalleştirme yöntemi Denklem 4.10 ile gerçekleştirilir. Bu yöntem ile her bir özelliğe ait x değerleri, o özellik içindeki maksimum (x_{max}) ve minimum (x_{min}) değerler kullanılarak normalize edilir ($\hat{x}$).

$$\hat{x} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{4.10}$$

Normalizasyon sonucunda artık 0-1 arasında olan gürültüsü giderilmiş IMU 2B verileri (I_n), 255 değeri ile çarpılarak tek kanallı bir gri uzaylı görüntü (0-255) elde edilir. Toplam 6x10'luk 60 adet pikselden oluşan bu görüntüdeki her piksel, ilgili ivmeölçer ya da jiroskop değerinin büyüklüğünü belirten bir renge (parlaklığa) sahiptir. Her bir I_n IMU verisi artık 2B bir gri uzay görüntüsüdür. IMU verisinden görüntü oluşturma, yani Şekil 4.8'teki *nümerikten görüntüye dönüşüm* adımı Denklem 4.11'deki gibi formüle edilebilir. Bu adım sonucunda elde edilen bazı IMU görüntüleri de Şekil 4.13'te gösterilmiştir.

$$IMU\ Görüntüsü = Normaliz.\left(SG\left(\begin{bmatrix} a_{x_t}, a_{y_t}, a_{z_t}, w_{x_t}, w_{y_t}, w_{z_t} \\ a_{x_{t+1}}, a_{y_{t+1}}, a_{z_{t+1}}, w_{x_{t+1}}, w_{y_{t+1}}, w_{z_{t+1}} \\ a_{x_{t+2}}, a_{y_{t+2}}, a_{z_{t+2}}, w_{x_{t+2}}, w_{y_{t+2}}, w_{z_{t+2}} \\ \vdots \\ a_{x_{t+9}}, a_{y_{t+9}}, a_{z_{t+9}}, w_{x_{t+9}}, w_{y_{t+9}}, w_{z_{t+9}} \end{bmatrix} \right) \right) x255 \tag{4.11}$$

Şekil 4.13. Farklı IMU nümerik değerleriyle oluşturulan IMU görüntüsü örnekleri

Şekil 4.13'teki görüntüler, içinde iki kare arasındaki hareket bilgilerini içeren IMU bilgilerini saklar. Bu görüntülerin boyutu Inception-V3 ağı için 299×299 olarak yeniden boyutlandırılmıştır. Görsel adımda olduğu gibi ardışık iki IMU görüntüsü birleştirilmiş ve bu iki görüntü arasına iki kare arasındaki hareket alanlarını gösteren OF görüntüsü eklenmiştir. Şekil 4.8'de de görüldüğü gibi bir n zamanına ait karenin pozunu tahmin etmek için sırasıyla IMU_{n-1}, OF_{n-1}, IMU_n görüntüleri birleştirilmiş ve sonuçta Inception-V3 girişi için $299 \times 299 \times 3$'lük yeni görüntüler elde edilmiştir. Sonraki adımlarda eğitim gerçekleştirilmiştir ve yapılanlar tam olarak görsel adımdaki ile aynıdır.

4.3.1.4 Görsel-Atalet Görüntüsü Verilerinin Füzyonu

Önceki adımlar görsel ve atalet bilgilerinin özelliklerini çıkarmaktadır, ve farklı iş parçacıklarında eğitimler gerçekleştirerek eğitilmiş bir Inception-V3 yapısı içermektedir. Ancak bir VIO çalışmasında bu özelliklerin birbirini tamamlayıcı özelliklerinden faydalanmak için füzyon esastır. Şekil 4.8'de görüldüğü gibi eğitilmiş görsel ve atalet adımına ait Inception-v3 mimarisi FC-1 katmanları aracılığıyla, görsel-OF ve atalet görüntüsü-OF özelliklerini çekirdek tabanlı olasılık modeli olan GPR (Williams ve Rasmussen, 1996; Williams ve Rasmussen, 2006; Le Gentil ve ark., 2020) algoritmasına beslemektedir.

Gauss Süreci (Gaussian process) bir uzaydaki rastgele değişkenlerin bir koleksiyonudur. Bayes tabanlı bir regresyon yöntemi olan GPR algoritması, deneyleri tarafsız olarak sonuçlandırmak için başarılı bir tekniktir (Richter ve Toledano-Ayala, 2015; Noack ve ark., 2020). GPR, ardışık gözlemlerin birbirleri hakkında bilgi taşıdığı varsayımına dayanmaktadır. Bilinmeyen bir amaç fonksiyonu için GPR, herhangi bir noktadaki cevap değerinin bir Gauss dağılımına sahip olduğunu varsayar. Yani bir fonksiyonda iki ya da daha çok girdi-çıktı örneği seçildiğinde, çıktıların çok değişkenli bir Gauss dağılımını takip edeceği düşünülür. Örneğin bir GPR tahmininde, f fonksiyonunun x girişindeki cevabı, y çıktısı, Denklem 4.12'deki gibi ifade edilir (Schulz ve ark., 2018; Jiang ve ark., 2021b). Buradaki $\in$ gürültüyü temsil eder ve Denklem 4.13'teki gibi ifade edilen Gauss dağılımına sahiptir.

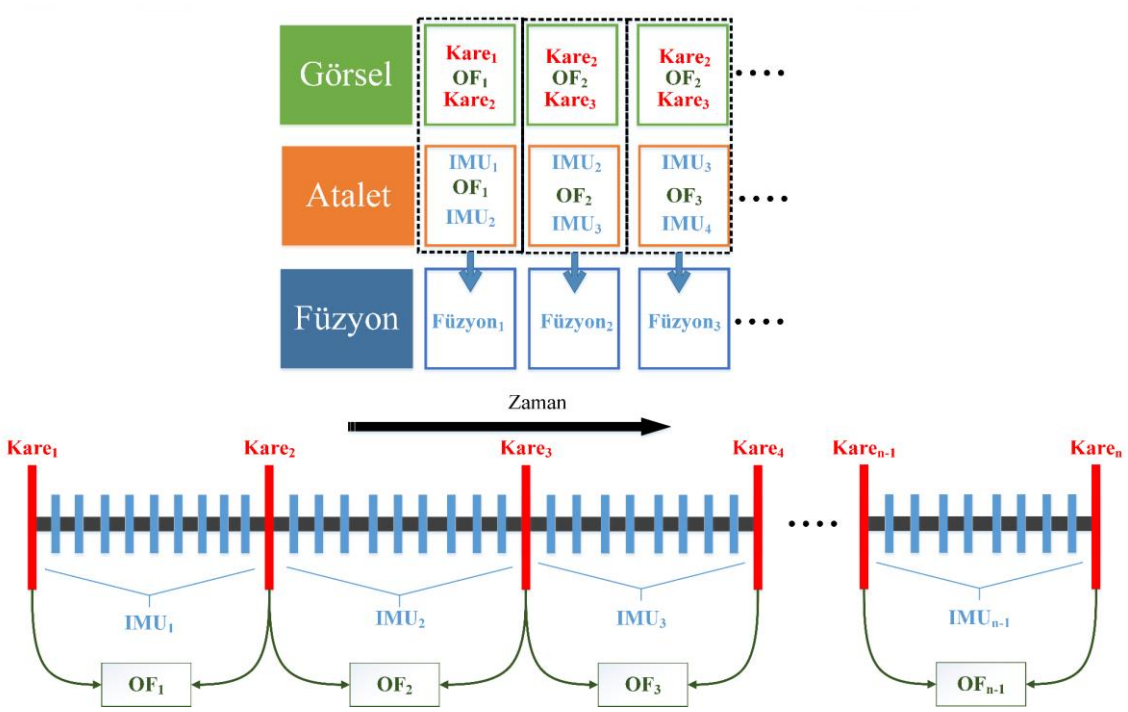
$$y = f(x) + \in \tag{4.12}$$

$$\in \sim N(0, \sigma^2) \tag{4.13}$$

GPR'de, Denklem 4.12'deki $f(x)$ fonksiyonunun ortalama ($m(x)$) ve kovaryans ($k(x, x')$) fonksiyonu ile tanımlanan bir Gauss süreci olarak dağıtıldığı varsayılır (bkz. Denklem 4.14). Kovaryans fonksiyonu çekirdek (kernel) olarak adlandırılır, ve x ve x' giriş sinyalleri arasındaki kovaryansı, yani bağımlılığı belirler. Denklem 4.14'te, pahalı hesaplamalardan kaçınmak ve sadece kovaryansa bağlı bir fonksiyon oluşturmak için $m(x) = 0$ olarak belirlenir. Sonuç olarak $f(x)$ fonksiyonu farklı çekirdek fonksiyonlarına bağlı olarak belirlenir. Bizim çalışmamızda üstel (exponansiyel) çekirdek fonksiyonu kullanılmıştır.

$$f(x) \sim GP(m(x), k(x, x')) \tag{4.14}$$

GPR son zamanlarda makine öğrenmesi algoritmaları arasında ön plana çıkmıştır. YSA'ya göre pratikte uygulanması kolaydır ve SVM'ye göre daha sağlam tahminler gerçekleştirmiştir (Pal ve Deswal, 2010). Bu çalışmada şimdiye kadarki VIO çalışmalarından farklı olarak görsel ve atalet füzyon özellikleri GPR yaklaşımı ile değerlendirilir. Füzyon adımında kullanılan kamera karesi, OF ve IMU bilgilerinin füzyon edilmesi ve zamansal analizi Şekil 4.14'te gösterilmiştir. Buna göre görsel adımdaki $Kare_{n-1}$, OF_{n-1}, $Kare_n$ ve atalet adımındaki IMU_{n-1}, OF_{n-1}, IMU_n dizisinden, eğitilmiş Inception-V3 aracılığıyla çıkarılan özellikler FC-1 katmanları ile GPR'a beslenmiştir. GPR sayesinde belirsizlikler modellenerek bir poz tahmini gerçekleştirilmiştir.



Şekil 4.14. IMU, kare ve OF görüntülerin zamansal ilişkisi ve füzyon işlemi

4.3.2. Pratik Uygulamanın Sonuçları

Bu bölüm hem EuRoC hem de simulasyon veri seti üzerinde uygulanan önerilen yöntemin sonuçlarını ele alır. Şekil 4.8'de tasarlanan mimarinin GPR ile gerçekleştirilen regresyon aşamasına kadar detaylı açıklaması önceki bölümlerde yapıldı. Bu bilgiler ışığında artık ağın eğitim ve test verileri belirlenerek önerilen yöntemin performansı belirlenebilir. Hem EuRoC hem de kendi veri setimiz için eğitim-test oranı %80-%20 olarak belirlendi. Eğitim ve test işlemleri sırasında NVIDIA GeForce GTX 1650 GPU'ya

sahip masaüstü bilgisayar kullanıldı. Önerilen algoritma uygulamasında ortalama bir test verisi için poz tahmin süresi yaklaşık 100 ms olarak hesaplandı.

4.3.2.1 EuRoC Veri Setinin Sonuçları

EuRoC veri seti önerilen çalışmanın performansını önceki çalışmalarla kıyaslamak için sıklıkla tercih edilmektedir. Bizim önerilen yöntemimiz, İHA pozunu üç adımda tahmin eder. Görsel ve atalet adımda bu tahminler Inception-V3 ile, füzyon adımında GPR ile gerçekleştirilmiştir. GPR ile yapılan ham poz tahminleri oldukça başarılı olmasına rağmen, gerçek poz değerleri gürültülü olarak tahmin edilir. Bu nedenle Şekil 4.8'de görüldüğü gibi GPR'den sonra tekrar bir Savitzky-Golay filtresi ile tahmin edilen poz değerleri yumuşatılır. Bu filtreleme sonucunda nihai poz tahmin değerleri elde edilir. Bu çalışmada kullanılan sırasıyla V1-01, V1-02, V1-03, V2-01, V2-02 ve V2-03 test verilerine ait elde edilen her üç adımdaki RMSE değerleri Tablo 4.12'de verilmiştir.

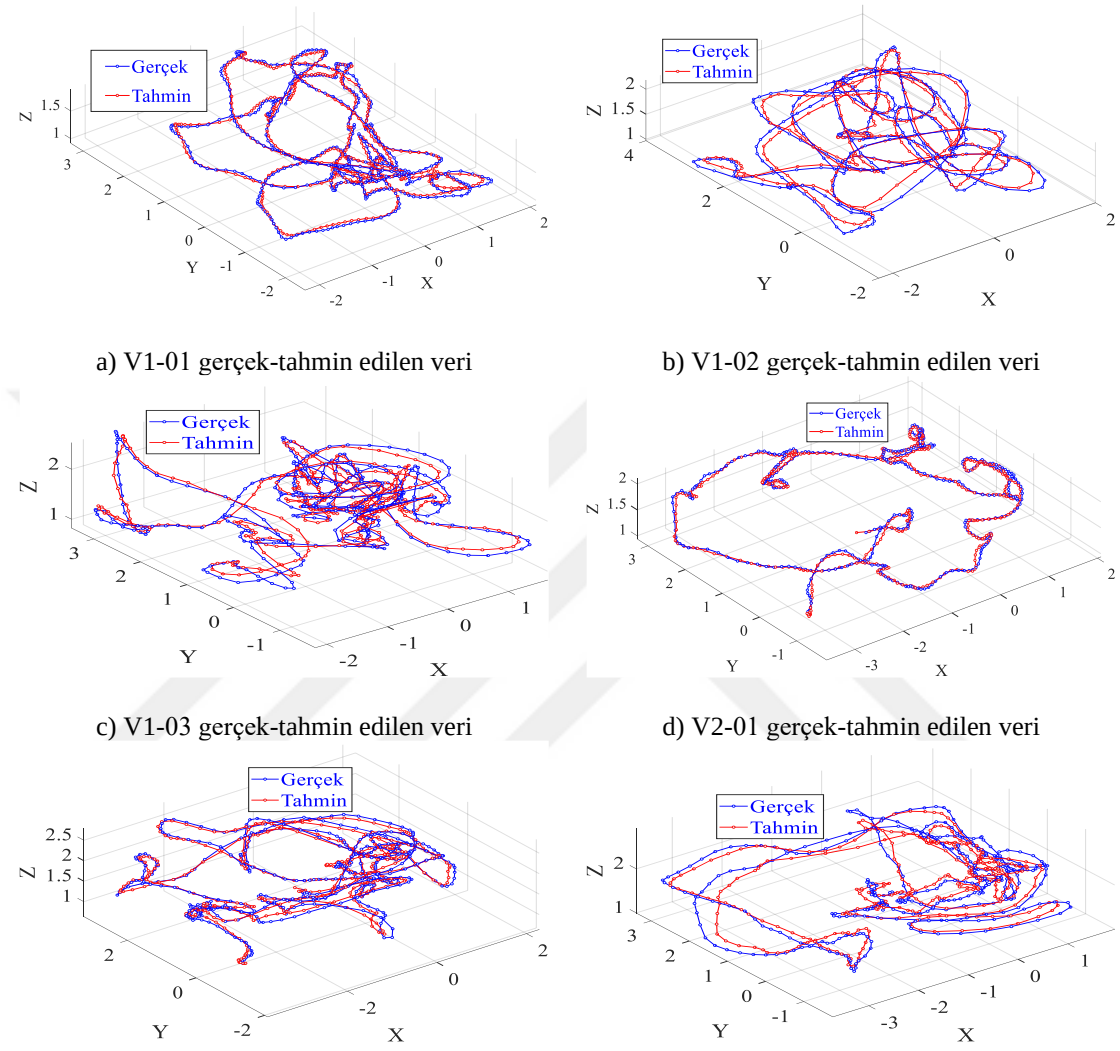
Tablo 4.12. EuRoC veri seti için tahmin edilen İHA pozlarının RMSE değerleri

	V1-01	**V1-02**	**V1-03**	**V2-01**	**V2-02**	**V2-03**
Görsel	0.109	0.091	0.1	0.077	0.094	0.1484
Atalet	0.1162	0.16	0.171	0.095	0.1463	0.229
Füzyon	**0.036**	**0.07**	**0.075**	**0.0302**	**0.0629**	**0.1247**

Tablo 2 incelendiğinde, EuRoC veri setindeki ortam ve uçuş koşulları zorlaştıkça, genel olarak tahmin performansının düştüğü görülür. Görüntü bulanıklığı, İHA hızı gibi unsurlar ortamın doku bilgisini olumsuz etkileyeceğinden, zorlu koşullarda tahmin performansının azalacağı beklenir. Önemli olan V-I adımlarını birlikte işleyerek, füzyon sonucunda minimum hatayı sağlamak ve zorlu koşullara karşı daha dayanıklı bir poz tahmini sağlamaktır. Tablo 4.12'deki füzyon sonuçları, diğer iki adıma göre oldukça daha kararlı sonuçların elde edildiğini göstermektedir. Bu tahmin edilen İHA yörüngesinin performansını daha iyi kavramak için, test verilerinin gerçek değerleri ve önerilen yöntemle tahmin edilen poz değerleri Şekil 4.15'te gösterilmiştir.

Şekil 4.15 önerilen yöntemin zorlu yörünge değişimlerinde de oldukça başarılı olduğunu göstermektedir. Yöntemimizin EuRoC veri setini kullanan farklı çalışmalarla kıyaslanması için Tablo 4.13 incelenebilir. Geçmişte yapılan başarılı V(I)O / V(I)SLAM çalışmalarının benimsediği yöntem genellikle geleneksel geometri tabanlıdır. Son zamanlarda popüler olan derin öğrenme tabanlı yaklaşımlar da EuRoC veri seti üzerinde performans geliştirme çalışmaları uyguladı. Tablo 4.13 hem geleneksel hem de derin öğrenme kullanan popüler, modern çalışmaların algoritma ve sonuç bilgilerini

göstermektedir. Tabloya göre son derece popüler yöntemlere göre, bizim algoritmamız oldukça dikkat çekici bir performansa sahiptir. Ayrıca önerilen VIIONet şimdiye kadar uygulanmamış farklı ve modern derin öğrenme yaklaşımları ile VI poz tahmini gerçekleştirmiştir.



a) V1-01 gerçek-tahmin edilen veri

b) V1-02 gerçek-tahmin edilen veri

c) V1-03 gerçek-tahmin edilen veri

d) V2-01 gerçek-tahmin edilen veri

e) V2-02 gerçek-tahmin edilen veri

f) V2-03 gerçek-tahmin edilen veri

Şekil 4.15. Füzyon adımında, EuRoC test veri setinin gerçek ve tahmin edilen değerleri

Tablo 4.13. Önerilen yöntemin önceki çalışmalarla karşılaştırılması

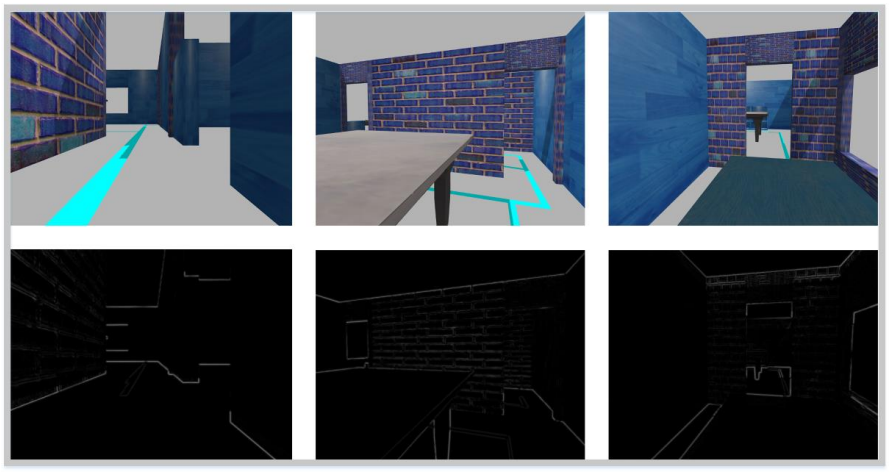
Önceki Çalışma	Algoritma	V1-01	V1-02	V1-03	V2-01	V2-02	V2-03
SVOMSF (Kaiser ve ark., 2016)[1]	VIO (Yarı doğrudan-Filtre tabanlı)	0.400	0.630	-	0.200	0.370	-
MSCKF (Mourikis ve Roumeliotis, 2007)[1]	VIO (Filtre tabanlı)	0.340	0.200	0.670	0.100	0.160	1.130
OKVIS (Leutenegger ve ark., 2015)[1]	VISLAM (Ana Kare tabanlı)	0.090	0.200	0.240	0.130	0.160	0.290
ROVIO (Bloesch ve ark., 2015)[1]	VIO (Filtre tabanlı)	0.100	0.100	0.140	0.120	0.140	0.140
VINS-MONO (Qin ve ark., 2018)[1]	VIO (Optimizasyon tabanlı)	0.070	0.100	0.130	0.080	0.080	0.210

VINS-MONO-LC (Lynen ve ark., 2013)[1]	VIO (Optimizasyon tabanlı)	0.040	0.060	0.110	0.060	0.060	0.090
VI-DSO(Stumberg ve ark., 2018) [2]	VIO	0.059	0.067	0.096	0.040	0.062	0.174
SVOGTSAM (Forster ve ark., 2016a)[1]	VIO (Yarı doğrudan-Optimizasyon Tabanlı)	0.070	0.110	-	0.070	-	-
ORB-SLAM2 (Mur-Artal ve Tardós, 2017b)3	VSLAM (Özellik Tabanlı)	0.087	0.065	0.092	0.071	0.061	0.184
STCM-SLAM (Chen ve ark., 2019)[3]	VISLAM (Ana Kare Tabanlı)	0.079	0.113	0.120	0.084	0.060	0.099
SELFVIO (Almalioglu ve ark., 2019)[3]	Derin Öğrenme	0.080	0.090	0.100	0.110	0.080	0.110
DVIO (Jiang ve ark., 2021a)[4]	VIO (Doğrudan Optimizasyon Tabanlı)	0.090	0.080	0.170	0.100	0.100	0.110
PL-SLAM(Gomez-Ojeda ve ark., 2019) [4]	VSLAM (Stereo, Nokta Çizgi (Point-Line)	0.0423	0.0459	0.0689	0.0609	0.0565	0.1261
OV^{2}SLAM(Ferrera ve ark., 2021) [4]	VSLAM (Stereo, Optimizasyon Tabanlı)	0.090	0.070	0.090	0.070	0.060	-
KIMERA (Rosinol ve ark., 2020) [2]	VISLAM (Stereo - Semantik)	0.050	0.110	0.120	0.070	0.100	0.190
Li ve Waslander (2020)[3]	Derin Öğrenme	2.070	2.200	2.830	1.490	1.490	2.220
Bizim Yöntemimiz	**Derin Öğrenme +GPR (VIIONet)**	**0.036**	**0.07**	**0.075**	**0.0302**	**0.0629**	**0.1247**

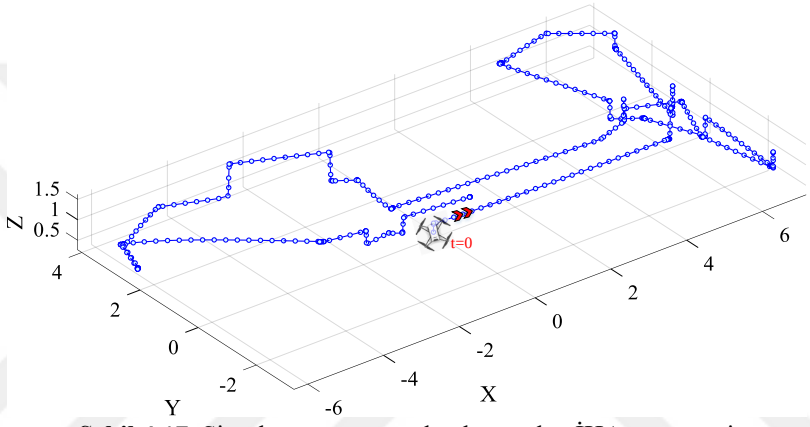
(V1-01, V1-02, V1-03, V2-01, V2-02, V2-03 farklı ortam koşullarındaki verileri temsil eder .)
[1](Delmerico ve Scaramuzza, 2018) çalışmasında belirtilen hatalar
[2](Campos ve ark., 2021a) çalışmasında belirtilen hatalar
[3] (Yusefi ve ark., 2021) çalışmasında belirtilen hatalar
[4] İlgili çalışmada belirtilen hatalar

4.3.2.2 Simülasyon Veri Kümesi İle Elde Edilen Sonuçlar

Önerilen yöntemin farklı ortamlarda ve veri setlerinde test edilmesi, algoritmanın sağlamlığını kanıtlar. Bu bağlamda, önerilen yöntem için Gazebo ortamındaki bir ortamda DJI Tello kullanılarak bir uçuş simulasyonu gerçekleştirilmiştir. Bu simülasyon ortamı uygulama 2'deki ile aynıdır. Kapalı bir ortamda gerçekleştirilen bu İHA hareketinde, Tello kamerası ile alınan bazı kareler ve ardışık iki karenin OF görüntüleri Şekil 4.16'da gösterilmiştir.



Şekil 4.16. Simülasyon ortamında Tello ile kaydedilen bazı orijinal kareler ve bu karelere göre oluşturulan OF görüntüleri



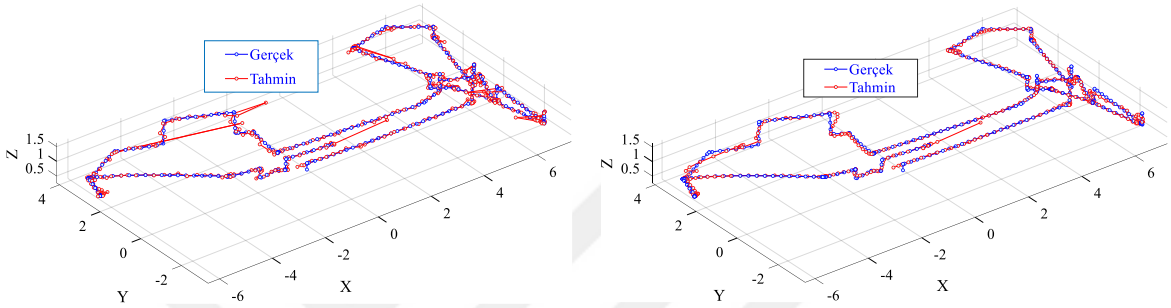
Şekil 4.17. Simülasyon ortamında oluşturulan İHA yörüngesi

Simulasyonda Tello $(0, 0, 0)$ noktasında harekete başlamakta ve kapalı ortamda rastgele bir yörüngede hareket ettirikmetedir. Şekil 4.17'deki yörüngeyi izleyen İHA, aynı zamanda 2432 karelik bir video, IMU ölçümleri, gerçek poz değerlerini ve zaman damgalarını kaydetmiştir. Yörünge boyunca var olan ani değişiklikler, oluşturulan veri setinin zorluğunu göstermektedir. Ani yön değişiklikleri konum tahminini zorlaştırır ve başarıyı düşürür. Ancak bizim önerdiğimiz yöntem oldukça başarılı poz tahminleri sağlamıştır. Şekil 4.18, simulasyon veri seti test verisinde, önerilen yöntem tarafından tahmin edilen poz değerlerini gerçek poz değerleri ile karşılaştırır. Savitzky-Golay filtresinin etkisini göstermek için, Şekil 4.18, GPR tahminleri ile birlikte, ayrıca bu tahmine uygulanan filtrenin poz tahminini ne ölçüde iyileştirdiğini göstermektedir. Doğrudan GPR çıkışlarının gösterildiği Şekil 4.18 (a)'da RMSE değeri 0.1234 iken, Savitzky-Golay filtresi sonrasında bu değer Şekil 4.18 (b)'de görüldüğü gibi 0.0796 olarak elde edilir. Konum Değerlerinin filtre sonucunda gerçek poz değerleri ile daha fazla örtüştüğü Şekil 4.18 (b) ile rahatlıkla anlaşılabilir. Sonuç olarak, simulasyon veri setinde, önerilen yöntemin üç adımda sağladığı başarı Tablo 4.14'te gösterilmiştir. Bu

sonuçlar EuRoC'da olduğu gibi zorlu ortamlarda önerilen yöntemin performansını kanıtlamaktadır.

Tablo 4.14. Simülasyon veri seti için tahmin edilen Tello pozisyonlarının RMSE değerleri

Görsel Veri	Atalet-OF	Füzyon (GPR- Savitzky-Golay)
0.0843	0.2597	0.0796



a) Savitzky-Golay filtresinden önce (RMSE: 0.1234)

b) Savitzky-Golay filtresinden sonra (RMSE: 0.0796)

Şekil 4.18. Simülasyon veri seti için gerçek ve tahmin edilen İHA yörüngesi

4.4. UYGULAMA 3: Yeni Bir İHA Yol Planlama Yaklaşımı, Yol Optimizasyonu ve Öğrenmeye Dayalı Yol Tahmini (GDRRT*, PSO-GDRRT* ve BiLSTM-PSO-GDRRT*)

Mobil robotların otonomize olması için temel şart, mobil robotun kendini ortamda lokalize etmesi ve ortamın yapısını bilmesidir. Bu şart sağlandıktan sonra bu mobil robota belirli bir görev verilir, ancak robotun bu görev noktası için nasıl bir navigasyon gerçekleştireceği önemli bir konudur. Özellikle son zamanlarda uygulaması artan İHA'ların, 3B bir ortamda yol planlaması yaygın bir problemdir. Bu uygulama büyük bir iç ortam oluşturarak İHA için başlangıç ve hedef noktaları arasında, çarpışmalardan kaçınmak ve en optimum yolu keşfetmek için RRT* algoritmasını temel almaktadır. Amaç RRT*'ın hedefe yavaş yakınsaması problemini çözmektir.

Önceki çalışmalardan hareketle çok sayıda RRT* tabanlı yeni yol planlama çalışmaları şimdiye kadar geliştirilmiştir. RRT* tabanlı algoritmik çalışmalar her ne kadar geliştirilebilir olsa da, gerçek zamanlı/hızlı yol planlama ve statik-dinamik nesne ayrımı gibi çözümler için umut verici değildir. Zaten şimdiye kadar önerilen yeni varyantlar da hala, ortam boyutunun artmasına bağlı olarak hesaplama süresinin artması problemi içerirler (Qureshi ve ark., 2020). Sonuç olarak her ne kadar yeni varyantlar,

algoritmik olarak yeni adımlar ekleyerek yolu daha optimize etseler bile, gerçek zamanlı bir mobil robot için hızlı keşif problemi hala mevcuttur. Ayrıca mobil robot kategorisinde son zamanlarda uygulama alanı genişleyen İHA'lara yönelik yol planlama çalışmaları yetersizdir. Çünkü 3B ortamlarda yol planlama ile ilgili çalışmalar, 2B ortamlardaki çalışmalara göre çok az sayıdadır. Arzu edilen yol planlama algoritması hem 2B hem de 3B ortamlar için optimal yolu en kısa sürede bulmalıdır. Ayrıca gerçek dünya ortamları sadece belirli yükseklikte engeller içermez, ortamda farklı yüksekliklere sahip engeller bulunur. Zaten engellerin yüksekliği sabit ve ortamın yüksekliğine eşitse, ilgili planlama 2B için geçerlidir. Gerçek dünya problemleri için yeterli bir yol planlama algoritması aşağıdaki şartları sağlamalıdır:

- Eğer hedefe ulaşım mümkünse, hedefe ulaşan optimal yolu bulmalıdır.
- Hedefe hızlı yakınsamalıdır, yani hızlı keşif sağlamalıdır.
- Büyük ortamlarda ya da yüksek boyutlu ortamlarda gerçek zamanlı performans göstermelidir.
- Hem 2B hem 3B ortamlarda verimli çalışmalıdır.
- Statik ya da dinamik engellerin varlığında, engelleri algılayarak yol planlama gerçekleştirebilmelidir.
- Ortamın karmaşıklığından etkilenmemeden optimal yolu hızlı bir şekilde hesaplayabilmelidir.
- Robot harekete başladıktan sonra yol hemen hesaplanabilmelidir.

Günümüzde oldukça aktif bir alan olan yapay zekâ, matematiksel karmaşıklığa sahip algoritmaları daha kolay, yüksek doğrulukta ve insansı bir şekilde çözebilmektedir. Yapay zekâ ile hareket planlamasının (motion planning) gerçekleştirildiği çalışmalarda (Bency ve ark., 2019; Qureshi ve ark., 2019; Qureshi ve ark., 2020) yukarıdaki şartların büyük bir kısmı sağlanmıştır. Ayrıca henüz gerçekleştirilemeyen şartların yapay zekâ ile ilerleyen zamanlarda geliştirilebileceği açıktır. Örneğin yol planlama sırasında statik-dinamik nesne ayrımı yapay zekâ ile geliştirilmeye çok uyumludur. Örnek olarak, dinamik obje içeren ortamlarda, statik ve dinamik objelerin ayırt edilmesi ve sonrasında SLAM gerçekleştirilmesine yönelik yapay zekâ tabanlı çalışmalar (Han ve Xi, 2020; Wen ve ark., 2021) yakın zamanda uygulanmıştır.

Yapay zekâ tabanlı yol planlama çalışmaları gerçek zamanlı gerçek dünya robotik uygulamalarının eksikliklerini karşılayabilir niteliktedir. Bu nedenle gelecek yol planlama çalışmalarının yapay zekâdan faydalanacağı açıktır. Ancak çoğu yapay zeka çalışması danışmanlı yapıda olduğundan, optimal yolu üreten geleneksel yöntemlere (örn.

RRT*) de hala ihtiyaç vardır. Bu bağlamda, bu uygulamada şimdiye kadarki çalışmaların katkıları ve eksiklikleri baz alınarak hem yeni bir RRT* varyantı hem de optimize edilmiş yolların yapay zeka tabanlı tahminine yönelik bir uygulaması sunulur. Uygulama İHA kullanılarak geliştirildiğinden, önerilen yöntem hem 2B hem de 3B ortamlar için yol planlaması sağlamaktadır. Genel olarak pratik uygulama üç aşamada ele alınabilir.

İlk aşamada yeni bir RRT* varyantı oluşturulur, bu yeni varyant Hedef Mesafeye dayalı RRT* (Goal Distance-based RRT* (GDRRT*)) olarak adlandırılır. GDRRT*, RRT*'ın hedefe yavaş yakınsama problemini çözerek hedefe, engelleri aşarak daha hızlı yakınsamayı amaçlamaktadır. RRT*'ın hedefe geç yakınsamasındaki en büyük neden, örneklerin hedeften bağımsız olarak rastgele üretilmesidir. Bu nedenle, GDRRT*'nin daha akıllı bir rastgele örnekleme yapması sağlanmıştır. Bunun için rastgele örneğin alabileceği maksimum ve minimum değerler belirlenir. Bu sayede rastgele örnek, her üç eksende (x, y, z) belirli miktarda yer değiştirebilir. Gereksiz örneklemeyi engellemek ve hedefe sürekli yakınsamak için, hedefe en yakın düğümler sürekli kaydedilir ve yeni rastgele örnekleme hedefe en yakın düğüm baz alınarak yapılır. Eğer hedefe yakınlaşma sağlandığı halde, mevcut düğüm ile hedef arasında bir engel varsa, bu durumda RRT*'de olduğu gibi tamamen rastgele örnekleme ile engelden kurtulma gerçekleştirilir. Engelden kurtulmak, ya da çarpmadan kaçınmak için RRT*'daki ağaç uzama değeri artırılır, engelsiz ortamda bu değer sabit tutulur. Eğer hedefe yakınlık belirli bir eşik değerinin altına düşerse, bu uzama değeri bu sefer azaltılır. Tüm bu algoritma sonunda rastgele oluşturulan bir 3B ortamda başlangıç ile hedef noktayı birleştiren yollar elde edilmiştir.

İkinci aşama GDRRT* tarafından bulunan yolun daha optimum hale getirilmesi için, mevcut yola PSO uygulanmasını içermektedir. PSO uygulanmadan hemen önce, bulunan yola ait kenarları birbirine bağlayan kritik noktaların sayısı interpolasyon tekniği ile sabit bir değere düşürülmüştür. Bu sayede daha doğrusal yollar oluşur ve PSO'nun işlem süresi kısaltılır. Çünkü PSO, bu mevcut kritik noktaları, engellere temas etmeden, en kısa yolu bulacak şekilde optimize etmektedir. Sonuçta GDRRT* ile oluşturulan ilk yol, PSO ile daha optimize bir yola dönüştürülmüştür (PSO-GDRRT*).

Son aşama, optimize edilmiş GDRRT* yollarının (PSO-GDRRT*), RNN tabanlı bir mimari ile tahminini ele almaktadır. PSO-GDRRT* tarafından oluşturulan yola, yapay zekâ için kullanılmadan önce interpolasyon uygulanmıştır. Bu sayede yolları oluşturulan kritik noktaların düzgün aralıklı dağılımı sağlanır. Çünkü bu düzgün aralık yapay zekânın eğitim aşamasında daha güçlü öğrenmesini sağlayacaktır. Eğitim ve test için kullanılan ve hedef (ground-truth) olarak ağa verilen dizi, bu interpole edilmiş PSO-GDRRT*

çıktısıdır. Son olarak öğrenme tabanlı bir yapı için BiLSTM katmanı kullanılır ve tüm ardışık yol dizisi uçtan uca bir mimari ile tahmin edilir (BiLSTM-PSO-GDRRT*). Ağın eğitimini sağlamak için 3B haritalar 2B görüntülere dönüştürülür. Hedef yollar interpole edilmiş PSO-GDRRT* tarafından oluşturulan yoldur. Sonuçta PSO-GDRRT* ile oluşturulan optimize edilmiş kısa yollar, BiLSTM-PSO-GDRRT* ile gerçek zaman/ hızlı yol planlama gereksinimini karşılayacak şekilde tahmin edilir.

Yukarıda genel olarak anlatılan çalışmanın, önceki çalışmalara kıyasla katkıları ve farklılıkları aşağıdaki gibidir:

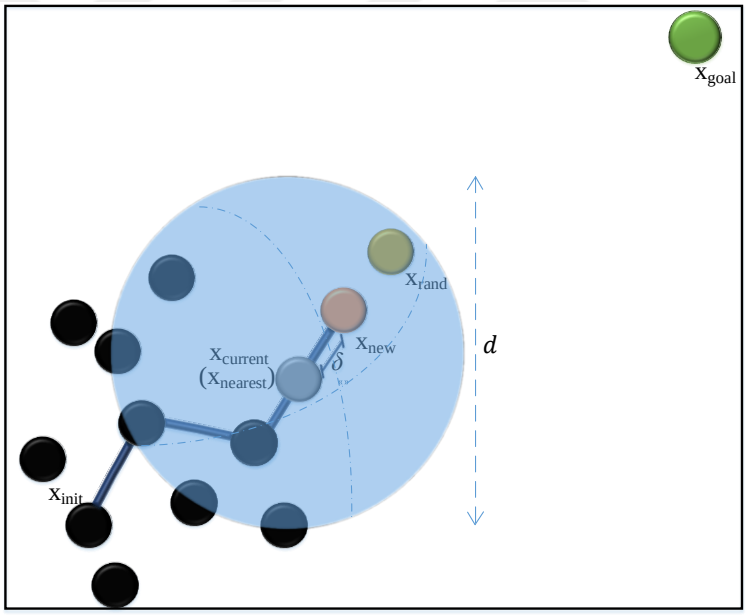
- İHA için hem 2B hem 3B ortamlarda uygulanabilen bir yol planlama çalışması önerilir.
- Hedef noktaya RRT*'a göre daha hızlı yakınsayan bir GDRRT* yol planlama yöntemi tanıtılır.
- GDRRT* ile bulunan yolu büyük ölçüde PSO ile optimize eden PSO-GDRRT* uygulaması gerçekleştirilir.
- PSO-GDRRT* sonucunda elde edilen optimum yollar BiLSTM kullanılarak sıralı (sequential) şekilde tahmin edilir.
- Sonuçlar gerçek zamanlı İHA yol planlama çalışmaları için uyumludur.

4.4.1. Hedef Mesafeye Dayalı RRT (GDRRT*)

Bu bölüm RRT*'ye alternatif olarak sunulan GDRRT*'yi tanıtmak için, GDRRT*'ı detaylı olarak ele almaktadır. Her iki algoritma da örnekleme tabanlıdır ve daha önce de bahsedildiği gibi yüksek boyutlu durum uzaylarında, diğer yöntemlere göre sağladığı üstünlükten dolayı daha popülerdir. Pratik uygulamalarda verimlilik açısından sağladığı avantaj nedeniyle örnekleme tabanlı yöntemler arasında RRT ön plana çıkmıştır. Daha sonra, Karaman ve Frazzoli (2011) tarafından önerilen asimptotik optimalliği garanti eden RRT*, örnekleme tabanlı yöntemler için yeni bir çığır açmış ve, bu çalışmada olduğu gibi farklı yeni yaklaşımlara ön ayak olmuştur. RRT*, RRT'den farklı olarak mevcut algoritmaya yeni $ChooseParent$ ve $Rewire$ fonksiyonlarını ekler. Sonuç olarak bu fonksiyonlar sayesinde bu algoritma asimptotik optimal olur, yani örnek sayısı sonsuza yaklaştığında optimal çözümü bulma olasılığı da bire yaklaşır (Li ve ark., 2020). RRT* her ne kadar uygun yolu bulma garantisi verse de, bu yolun oluşma süresini kısaltmak, yani hedefe hızlı yakınsamak çok önemlidir. Zaten tüm yeni varyantlar daha hızlı keşife odaklanır. Bu çalışmada önerilen GDRRT* algoritması da benzer şekilde hedefe daha hızlı yakınsamayı amaçlamaktadır.

RRT*'in hedefe yavaş yakınsamasındaki en büyük neden, hiçbir sınırlama olmadan rastgele örnekleme gerçekleştirmesidir. RRT* algoritması, örnek sayısı sonsuza giderken, optimal yolu bulmayı garanti etmesine rağmen, yüksek boyutlu ortamlar için RRT*'ın yakınsama hızı maalesef katlanarak azalır. Bu nedenle daha akıllı bir örnekleme yavaş yakınsama problemini çözebilir. Bundan hareketle, GDRRT*, rastgele örneklemeyi kısıtlandırarak, her bir örneklemede hedefe yakınsamayı amaçlar. Ancak bu yakınsamayı kontrollü olarak gerçekleştirir. Kontrollü yakınsama bir d değeri aracılığıyla sağlanır, yeni örnek mevcut düğüm (node) merkezli, d çapına sahip bir küre sınırları içinde olmalıdır. Başka bir ifade ile, yeni rastgele örnek, mevcut düğümden üç eksende (x, y, z) de maksimum $d/2$ uzaklıkta olabilir. Bu durumda, yeni örneğin maksimum uzaklığı $(d/2)$, ağaç uzama mesafesinden (δ) az olmalıdır, yani Denklem 4.15 şartı sağlanmalıdır.

$$\frac{d}{2} > \delta \tag{4.15}$$



Şekil 4.19. GDRRT* örnekleme tekniği

Şekil 4.19, GDRRT*'nin örnekleme stratejisini göstermektedir. Şekil 4.19'da görüldüğü gibi, yeni örnek bu üç boyutlu uzay içinde herhangi bir noktada olabilir. Dolayısıyla yeni örnekleme belirli bir sınırlı uzayda rastgele yapılır. Bu uzayın sınırlarını, merkezdeki mevcut düğümün $(x_{current})$ koordinatları belirler. $x_{current}$ aynı zamanda tüm düğümler içinde hedefe en yakın düğümü temsil etmektedir. Yani, uzaydaki yeni örnek, eğer mevcut düğüme göre hedefe daha uzaksa (Öklid mesafesi olarak) $x_{current}$ değişmez. Sürekli olarak, yeni örnek hedefe en yakın olan $x_{current}$ düğümü baz alınarak

oluşturulur. En yakın düğüme ait koordinat değerleri pos_temp değişkeni içinde saklanır. Geçici pozisyon değeri olan pos_temp'in ilk değeri başlangıç koordinatlarıdır, yani x_{init}'e eşittir. Hedefe yaklaşıldıkça pos_temp güncellenir. pos_temp ve d değerine bağlı olarak örnekleme Denklem 4.16 ile gerçekleştirilir.

$$x_{rand} = (pos\_temp - \frac{d}{2}) + rand(0,1) \times ((pos\_temp + \frac{d}{2}) - (pos\_temp - \frac{d}{2})) \quad (4.16)$$

Algoritma 4.2. GDRRT*

```
1.  T ← InitializeTree();
2.  T ← InsertNode(∅, x_init, T);
3.  x_rand = sample(d, pos_temp);
4.  for i = 0 to i = N do
5.      x_nearest = Nearest(T, x_rand);
6.      x_new = Steer(x_nearest, x_rand, δ);
7.   if Collisionfree(x_new) then
8.      δ=δ_begin;
9.       if dist_new < dist_min then
10.        pos_temp = x_new;
11.        dist_min = dist_new;
12.      if dist_min < dist_goal_min then
13.         δ=δ_min;
14.     x_rand = sample(i, d, pos_temp);
15.     x_near ← Near(T, x_new, V, R_near );
16.     x_min ← Chooseparent(x_near, x_nearest, x_new);
17.     T ← InsertNode(x_min, x_new, T);
18.     T ← Rewire(T, x_near, x_min, x_new);
19.  else
20.     δ=δ + k;
21.     x_rand = sample(i);
22. return T
```

Denklem 4.16 sayesinde örneğin maksimum ve minimum sınırlar belirlenmiştir. GDRRT* ile RRT*'da çok sayıdaki adım aynıdır. GDRRT*'ın sözde kodu Algoritma 4.2'de gösterilmiştir. Örnek oluşturulduktan sonra $Nearest$ ve $Steer$ fonksiyonları RRT*'daki gibi uygulanır. Sonuçta oluşan yeni örnek, x_{new}, için RRT*'a benzer şekilde çarpışma kontrolü yapılır. Çarpışma yoksa δ değeri, başlangıç değerine (δ_{begin}) eşitlenir. Daha sonra, x_{new} düğümünün x_{goal}'e olan öklid mesafesi ($dist\_new$), hedefe en yakın olan $x_{current}$ düğümünün mesafesi ($dist\_min$) ile karşılaştırılır. $dist\_new$'in başlangıç değeri, x_{init} ve x_{goal} arasındaki mesafedir. Eğer x_{new} hedefe daha yakınsa, pos_temp değeri x_{new} koorninarlarına eşitlenir ve $dist\_min$ de $dist\_new$ değerine eşitlenir. Bu durum hedefe yaklaşmayan düğümlerin elenmesini sağlar. Bu hem az düğüm kullanılmasını hem de hedefe daha hızlı yakınsamayı sağlar. Bir sonraki şart ile, eğer $dist\_min$, hedefe minimum mesafe eşik değerinden ($dist\_goal\_min$) küçükse δ değerini azaltır. Bu durum hedefe yakınlaşıldığında hedeften tekrar uzaklaşma ihtimalini azaltır.

Sonraki adım, pos_temp ve d değerine göre yeni bir örnekleme yapar. GDRRT*, RRT*'daki gibi çarpışma varken ya da yokken aynı örnekleme stratejisini kullanmadığından, örnekleme işlemleri çarpışma şartı içinde gerçekleştirilir. Sonraki fonksiyonlar olan $Near$, $Chooseparent$, $InsertNode$ ve $Rewire$ RRT* ile tamamen aynıdır. Sonraki şart, yani x_{new} için çarpma durumu varsa bu durumda amaç engelden olabildiğince hızlı kurtulmaktır. Hedefi hızlı aşmak için bir sabit k değeri ile, δ artırılır. Daha sonra RRT*'dakine bezer şekilde tamamen rastgele bir örnek oluşturulur. Döngü tamamlandığında ağaç yapısı cevap olarak döndürülür. Daha kolay anlaşılması için, Algoritma 4.2'ye göre her bir satırın anlatımı aşağıdaki gibidir.

Satır 1: Başlangıç ayarları, engelsiz alanda $T = (V, E)$ ağaç yapısını oluşturmak için etkinleştirilir. Burada V, örneklenen köşeleri temsil eder ve E, bu köşeleri birleştiren kenarları temsil eder.

Satır 2: x_{init} ağacın ilk kenarsız düğümü olarak atanır.

Satır 3: $Sample$ fonksiyonu ile döngü başlamadan önce ilk olarak üç eksende maksimum $d/2$ uzaklıkta bir örnek (x_{rand}) başlangıç noktasına (x_{init}) göre oluşturulur.

Satır 4: Döngü başlar.

Satır 5: $Nearest$ fonksiyonu ile hedef noktaya ulaşılmadığı müddetçe üretilen örneğe en yakın düğüm noktası ($x_{nearest}$) bulunur.

Satır 6: $Steer$ fonksiyonu ile en yakın düğümden ($x_{nearest}$) örneğe (x_{rand}) doğru maksimum uzatma mesafesi (δ) baz alınarak yeni bir düğüm noktası (x_{new}) oluşturulur.

Satır 7: $Collisionfree$ fonksiyonu $x_{nearest}$ ile x_{new} arasındaki yolun engele çarpıp çarpmadığını denetler.

Satır 8: Atlama mesafesi (δ) ilk değerine (δ_{begin}) atanır.

Satır 9: x_{new}'in hedefe olan uzaklığı ($dist\_new$), şimdiye kadarki hedefe olan minimum uzaklık değeri ($dist\_min$) ile karşılaştırır.

Satır 10: Eğer x_{new} hedefe daha yakınsa, pos_temp, x_{new}'e eşitlenir. Bu sayede bir sonraki rastgele örneklemenin pos_temp'e göre yapılması sağlanır.

Satır 11: Ayrıca yakınlaşma olduğundan $dist\_max$ değeri, $dist\_new$ değerine eşitlenir. Bu sayede hedefe olan en yakın mesafe sürekli kaydedilir.

Satır 12: Hedefe olabilecek en düşük mesafe ($dist\_goal\_min$) ile $dist\_max$ değerini karşılaştırır.

Satır 13: Eğer eşik değerin altında bir yakınlık mevcutsa, hedefe daha kolay yaklaşmak için, δ, minimum atlama değerine (δ_{min}) eşitlenir.

Satır 14: pos_temp ve d değerine göre üç eksende maksimum $d/2$ uzaklıkta uzaklıkta yeni bir x_{rand} oluşturur.

Satır 15: $Near$ fonksiyonu ile x_{new} merkezli belirli bir R_{near} yarıçapına sahip bir küre içindeki bir dizi köşe noktaları bulur.

Satır 16: $Chooseparent$, optimum x_{new} yolunu oluşturan tepe noktasını (vertex) bulmak için x_{near} noktalarında arama yapar. Herhangi bir x_{near} noktasının ana düğümünü (parent node) x_{rand} ile değiştirmek, x_{init}'den x_{near}'a giden yolun maliyetinin düşmesine bağlıdır. Eğer daha optimum yolu sağlayan bulunursa, bu yeni düğüm x_{min}'dir.

Satır 17: $InsertNode$ yeni düğümler (x_{min}, x_{new}) mevcut düğümlere eklenir

Satır 18: *Rewire* fonksiyonu yeni düğümlere göre kablolamayı günceller.

Satır 19: $x_{nearest}$ ile x_{new} arasındaki yolda çarpma varsa,

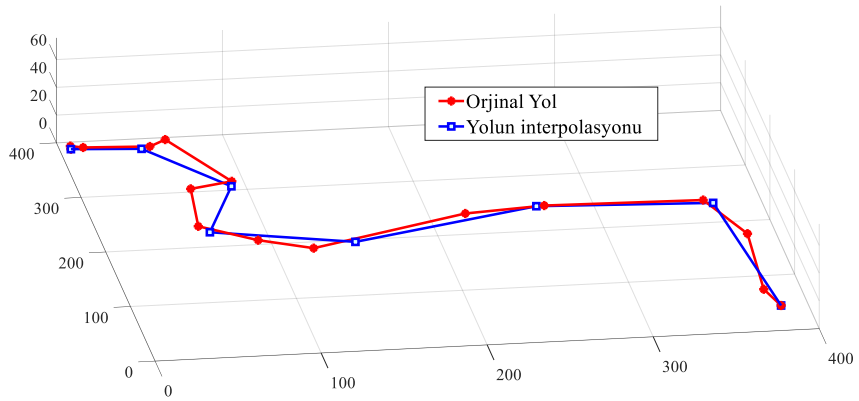
Satır 20: δ değeri sabit bir k değeri ile artırılarak, engelden daha kolay kurtulma amaçlanır.

Satır 21: Yeni x_{rand}, sınırlama olmadan rastgele üretilir.

Satır 22: GDRRT*'ın çıktısı olarak ağaç yapısı (T) döndürülür.

4.4.2. GDRRT*'NİN PSO İLE OPTİMİZASYONU (PSO-GDRRT*)

GDRRT* tarafından oluşturulan yol, RRT*'a göre hedefe daha hızlı yakınsamasına rağmen, optimal değildir. Bulunan yol uzun olabilir ve gereksiz düğüm ve kenarlar içerebilir. Bu nedenle bu çalışma, PSO-GDRRT* olarak ifade edilen yöntem ile, GDRRT* ile üretilen mevcut yolu PSO tekniği ile optimize ederek daha optimum yollar oluşturmuştur. GDRRT* tarafından üretilen yoldaki mevcut düğümlerin dağılımı, engelsiz uzayda minimum yol sağlayacak şekilde PSO ile optimize edilir. Ancak bunun öncesinde, PSO tarafından üretilen yolun daha doğrusal olmasını sağlamak ve çözüm hızını artırmak için mevcut yoldaki düğüm sayısı azaltılır. 3B uzayda bir yolun interpolasyonu Şekil 4.20'deki gibidir. İnterpolasyon tekniği sayesinde daha az düğüme sahip, benzer bir yol elde edilir.



Şekil 4.20. Ham yolun PSO öncesi interpolasyonu

Kennedy ve Eberhart (1995), PSO yöntemi için, sürü halinde yaşayan hayvanların ortak bir temel ihtiyaç için (örn. yemek arama) sürü halinde nasıl etkileşimde bulunduklarından esinlendiler. Sürüdeki her bir parçacık, deneyimini sürü ile paylaşarak, gerçekleştirilecek ortak göreve katkıda bulunurlar. Bu bağlamda PSO, birey sayısının fazlalığı avantajını kullanarak, uzayda sürü halinde arama yaparak farklı problemler için optimal çözümleri bulur. Optimal çözüme ulaşmak için, uzaya rastgele dağıtılan parçacıkların birbirleri arasında bilgi paylaşımı temel alınır. İlk aşamada uzaya rastgele dağıtılan parçacıkların uygunluk değeri (fitness value) her bir iterasyonda hesaplanır, daha sonra her parçacığa ait değerlerin en optimumu (p_{best}), hem de sürüye ait değerlerin en optimumu (g_{best}) kaydedilir. Bu değerler ve sabit parametreler kullanılarak her bir parçacığa ait yeni pozisyon ve hız değerleri Denklem 4.17 ve Denklem 4.18 ile hesaplanır. Denklem 4.17 ve Denklem 4.18'de, i parçacık indeksini, w imertia weight değerini, c_1, c_2 ivme katsayılarını, V parçacık hızını, P parçacık konumunu temsil eder. $rand_1$ ve $rand_2$ fonksitonları 0-1 arasında rastgele değer üretir (Sabanci ve Balci, 2020).

$$V_{i+1} = wV_i + c_1 \times rand_1(p_{best} - V_i) + c_2 \times rand_2(g_{best} - V_i) \tag{4.17}$$

$$P_{i+1} = P_i + V_{i+1} \tag{4.18}$$

Bu çalışmada problem yol planlama olduğundan, Flores-Caballero ve ark. (2020)'in yaptığına benzer şekilde bir modelleme ile PSO gerçekleştirilmiştir. Amaç fonksiyonu ile başlangıç ve hedef noktaları arasındaki yolun uzunluğu belirlenmiştir. Amaç, uzaydaki ardışık düğümler arasındaki Öklid mesafesini minimum yapmaktır. Başlangıç ve hedef düğümleri hariç, planlanan yolu oluşturan bu düğümler uzayda hareket ettirilerek hedefe daha kısa mesafeli yollar elde edilir. Dolayısıyla optimizasyon için amaç fonksiyonu Denklem 4.19'daki gibi ifade edilir. pos_node'lar uzaydaki n adet

düğümün her birinin koordinat değerlerini ifade eder. pos_node'lar arasındaki fark, ilgili iki pos_node arasındaki öklid mesafesini hesaplar.

$$Path\ Length = \sum_{i=1}^{n-1} |pos\_node_{i+1} - pos\_node_i| \tag{4.19}$$

Ancak Denklem 4.19'daki gibi düğümlerin konumu değiştirilerek en kısa yol aranırken, engeller kısıtlama olarak modellenmelidir. Aksi takdirde karmaşık bir ortamda en kısa yol engellere çarpacak şekilde bulunur. Bu nedenle hem düğümler hem de düğümler arasındaki kenarlar engellere ait pozisyon değerleri ile kesişmemelidir. Kısacası bu çalışma için kullanılan PSO, başlangıç ve hedef düğümleri arasındaki toplam yol uzunluğunu, engel kısıtlamalarına ve uzay sınırlarına bağlı olarak minimize etmektedir. Sonuçta yol planlama çalışması, belirli kısıtlamalara sahip bir küresel (global) optimizasyon problemi olarak formüle edilmiştir. Ayrıca bu çalışma bir İHA uygulaması olduğundan, İHA'nın engellere çarpmaması için, engeller İHA boyutu kadar şişirilmiştir (Minimum engel mesafesi).

Belirli uzamsal kısıtlamalara sahip olarak modellenen yol planlama problemi için, başlangıç ve hedef düğüm arasındaki düğüm konumları sürekli güncellenir, sonuçta yol uzunluğu hesaplanır. Uygulanan PSO için kullanılan parametreler Tablo 4.15'te verilmiştir. Bu çalışmadaki PSO uygulamasında, geleneksel PSO başlangıcında olduğu gibi, parçacıklar başlangıçta rastgele dağıtılmaz. Amaç GDRRT* tarafından oluşturulan yolu geliştirmek olduğundan, ilk çözüm GDRRT*'ın çıktısıdır. Bu nedenle PSO için başlangıç değerleri Şekil 4.20'deki gibi interpole edilmiş yola ait kontrol noktalarıdır (düğümler).

Tablo 4.15. PSO parametre değerleri

PSO Parametreleri	Parametre Değeri
Popülasyon Boyutu (Parçacık Sayısı)	150
İterasyon sayısı	150
İvme Katsayıları (c_1, c_2)	1.5, 1.5
Atalet Katsayısı (w)	1

4.4.3. Öğrenme Temelli Optimal Yol Tahmini (BiLSTM-PSO-GDRRT*)

Daha önce de bahsedildiği üzere RRT* genel olarak hedefe yavaş yakınsama problemine sahiptir. Bizim çalışmamızda tanıtılan GDRRT*, hedefe yakınsamayı daha hızlı gerçekleştirir. Ancak özellikle İHA gibi gerçek zamanlı uygulamalar için görev sırasında yol planlama gerçekleştirilmelidir. Geleneksel yol planlama yaklaşımları ve

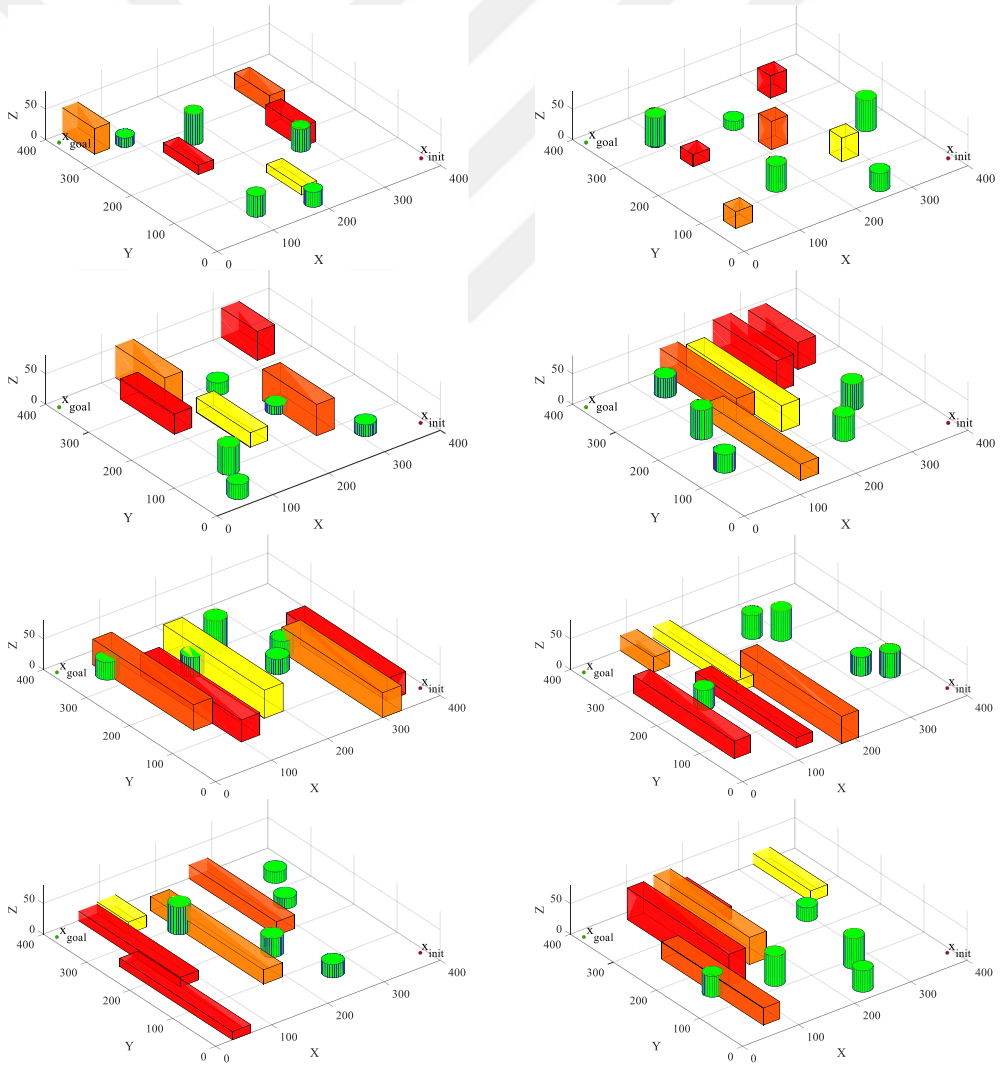
optimizasyon gibi iteratif çözümlerle bunu sağlamak oldukça zordur. Bizim çalışmamızdaki PSO, (Flores-Caballero ve ark., 2020)'deki gibi ilk yol planlama çözümünün üstüne uygulanmıştır. PSO, GDRRT* çıkışını optimize ederek daha kısa yol bulma pahasına, ilave işlem süresine neden olur. Genel olarak optimizasyon tabanlı çalışmalar sadece en kısa yolu bulmayı amaçlar, gerçek zaman gereksinimleri dikkate alınmaz. Ancak bizim çalışmamız, büyük uzaylarda bile gerçek zamanlı uygulamalar için hızlı çözüm sağlayan danışmanlı yapıda yapay zekâ tabanlı bir çözüm sunmuştur.

Öğrenme tabanlı danışmanlı bir çözüm sağlamak için ilk olarak eğitim ve test verileri oluşturulmalıdır. Wang ve ark. (2020b) tarafından 2B ortamlar için gerçekleştirilen Neural-RRT* algoritması, çok sayıda optimal yol verisini oluşturmak için A* algoritmasını kullanmıştır ve daha sonra CNN ile eğitim gerçekleştirmiştir. Büyük ortamlar için A* çok zaman alıcı olduğundan, bu çalışma sınırlıdır. Bizim çalışmamızın bu sınırlamalara bağlı kalmaması için, yani büyük ortamlar için de çözümleri hızlı şekilde üretebilmesi amacıyla GDRRT* kullanılmıştır. Çünkü GDRRT* büyük ortamlarda çözüm sağlar ve RRT*'a göre hedefe daha hızlı yakınsama sağlar. Ancak tabi ki optimum yolu garanti etmez, bu nedenle PSO-GDRRT* büyük haritada hızlı planlanmış yolu optimize eder. Aslında bölüm 4.4.1'de anlatılan GDRRT* ve bölüm 4.4.2'de anlatılan PSO-GDRRT*, yapay zekâ tabanlı yol planlama için bir altyapı sağlar. İlk olarak GDRRT* sayesinde büyük bir uzayda (400x400x80) hızlı bir şekilde yol planlama gerçekleştirilir. Daha sonra bu GDRRT*'nin ürettiği yola, danışmanlı bir ağ için hedef yol oluşturmak amacıyla, yani optimal yol, PSO uygulanır, yani PSO-GDRRT*.

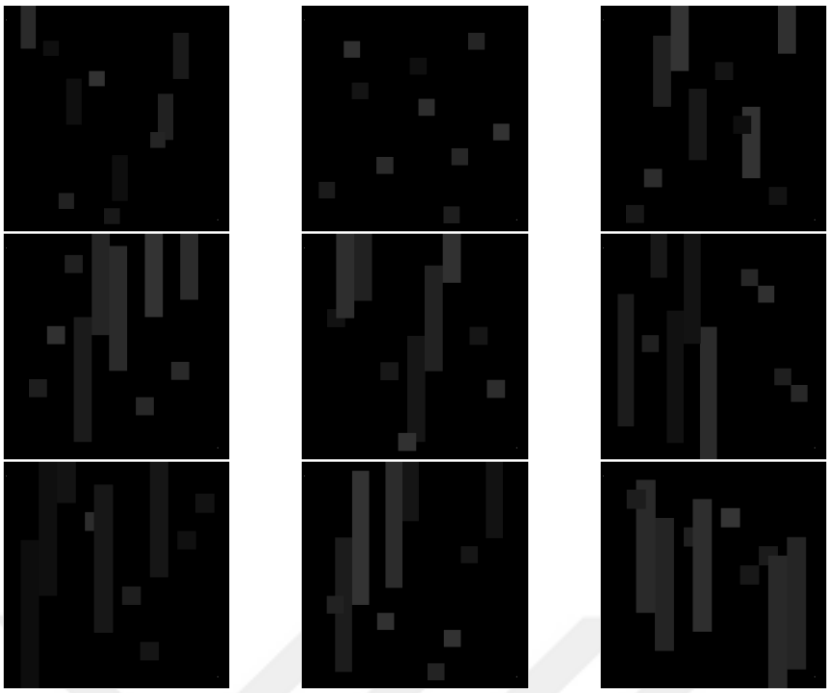
Danışmanlı bir yapıda eğitim gerçekleştirmek için çok sayıda farklı özelliklere sahip ortama ve bu ortamlar üzerinde planlanan optimum yollara ihtiyaç vardır. Dolayısıyla bu çalışmada, eğitim verisi oluşturmak amacıyla 2000 adet ortam verisi oluşturulmuştur. Oluşturulan ortamlarda, engellerin konumları, uzunlukları ve yükseklikleri, uzay sınırlarını aşmayacak şekilde, rastgele belirlenmiştir. Tasarlanan 3B ortamların sınırı $400 \times 400 \times 80$ olarak belirlenmiştir, önceki çalışmalardaki ortamlara göre çok daha büyüktür. Oluşturulan bazı rastgele 3B haritalar Şekil 4.21'de gösterilmiştir.

Önce GDRRT* ile 2000 farklı 3B harita üzerinde yol planlama gerçekleştirilip, daha sonra PSO-GDRRT* ile planlanan yollar optimize edildikten sonra, artık tasarlanan RNN tabanlı mimari ile eğitim ve test adımları gerçekleştirilebilir. Ancak Şekil 4.21'de görülen haritaların ağın giriş katmanına giriş olarak uygulanması için bir dönüşüm gereklidir. Bu nedenle bu çalışma üretilen 3B rastgele ortamları, 2B ortamlara dönüştürür.

Şekil 4.22'deki görüntüler, Şekil 4.21'deki 3B ortamların 2B'ye dönüştürülmesi sonucunda oluşan 2B ortam görüntülerine karşılık gelir. Şekil 4.22'de görüldüğü gibi, dönüşüm işleminde her bir pikselin değeri, 3B ortamdaki engelin yüksekliğine göre belirlenmiştir. 3B ortamdaki engel içermeyen piksellerin karşılığı 2B görüntüde siyahtır, yani değeri 0'dır. 3B ortamda engel varsa, 2B ortamlarda bu engelin piksel olarak karşılığı engelin yüksekliğine göre belirlenir. Engelin değeri maksimum *z* değeri olan 80'e yaklaştıkça, engele ait piksel değeri 255'değerine, yani beyaz renge yaklaşır. Rastgele oluşturulan her bir 3B ortam, Şekil 4.22'deki gibi 2B haritalara dönüştürüldükten sonra eğitim gerçekleştirilir. Dönüşüm işlemi oldukça basit olduğundan, sistem hızını etkilemez. Dönüşüm sırasında, daha güvenilir bir yol planlamak için, 3B ortamdaki silindir yapıdaki engeller, 2B görüntülerde bu silindiri kapsayacak bir kare prizma ile temsil edilir.

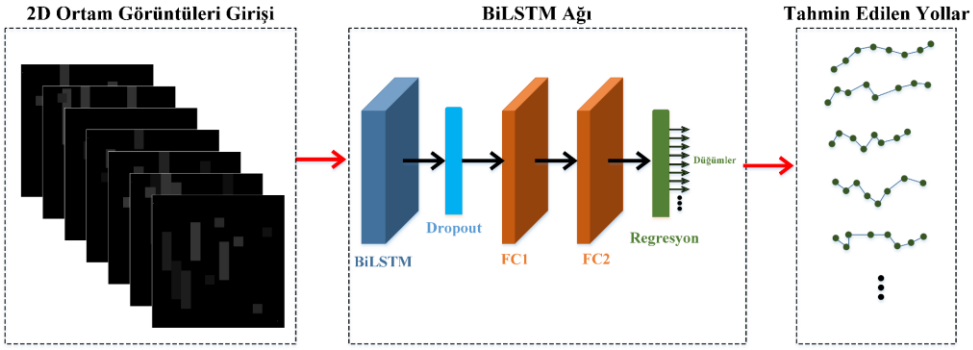


Şekil 4.21. Rastgele oluşturulmuş bazı 3B ortamlar



Şekil 4.22. Rastgele oluşturulmuş 3B ortamların 2B görüntülere dönüştürülmesi

Her bir 3B ortam için, GDRRT*, PSO-GDRRT* ve 2B dönüşümler yapıldıktan sonra, ağ için 2B ortamlar giriş verisi, PSO-GDRRT* çıktısı ise çıkış verisi olarak kullanılır. Eğitim için kullanılan ağ RNN tabanlı BiLSTM katmanıdır ve Şekil 4.23'de gösterilmiştir. RNN tabanlı ağlar özellikle sıralı tahminler gerçekleştirmek için kullanılır ve bu özelliği ile CNN'den ayrılırlar. Yol planlama çıktısı da sıralı yapıda olduğundan uygulama için RNN tabanlı mimari tercih edilmiştir. RNN tabanlı mimariler arasında BiLSTM uzun süreli verileri başarılı bir şekilde tahmin etmesi ile öne çıkar. Bu nedenle BiLSTM katmanı tercih edilmiştir. Sonuç olarak, yol planlama için tasarlanan ağ BiLSTM katmanı, Dropout katmanı, FC ve regresyon katmanlarını içerir. BiLSTM katmanından sonraki Dropout katmanı ile aşırı öğrenme durumunun oluşması engellenir, FC katmanları ise özellikleri nöronlara bağlayarak öğrenme işlemini sağlar. Son olarak regresyon katmanı ile planlanan yola ait düğümlerin 3 eksendeki değerleri tahmin edilir.



Şekil 4.23. Yol planlama tahmini için RNN mimarisi

Tablo 4.16. BiLSTM mimarisi için kullanılan katmanların hiperparametreleri ve eğitim parametreleri

BiLSTM			**FC1**		**FC2**	
Gizli Katman Sayısı	**Aktivasyon Fonksiyonu**	**Kapı Aktivasyon Fonksiyonu**	**Çıkış Boyutu**	**Aktivasyon Fonksiyonu**	**Çıkış Boyutu**	**Aktivasyon Fonksiyonu**
150	$tanh$	$sigmoid$	150	$tanh$	400	$tanh$
Eğitim Parametreleri						
Optimizasyon Algoritması	$\boldsymbol{\beta_1}$	$\boldsymbol{\beta_2}$	**Maks. Epok**	**Küme (Mini Batch) Boyutu**	$\boldsymbol{\alpha}$	ϵ
Adam	0.9	0.999	80	32	0.001	10^{-8}

Şekil 4.23'te gösterilen ağ katmanlarına ait hiperparametreler, ve ağın eğitimi için gerekli olan parametre değerleri Tablo 4.16'da gösterilmiştir. FC2'nin çıkış boyutu 400 olarak ayarlanmıştır. Bu sayede BiLSTM ağının 400 adet sıralı düğüm tahmini üretmesi sağlanır. Bu durum düğümler arasında daha yumuşak yollar üretilmesini sağlayacaktır. Eğitim aşamasında, ağırlıkları hata değerine göre güncellemek için Adam optimizasyon algoritması kullanılır (bkz. Denklem 4.2- Denklem 4.4).

4.4.4. Sonuçlar

Bu bölüm önerilen yöntemlerin sonuçlarını açıklamaktadır. İlk olarak önerilen GDRRT*'ın RRT*'a göre üstünlüğünü göstermek için farklı ortamlarda elde edilen sonuçlar karşılaştırılır. Sonra PSO-GDRRT* tarafından bulunan yolların, GDRRT*'a göre uzunluk açısından daha optimize olduğu gösterilir. Son olarak BiLSTM ile yapılan yol tahminleri, PSO-GDRRT* çıktısındaki yollar ile karşılaştırılır.

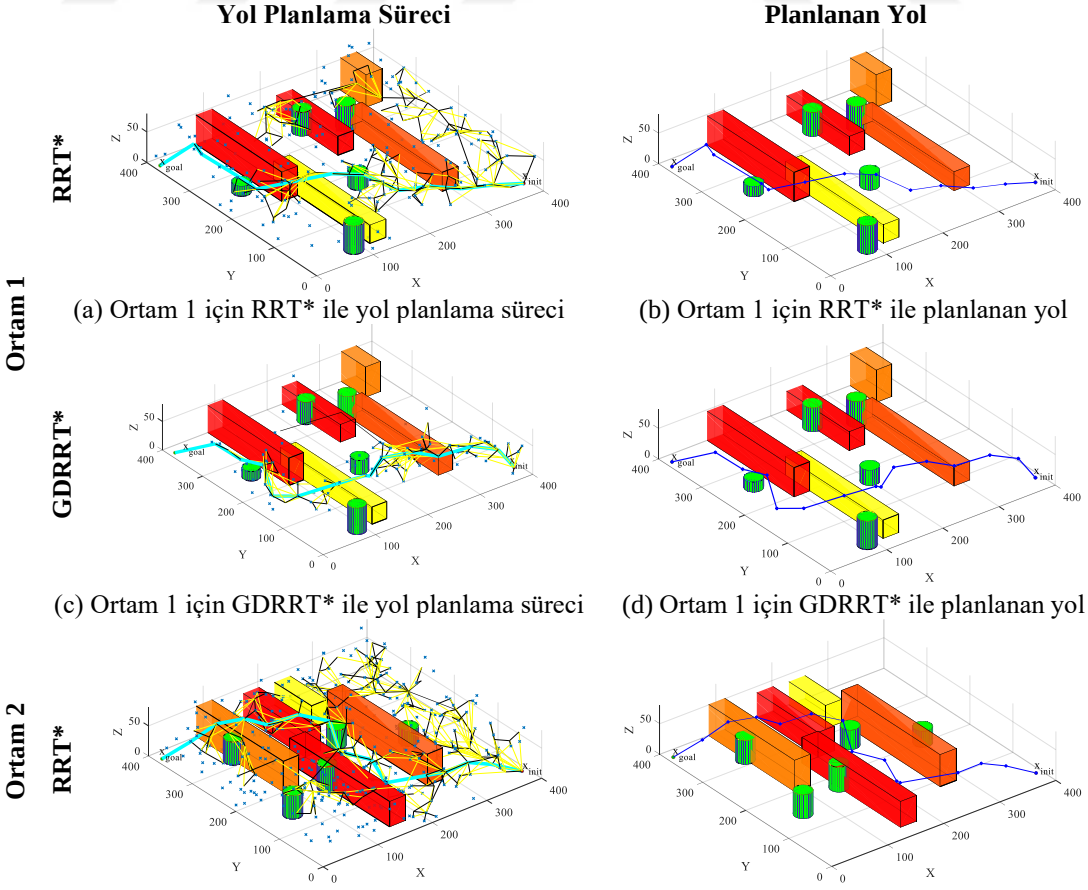
4.4.4.1 RRT* ve GDRRT* Sonuçlarının Karşılaştırılması

Bu bölümde RRT* ve GDRRT* performanslarını analiz edilmiştir. İlk olarak her iki algoritma için kullanılan parametreler Tablo 4.17'de belirtilmiştir. RRT* için bir örnekleme sınırı olmamasına rağmen, $d = 60$ olduğundan GDRRT*, hedefe doğru maksimum 30 piksellik adımlarla yaklaşır.

Tablo 4.17. RRT* ve GDRRT* için parametre ayarları

Parametreler	**RRT***	**GDRRT***
R_{near}	60	60
δ	24	24
Maksimum Düğüm Sayısı	1000	1000
d	-	60
δ_{min}	-	5
$dist\_goal\_min$	-	20
Minimum Engel Mesafesi	-	10

Parametre ayarları yapıldıktan sonra, her iki algoritma 2000 farklı 3B ortam üzerinde uygulanmıştır. Bu bölümde, rastgele seçilen bu ortamlar üzerinden, RRT* ve GDRRT* tarafından planlanan yollar değerlendirilmiştir. GDRRT*'ın önerilme sebebi, RRT*'a göre daha hızlı çözümler üretmektir. Bu nedenle Şekil 4.24'te beş farklı ortam için hem yol planlama süreci ve hem de bu süreç sonunda planlanan yollar gösterilmiştir. Yol planlama süreci ile kastedilen, algoritma başladıktan sonra uygun yol bulunana kadar yapılan örnekleme sayısı ve geçen süredir. GDRRT*'nın en büyük katkısının yol planlama sürecinde olması beklenir. Çünkü hedefe hızlı yakınsayan GDRRT*, RRT*'a göre daha az örnekleme ile hedefe daha hızlı yakınsar. Şekil 4.24'te görüldüğü üzere GDRRT* beş farklı harita için GDRRT*, RRT*'ye göre daha az örnekleme ile yol planlama gerçekleştirmiştir. Yani hedefe daha hızlı yaklaşmıştır. Örnek sayısının artması, sistemin hızını artıracağından, bellek tüketimi ve hızlı çözüm avantajı ile GDRRT*, RRT*'a göre üstünlük sağlamıştır. Şekil 4.24'teki her bir harita için Yol Uzunluğu, Zaman (sn) ve Örnek Sayısı değerleri değerleri Tablo 4.18'te belirtilmiştir. Ayrıca tüm ortamlar için bu değerlerin ortalaması da Tablo 4.19'da gösterilir. Tablolardan da anlaşılacağı üzere, GDRRT*, planlanan yolun daha kısa olmasını garanti etmezken, hedefe daha hızlı ulaşmayı garanti etmektedir.



(a) Ortam 1 için RRT* ile yol planlama süreci

(b) Ortam 1 için RRT* ile planlanan yol

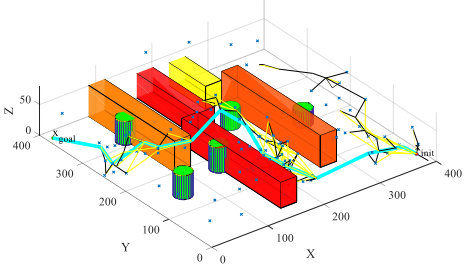
(c) Ortam 1 için GDRRT* ile yol planlama süreci

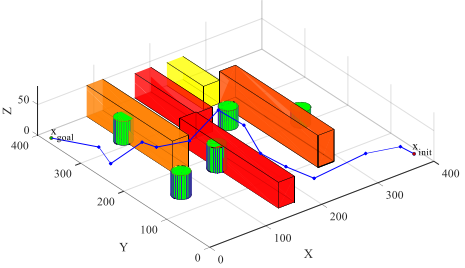
(d) Ortam 1 için GDRRT* ile planlanan yol

(e) Ortam2 için RRT* ile yol planlama süreci

(f) Ortam 2 için RRT* ile planlanan yol

GDRRT*



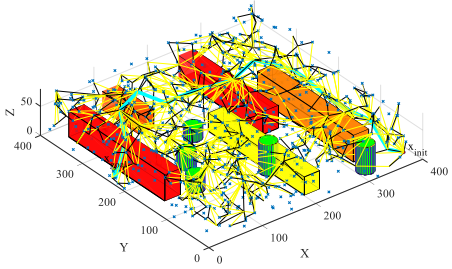


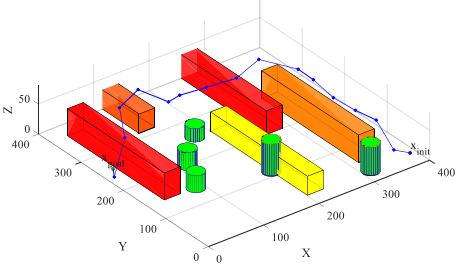
(g) Ortam2 için GDRRT* ile yol planlama süreci

(h) Ortam 2 için GDRRT* ile planlanan yol

Ortam 3

RRT*

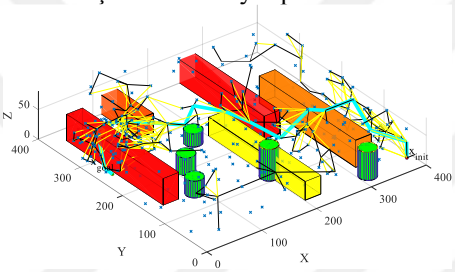


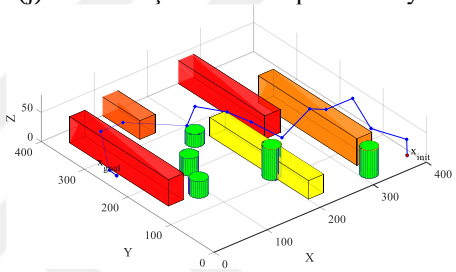


(i) Ortam 3 için RRT* ile yol planlama süreci

(j) Ortam 3 için RRT* ile planlanan yol

GDRRT*



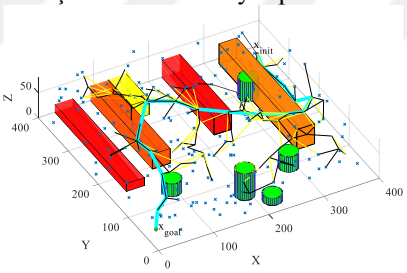


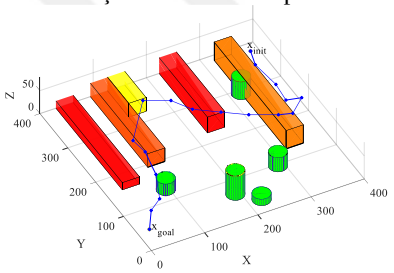
(k) Ortam 3 için GDRRT* ile yol planlama süreci

(l) Ortam 3 için GDRRT* ile planlanan yol

Ortam 4

RRT*

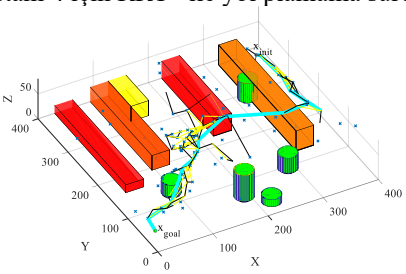


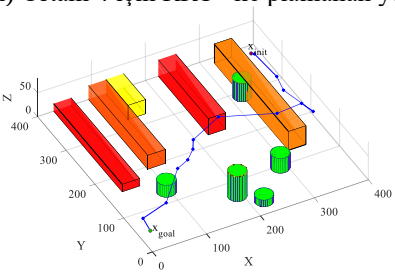


(m) Ortam 4 için RRT* ile yol planlama süreci

(n) Ortam 4 için RRT* ile planlanan yol

GDRRT*



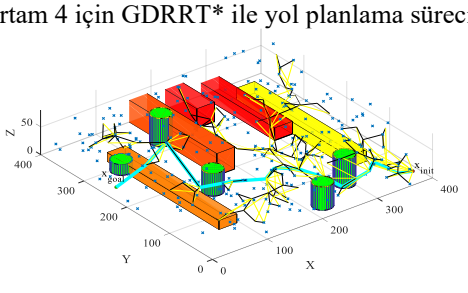


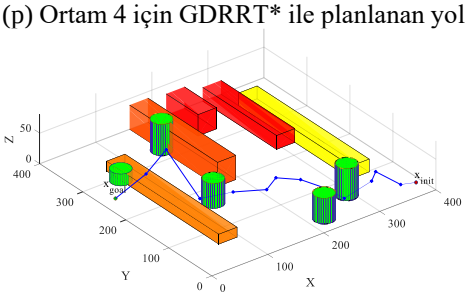
(o) Ortam 4 için GDRRT* ile yol planlama süreci

(p) Ortam 4 için GDRRT* ile planlanan yol

Ortam 5

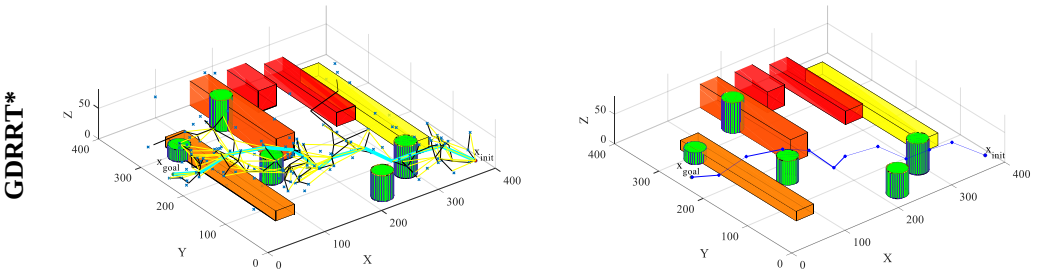
RRT*





(q) Ortam 5 için RRT* ile yol planlama süreci

(r) Ortam 5 için RRT* ile planlanan yol



(s) Ortam 5 için GDRRT* ile yol planlama süreci (t) Ortam 5 için GDRRT* ile planlanan yol

Şekil 4.24. RRT* ve GDRRT* performansının karşılaştırılması için seçilen rastgele 3B ortamlar

Tablo 4.18. Şekil 6'daki beş ortam için RRT* ve GDRRT* karşılaştırması

	Bilgi	RRT*	GDRRT*
Ortam 1	Yol Uzunluğu	623.63	748.65
	Zaman (sn)	13.62	7.62
	Örnek Sayısı	195	90
Ortam 2	Yol Uzunluğu	686.01	732.56
	Zaman (sn)	24.78	5.56
	Örnek Sayısı	364	121
Ortam 3	Yol Uzunluğu	773.50	673.85
	Zaman (sn)	50.38	15.39
	Örnek Sayısı	779	332
Ortam 4	Yol Uzunluğu	820.66	718.49
	Zaman (sn)	9.46	4.83
	Örnek Sayısı	187	94
Ortam 5	Yol Uzunluğu	634.72	554.57
	Zaman (sn)	14.37	6.87
	Örnek Sayısı	278	139

Tablo 4.19. Tüm ortamlar için RRT* ile GDRRT* karşılaştırması

	Ortalama Yol Uzunluğu	Ortalama Zaman (sn)	Ortalama Örnek Sayısı
RRT*	669.05	25.64	398.78
GDRRT*	671.40	8.39	154.66

4.4.4.2 GDRRT*, PSO-GDRRT* ve BiLSTM-PSO-GDRRT* Sonuçlarının Karşılaştırılması

Bu bölümde, GDRRT*, PSO-GDRRT* ve BiLSTM-PSO-GDRRT* algoritmaları sonucunda üretilen yollar karşılaştırılmıştır.

GDRRT*'ın deneysel çalışmalarda, hedefe RRT*'a göre daha hızlı yakınsadığı görülmesine rağmen, planlanan yolun uzunluğu RRT*'a göre daha uzun olabilmektedir. Zaten ne RRT* ne de GDRRT* optimal yolu garanti etmez. Ancak daha kısa yolu kullanarak hedefe ulaşmak ve bu sayede hedefe minimum enerji sarf etmek özellikle İHA'lar için elzemdir. Sınırlı pil kapasitesi nedeniyle, İHA görevi olabildiğince en kısa sürede tamamlamak zorundadır. Bu nedenle, bu uygulama GDRRT* ile planlanan yolu

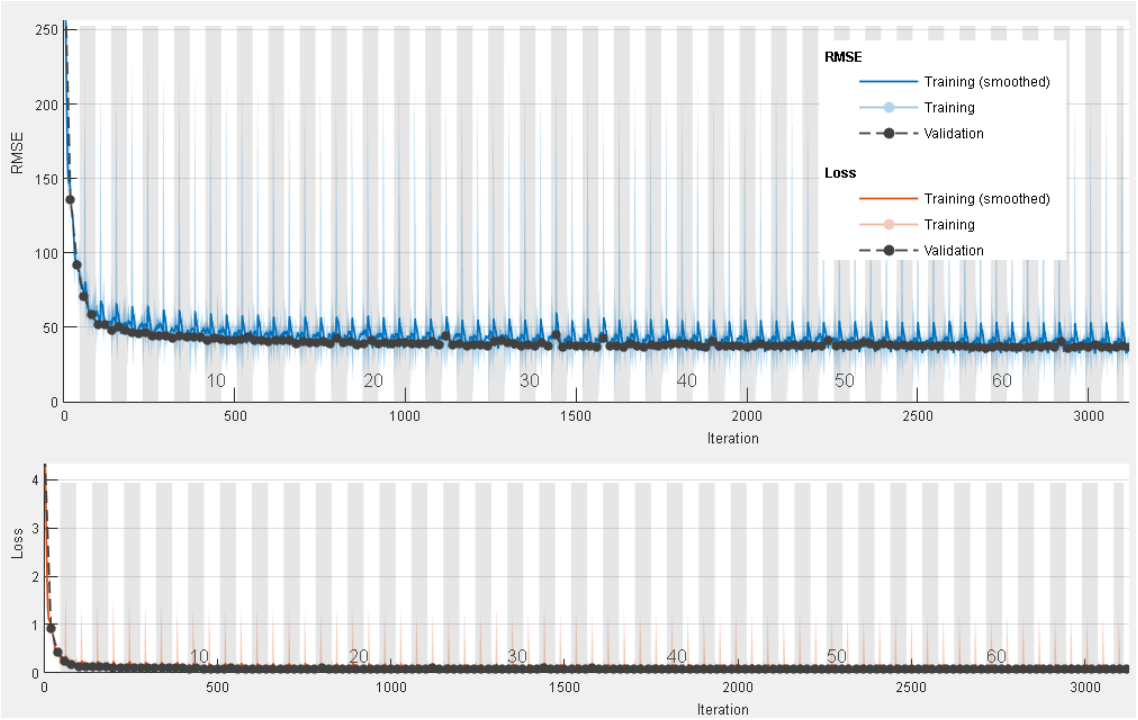
PSO kullanarak, nispeten kısa sürede planlanan yolları uzunluk açısında da optimize eder. Bunun için ilk olarak GDRRT* bir 3B ortam için uygulanır ve ardından PSO bu mevcut yolu optimize eder. Bu durum aslında ilave bir zaman kaybına neden olur. Bu çalışma için PSO iterasyon sayısı ve parçacık sayısı 150 olarak belirlenmiştir. GDRRT* tarafından planlanan yolun optimize edilmesi için geçen süre ortalama olarak 37.51 saniye olarak hesaplanmıştır. Bu durum iterasyon sayısı ve parçacık sayısı arttıkça daha da artacaktır.

PSO daha kısa yollar sağlarken, ilave bir süre gereksinimine neden olmaktadır. Ancak bu çalışmada amaç PSO-GDRRT* çıktısını, hedef yol olarak kullanmak olduğundan, bu ilave süre sadece öğrenme amaçlı veri üretimi aşaması için gereklidir. Yani gerçek zamanlı bir uygulama için PSO uygulanmaz. PSO daha optimize yollar sağladığından, PSO-GDRRT* çıktısını hedef olarak ağ çıkışında kullanmak daha kısa yollar kullanarak eğitilmiş bir ağ sağlayacaktır.

PSO-GDRRT*, bir sonraki işlem olan BiLSTM-PSO-GDRRT*'deki hedef veriyi oluşturmak için kullanılan ara iş parçacığıdır. BiLSTM-PSO-GDRRT*, girişte aldığı 3B ortam bilgisini kullanarak, PSO-GDRRT* ile optimize edilen yolu tahmin etmeyi amaçlar. BiLSTM-PSO-GDRRT* gerçek zamanlı yol planlama uygulamalarını gerçekleştirmek için gereklidir. Gerçek zaman gereksinimi açısından hızlı yol planlama sağlayan GDRRT*'ın katkısı büyüktür, ancak yeterli değildir. Genel olarak şimdiye kadarki çalışmalar yol planlamasını mobil robot harekete geçmeden çözmektedir. Ancak bilinmeyen bir ortama giren bir mobil robotun engellere göre hızlı bir şekilde yol planlama sağlaması arzu edilir. Bu durum ancak eğitilmiş bir ağ kullanarak, bu ağa yeni ortam görüntülerini giriş olarak vermek suretiyle gerçekleştirilebilir. Bu çalışmada rastgele üretilen 2000 adet 3B ortam, basit bir dönüşüm işlemi ile 2B görüntülere dönüştürülmüştür. Sonra bu 2B görüntüleri giriş olarak, PSO-GDRRT* tarafından oluşturulan yolları ise hedef olarak kullanarak, tasarlanan BiLSTM ağı eğitilmiştir. Tüm verinin %80'i eğitim, kalanı test olarak ayrılır. Eğitim işlemi sürecinde RMSE ve kayıp (loss) grafiği Şekil 4.25'te gösterilmiştir. Eğitim sürecinin sonunda, test verisinin RMSE değeri 36.3 olarak hesaplanır. $400 \times 400 \times 80$ boyutlarına sahip bir ortam için bu hata oldukça başarılıdır. Ayrıca BiLSTM tabanlı mimari, giriş olarak verilen her bir 2B ortam görüntüsünü, ortalama 0.019 saniyede yorumlayarak yol planlama tahminini üretmiştir.

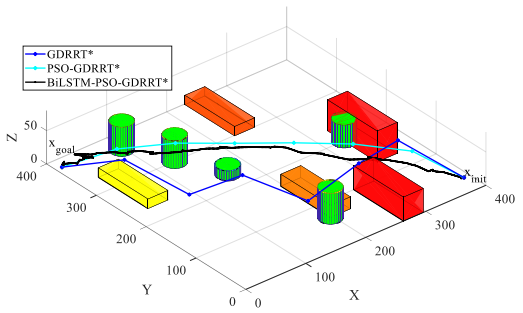
Eğitim süreci tamamlandıktan sonra BiLSTM-PSO-GDRRT* yöntemi ile her bir test ortamı için planlanan yollar üretilmiştir. Şekil 4.26, farklı test ortamları için GDRRT*, PSO-GDRRT* ve BiLSTM-PSO-GDRRT* tarafından planlanan yolları

göstermektedir. Şekil 4.26'da 10 adet farklı 3B ortam üzerinde her üç yöntemin bulduğu yollar gösterilir. Her bir yöntemin bulduğu yolun uzunluğu ve hesaplama süresi de Tablo 4.20'de verilmiştir. Tüm test verisi üzerinde bu üç yöntemin uygulanması sonucu elde edilen ortalama değerler de Tablo 4.21'deki gibidir.

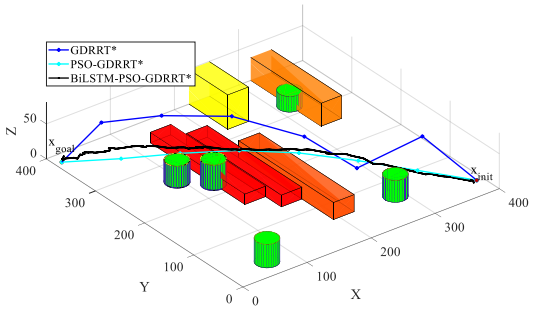


Şekil 4.25. BiLSTM tabanlı yol planlama ağı için eğitim ve test süreçlerinin hata ve kayıp grafiği

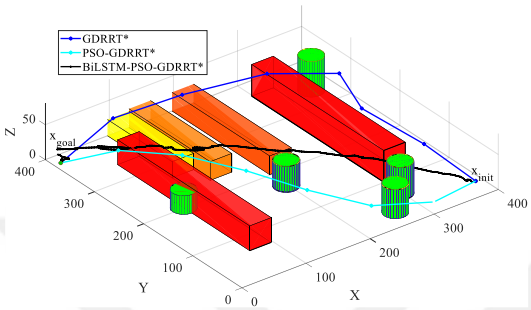
Şekil 4.26, Tablo 4.20 ve Tablo 4.21 incelendiğinde GDRRT*'ın 3B büyük ortamlar için hızlı (ortalama 7.51 sn) bir şekilde yol planladığı görülmektedir. Ancak GDRRT* tarafından planlanan yollar gereksiz zikzaklar ya da optimum olmayan dağılımlar içerir. GDRRT*'ın planladığı yola PSO uygulanması, yolu oldukça optimum hale getirir, sonuçta yol uzunluğu önemli miktarda azalır. Zaten Şekil 4.26 incelendiğinde PSO-GDRRT* çıktısı, genellikle engelleri aşarak doğrudan hedefe yönelir. Ancak Tablo 4.20'den de görüleceği üzere, PSO için gerekli ilave zaman, gerçek zamanlı uygulamalar için PSO-GDRRT*'yı elverişsiz kılar. Ancak PSO-GDRRT* tarafından üretilen yolların genellikle en kısa yola yakın olduğu rahatlıkla görülebilir. Bu nedenle bir ağ için hedef niteliği taşıyabilir. Son olarak BiLSTM-PSO-GDRRT* incelendiğinde yollar daha yumuşak geçişlere sahiptir ve zamansal olarak diğer iki yönteme göre çok üstündür. Bu nedenle hareket halindeki bir İHA için gerçek zamanlı uygulama ihtiyaçlarını karşılar. Ancak her ne kadar hızlı cevap üretse de, yol uzunlukları genel olarak her iki yöntemden daha fazladır. Ayrıca engellerin çok yakınından geçme ya da engellere çarpma gibi durumlar meydana gelebilmektedir.



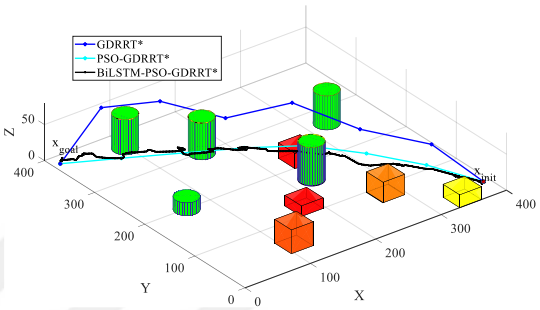
(a) Ortam 1



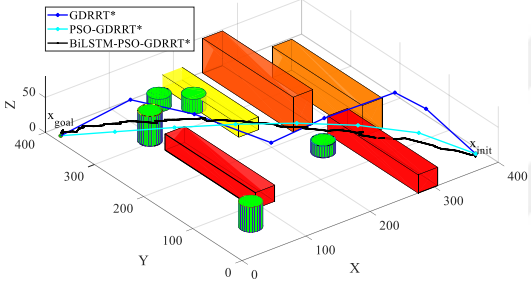
(b) Ortam 2



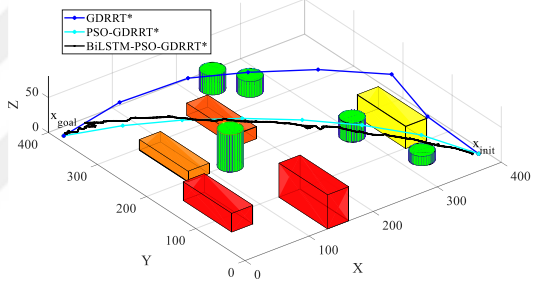
(c) Ortam 3



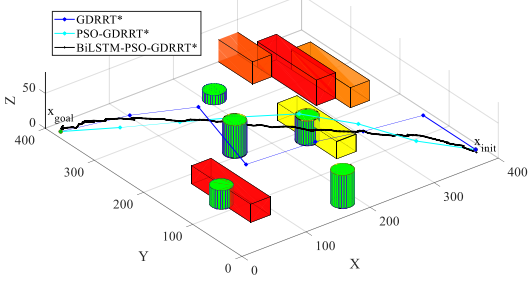
(d) Ortam 4



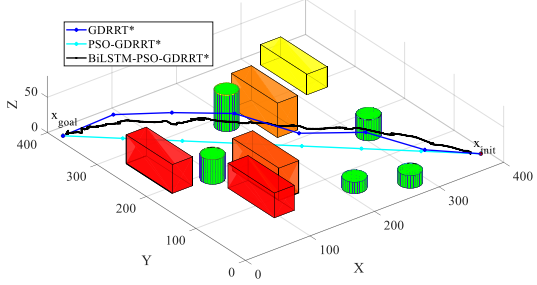
(e) Ortam 5



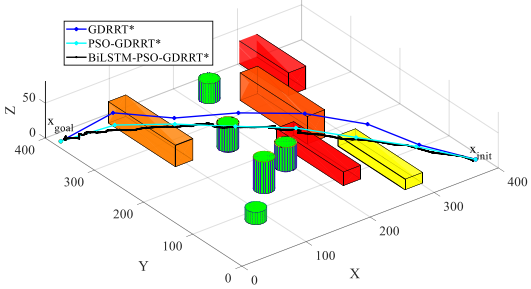
(f) Ortam 6



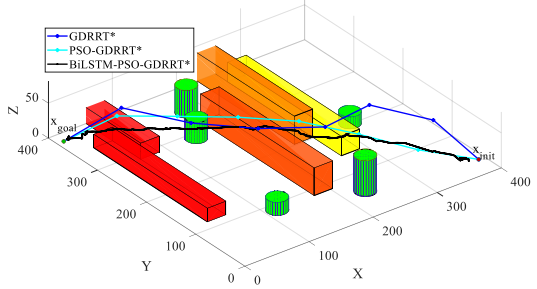
(g) Ortam 7



(h) Ortam 8



(i) Ortam 9



(j) Ortam 10

Şekil 4.26. 10 farklı 3B ortamda GDRRT*, PSO-GDRRT* ve BiLSTM-PSO-GDRRT* için yol planlama sonuçları

Tablo 4.20. Şekil 8'deki 10 ortam için GDRRT*, PSO-GDRRT* ve BiLSTM-PSO-GDRRT* karşılaştırması

Ortam	Bilgi	GDRRT*	PSO-GDRRT*	BiLSTM-PSO-GDRRT*
Şekil 4.26 (a)	Yol Uzunluğu	665.23	529.18	667.35
	Zaman (sn)	7.12	42.54	0.012
Şekil 4.26 (b)	Yol Uzunluğu	629.52	519.72	601.19
	Zaman (sn)	7.21	42.11	0.010
Şekil 4.26 (c)	Yol Uzunluğu	703.45	573.27	753.04
	Zaman (sn)	10.25	45.28	0.013
Şekil 4.26 (d)	Yol Uzunluğu	636.51	523.07	688.54
	Zaman (sn)	6.74	43.51	0.009
Şekil 4.26 (e)	Yol Uzunluğu	598.81	526.85	604.83
	Zaman (sn)	8.42	40.74	0.011
Şekil 4.26 (f)	Yol Uzunluğu	635.78	522.32	651.79
	Zaman (sn)	5.56	39.78	0.012
Şekil 4.26 (g)	Yol Uzunluğu	623.62	522.52	618.42
	Zaman (sn)	3.96	40.45	0.009
Şekil 4.26 (h)	Yol Uzunluğu	544.92	516.41	644.10
	Zaman (sn)	4.39	42.87	0.021
Şekil 4.26 (i)	Yol Uzunluğu	549.03	525.43	616.11
	Zaman (sn)	7.40	43.15	0.014
Şekil 4.26 (j)	Yol Uzunluğu	606.21	528.79	608.09
	Zaman (sn)	11.53	48.73	0.013

Tablo 4.21. Tüm test ortamları için GDRRT*, PSO-GDRRT* ve BiLSTM-PSO-GDRRT* performansının karşılaştırması

Yöntem	Ortalama Yol Uzunluğu	Ortalama Zaman (sn)
GDRRT*	645.84	7.51
PSO-GDRRT*	**524.85**	42.85
BiLSTM-PSO-GDRRT*	669.22	**0.019**

5. SONUÇLAR ve TARTIŞMA

Yapılan bu tez çalışması kapsamında kapalı alanda otonom İHA sistemlerinin geliştirilmesine yönelik 3 farklı uygulama gerçekleştirilmiştir. Her bir uygulama için değerlendirme ve tartışma aşağıdaki gibi ayrı ayrı yapılmıştır.

5.1. Uygulama 1 için Sonuçlar

Sensörlerin yorumlanması ile otonom navigasyon ve keşif, askeriden uzay araştırmalarına kadar en popüler konulardır. Lokalizasyon, navigasyon ve kapalı ortamlarla güvenli etkileşim için robotların iyi bilinen bir sorunudur. Uygulama 1'de sunulan çalışmada, hava aracının konumunu tahmin etmek için kamera ve IMU arasında derin öğrenme tabanlı sensör füzyonu gerçekleştirilmiştir. Sensör füzyonu, hibrit bir yapıda tasarlanmış uçtan uca bir derin öğrenme mimarisi ile gerçekleştirilir. Genel sistemde üç aşamalı öğrenme vardır. İlk olarak, BiLSTM kullanılarak IMU verilerinden zamana bağlı öznitelikler çıkarılır. İkinci olarak kameradan alınan görüntüler CNN ağına verilerek görsel özellikler çıkarılmaktadır. Son olarak, elde edilen bu görsel ve atalet öznitelikleri birleştirildikten sonra, BiLSTM kullanılarak bu birleştirilmiş özniteliklerden konum tahmini başarıyla gerçekleştirilir.

Tasarlanan mimari hem EuRoC veri setinde hem de ROS Gazebo ortamında üretilen veri setimizde başarı tahminler sağlamıştır. Bu da tasarlanan mimarinin İHA ile gerçekleştirilen VIO uygulamalarında kullanılabileceğini kanıtlamıştır. Bu çalışma ile önceki standart V(I)SLAM çalışmalarında olduğu gibi öznitelik çıkarma, optimizasyon, ana kare seçimi gibi zor adımlar kullanılmadan İHA konum tahmini yapılmaktadır. Ayrıca önerilen yöntem gerçek zamanlı İHA uygulamaları için uygundur. Önerilen mimari, BiLSTM kullanımı ve mimari tasarımı açısından farklıdır. Üç eğitimli ağ içerir, görsel ve eylemsiz mimariler füzyon mimarisini eşit olarak beslemektedir.

5.2. Uygulama 1 için Tartışma

Önerilen çalışma başarılı konum tahminleri üretmesine rağmen, daha doğru tahminler üretmek için geliştirilmelidir. Ayrıca bu uygulama sadece konum tahmini yapar. İHA'nın açısal yönelimini de içeren poz tahmini gerçekleştirmez. Gelecek çalışmalar için şu şekilde öneriler düşünülmüştür: Doğruluk değerinin artırılması için IMU ve görsel veriler daha verimli bir şekilde birleştirilebilir. Sağladıkları doğruluklar farklı olduğundan, özellik sayısı görsel ve atalet mimarilerinin hataları ile orantılı olarak belirlenebilir. Yani görsel mimariden çıkarılan özellik sayısının orantılı olarak artırılması

ve böylece daha doğru tahminlerde bulunulması sağlanabilir. Ayrıca görsel ve atalet sensörlerinin yanı sıra farklı sensörler kullanılarak, derin öğrenme tabanlı VIO uygulamaları için daha zengin bilgi füzyonu sağlayarak daha doğru tahminler üretebilir. Bu sayede konumlandırma hatalarının azaltılması sağlanabilir.

5.3. Uygulama 2 için Sonuçlar

Bu uygulama, navigasyon ve kapalı ortamlarla güvenli etkileşim amacıyla İHA'lar için önemli ve aktif bir konu olan poz tahminine yönelik yeni bir yöntem ileri sürmüştür. Bunun için görsel ve atalet sensör bilgilerini, son zamanlarda oldukça popüler olan derin öğrenme tabanlı geliştirilen bir mimari ve Bayes tabanlı GPR yaklaşımı ile işlemiştir. Ölçek kayması minimum ve genelleştirme yeteneği maksimum bir öğrenme tabanlı mimari oluşturmak amacıyla, derin Inception-V3 ağı kullanılarak ham görüntülerin, hareket dinamiklerini gösteren OF görüntü alanlarının ve gürültüsü giderilmiş IMU verilerinin özellikleri çıkarılır. Önerilen yöntemin genel mimarisi üç aşamalı öğrenme içermektedir. Görsel adımda, ham kare ve OF alanları birleştirilir ve Inception-V3 ağına poz tahmini öğretilir. Atalet adımı, öncelikle kareler arasındaki IMU bilgilerini paketler, sonrasında çalışmamızı farklı kılan *nümerikten görüntüye dönüşüm* yapısı, IMU sayısal verilerini görüntüye dönüştürür. Bunun için, öncelikle Savitzky-Golay filtre ile IMU gürültüleri azaltılır, sonrasında tüm IMU verileri normalize edilir. Gürültüsü azaltılmış ve normalize edilmiş iki kare arasındaki IMU verileri parlaklık bilgisi baz alınarak görüntülere dönüştürülür. Bu sayede atalet (veya IMU) görüntüleri elde edilir. Bu IMU görüntüleri daha sonra OF alanları ile birlikte görsel adımdaki gibi Inception-V3 ağına verilerek ikinci öğrenme gerçekleştirilir. Son aşama ise görsel ve atalet adımlarındaki bilgileri birleştirerek GPR ile poz tahmini gerçekleştiren nihai öğrenmedir. GPR tahmininden sonra tekrar uygulanan Savitzky-Golay filtresi, poz tahminlerini daha kararlı hale getirmiştir. Hem EuRoC hem de kendi simülasyon veri setimiz üzerinde, önerilen yöntemin uygulanması sonucunda elde edilen sonuçlar, yöntemimizin sağlamlığını ve başarısını kanıtlamıştır.

5.4. Uygulama 2 için Tartışma

Önerilen uygulama, derin öğrenme tabanlı olması, görsel-atalet bilgilerini içermesi, OF hareket bilgisinden faydalanması gibi önceki çalışmalara benzer özellikler içerir. Ancak genel olarak bu çalışmalar yüksek doğruluk sağlamak için LSTM kullanır. Bizim çalışmamız ise farklı olarak V-I bilgileri derin ve popüler bir mimari olan

Inception-V3 ağı ile değerlendirir. Bunu sağlamak için de 3 kanallı görüntü verisini ham görüntülerden, OF alanlarından, IMU görüntülerinden oluşturur. Ardışık IMU verilerinin görüntüye dönüştürülmesi önerisi, bu ham verilerin derin CNN modellerinden faydalanabileceğini göstermiştir. IMU'nun doğrudan kullanılmaması, öncesinde Savitzky-Golay ile gürültü giderme işlemi de bu çalışmayı önceki çalışmalardan ayırmaktadır. Ayrıca son olarak görsel ve atalet özelliklerini Bayes yaklaşımına dayalı olasılıksal GPR yöntemi ile tahmin etmesi de bu çalışmayı farklı kılmaktadır. Önerilen yöntemin sonuçları incelendiğinde de, çok sayıda popüler yönteme üstünlük sağladığı görülmektedir.

Her ne kadar farklı yaklaşımlar birlikte kullanılarak daha doğru poz tahminleri sağlanabilir olsa da, önemli olan poz tahmin süresi ile İHA hareketi arasında senkronizasyon gerektiren gerçek zamanlı uygulama sınırları dışına çıkmamaktır. Bizim çalışmamızı farklı kılan GPR, Savitzky-Golay filtresi, normalizasyon, IMU verilerinin görüntüye dönüştürülmesi gibi adımlar gerçek zamanlı sistemleri sınırlayacak gecikmelere neden olmaz. GPR yaklaşımını gerçek zamanlı robotik çözümler için kullanan (Kang ve Park, 2015; Yang ve ark., 2017; Jang ve ark., 2020) gibi çalışmalar oldukça fazladır. Ayrıca Savitzky-Golay filtresinin IMU verilerini iyileştirdiği ve diğer filtrelere göre daha hızlı çalıştığı önceki çalışmalarda (Karaim ve ark., 2019) belirtilmiştir. Bizim çalışmamızda da Savitzky-Golay filtresi mimaride iki kere kullanılmış ve her iki yerde hızlı sonuçlar elde edilmiştir. Zaten *nümerikten görüntüye dönüşüm* aşaması karmaşık değildir ve bu anlamda sistemi yavaşlatmaz. Ancak 100 ms 'lik bir gecikme, yüksek frekanslı kamera ve IMU değerlerine sahip bir sistem için gerçek zamanlı bir uygulamayı sınırlar. Bu nedenle, gelecek uygulamalarda, bu problemi çözmek amacıyla tüm kareler değil, sadece ana kareler üzerinde işlemler ve tahminler gerçekleştirilecektir. Bu durumda gerçek zamanlı bir sisteme uygun sonuçların alınabileceği öngörülmektedir.

5.5. Uygulama 3 için Sonuçlar

Şimdiye kadarki yol planlama çalışmaları genellikle tekerlekli mobil robotlar için, küçük 2B ortamlarda geliştirildi. Ancak yeni eğilim, hareket esnekliği ve yaygın kullanım özellikleriyle öne çıkan İHA için yol planlama yöntemleri geliştirmektir. Son uygulama 3B büyük ortamlar için İHA ile yol planlama amaçlı yapay zekâ tabanlı bir çözüm önerisi sunmuştur. Uygulama üç adıma ayrılabilir: ilk adım yeni bir RRT* varyantı olan GDRRT* yöntemini geliştirmiştir. İkinci adımda GDRRT* tarafından üretilen yolu

kısaltmak amacıyla, mevcut yola PSO uygulanmıştır. Son adımda optimize edilmiş GDRRT* tarafından planlanan yolu, hızlı bir şekilde tahmin etmek için BiLSTM katmanına bağlı bir yapay zekâ uygulaması gerçekleştirilmiştir.

İlk adımda önerilen GDRRT* için odak, hedefe daha hızlı yakınsayan bir örnekleme stratejisi geliştirmektir. Bu nedenle her bir örneklemenin, mevcut ağaç yapısındaki hedefe en yakın düğüme göre kontrollü olarak gerçekleştirilmesi sağlanmıştır. GDRRT*, RRT*'nin değiştirilmesiyle oluşturulmuştur ve sonuçta GDRRT*, genellikle RRT*'ye hız açısından büyük üstünlük sağlamıştır. Ancak GDRRT* hızı gerçek zamanlı hareket halindeki bir İHA'nın anlık yol planlaması için yeterli değildir. Üstelik planlanan yollar optimallikten oldukça uzaktır. Gerçek zamanlı hızlı bir çözüm için bu çalışma, 3B harita için bir ağ tahmini ile hız problemini çözer. Ancak öncesinde ağın eğitimi için optimum yollar gereklidir. İkinci adımda, GDRRT* yolunu kısaltmak için, GDRRT*'den sonra PSO uygulaması gerçekleştirmiştir. PSO için ilk değerler ham GDRRT* çıktısındaki düğümlerdir. Daha sonra her bir iterasyonda bu düğümlerin konumu engel kısıtlaması dikkate alınarak güncellenir. İterasyon sonunda GDRRT*'a göre hedefe çok daha kısa yolla ulaşan yollar elde edilmiştir. PSO-GDRRT* sadece yapay zekâ için hedef yol verisi üretimi için kullanılmıştır. Aksi takdirde PSO, mevcut yolu optimize etmek için çok büyük ilave süre gerektirir. İlk iki adım, bu çalışma için üretilen rastgele 2000 adet 3B ortam için uygulanmıştır. Giriş (ortamlar) görüntüleri ve PSO-GDRRT* ile üretilen hedef yollar artık mevcut olduğundan, son adımda danışmanlı yapıda tasarlanan BiLSTM tabanlı mimari ile yol planlama tahmini gerçekleştirilmiştir. Öncesinde 3B ortamlar basit bir dönüşüm ile, engel konumlarını ve yüksekliklerini içeren 2B görüntülere dönüştürülmüştür. Bu 2B görüntüler BiLSTM ağına giriş olarak verilmiştir, hedef ise PSO-GDRRT* ile üretilen yollardır. Gerçekleştirilen eğitim sonucunda, eğitilmiş ağ test ortamları üzerinde uygulanır ve performanslar önceki iki adım ile karşılaştırılmıştır. BiLSTM-PSO-GDRRT* sonuçları, hareket halindeki bir İHA için gerçek zamanlı bir yol planlamayı başarabilecek ölçüde, hızlı bir şekilde, sıralı düğümleri tahmin etmeyi başarmıştır.

5.6. Uygulama 3 için Tartışma

Bu uygulamanın iki en önemli katkısı GDRRT*'ı tanıtması ve BiLSTM-PSO-GDRRT* ile hızlı yol planlama sağlamasıdır. GDRRT* genel olarak RRT*'a göre daha hızlı çözüm sağlar. Ancak engel açısından çok karmaşık ortamlarda hedefe yaklaşmak için engellere çok yakınlaşabilir, bu da çarpma ve engelden kurtulamama ihtimalini

artırır. GDRRT* tamamen hedefe doğru yakınsamaya odaklandığından, hedefe yakın ve dar bir serbest (engelsiz) uzayda takılıp kalabilir. Ayrıca d değeri eğer çok küçük seçilirse, GDRRT* hedefe çok yavaş bir yakınsar. Hareket halindeki bir İHA'nın gerçek zamanlı yol planlama uygulamalarını mümkün kılmak için geliştirilen BiLSTM-PSO-GDRRT*'nın hızlı planlama sağlaması, bu uygulamayı değerli kılmaktadır. Ancak BiLSTM-PSO-GDRRT* ile tahmin edilen yollar, uzunluk olarak arzu edilen başarıyı sağlamamıştır. Bunun için çok daha fazla sayıda optimal yolla ağ eğitilmelidir. Ancak çok sayıda veri için, çok sayıda ortam ve çok sayıda optimal yollar oluşturulmalıdır. Bu da zaman ve bellek tüketimi açısından bir problemdir. Bizim çalışmamızda GDRRT* ve PSO-GDRRT* yöntemleri ile 2000 adet ortamda yol planlama gerçekleştirilmesi ve yolların optimizasyon süreci yaklaşık 36 saat sürmüştür. Ayrıca BiLSTM-PSO-GDRRT* çıkışındaki yolların engele çarpma ihtimali vardır. Üstelik bu durum kara kutu yapısındaki eğitilmiş bir ağ için önlenebilir değildir. Engele hızlı ulaşma pahasına engele çarpma ve bunun sonucunda İHA'nın hasar görmesi istenen bir durum değildir. Bunun için gelecekte Açıklanabilir Yapay Zeka (Explainable Artificial Intelligence) çözümleri ile, yapay zeka ile tahmin edilen yolların daha güvenilir olması sağlanabilir.

5.7. Tüm Uygulamaların Değerlendirilmesi

Genel olarak tez kapsamında gerçekleştirilen tüm uygulamalar, iç ortamda otonom görevler gerçekleştirebilecek bir İHA'ya yönelik problemleri çözmeye odaklanmıştır. Bu anlamda geçmişte yapılan çalışmalardan farklı olarak yeni metodolojik katkılar ileri sürülmüştür. Tüm katkıların yapay zekâ tabanlı olması, önerilen uygulamaların modern çözümler içerdiğini doğrulamaktadır. Geleneksel ya da geometrik tabanlı yöntemler her ne kadar doğruya yakın sonuçlar üretse de gerçek dünya davranışını tamamen yansıtmaz. Bu anlamda bu tezin içerdiği uygulamalar, kapalı ortamdaki bir İHA'nın geometrik ve karmaşık hesaplamalara minimum gereksinim duyarak otonom görevler gerçekleştirilmesini mümkün kılmaktadır. İlk uygulama görsel ve atalet verilerinin farklı bir füzyonunu içeren derin bir mimari ile öne çıkmıştır. İkinci uygulama IMU verisini görüntü olarak değerlendirerek, farklı bir derin mimarinin yanında farklı bir bakış açısı ile iç ortamda konumlandırma problemini başarılı bir şekilde çözmüştür. Son uygulama ise hız ve doğruluk açısından oldukça yüksek performans gösteren bir yol planlama metodu önermiştir. Her üç uygulama da iç ortamda otonom bir görev gerçekleştirecek İHA için hayati önem taşıyan işlem parçalarıdır. Her uygulama benzer nitelikli önceki uygulamalar/yöntemlerle karşılaştırılmıştır. Sonuçlar, önerilen

yöntemlerin önceki çalışmaların büyük bir kısmına göre daha üstün olduğunu göstermiştir. Bu anlamda bu tez kapsamındaki uygulamalar, otonom İHA çözümleri için önemli metodolojik katkılar sağlamaktadır.

KAYNAKLAR

Adiyatov, O. ve Varol, H. A., 2013, Rapidly-exploring random tree based memory efficient motion planning, *2013 IEEE International Conference on Mechatronics and Automation*, 354-359.

Aggarwal, S. ve Kumar, N., 2020, Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges, *Computer communications*, 149, 270-299.

Ajeil, F. H., Ibraheem, I. K., Sahib, M. A. ve Humaidi, A. J., 2020, Multi-objective path planning of an autonomous mobile robot using hybrid PSO-MFB optimization algorithm, *Applied soft computing*, 89, 106076.

Alatise, M. B. ve Hancke, G. P., 2020, A Review on Challenges of Autonomous Mobile Robot and Sensor Fusion Methods, *IEEE Access*, 8, 39830-39846.

Almalioglu, Y., Turan, M., Sari, A. E., Saputra, M. R. U., de Gusmão, P. P., Markham, A. ve Trigoni, N., 2019, Selfvio: Self-supervised deep monocular visual-inertial odometry and depth estimation, *arXiv preprint arXiv:1911.09968*.

Alotaibi, E. T., Alqefari, S. S. ve Koubaa, A., 2019, Lsar: Multi-uav collaboration for search and rescue missions, *IEEE Access*, 7, 55817-55832.

Ardakani, A. A., Kanafi, A. R., Acharya, U. R., Khadem, N. ve Mohammadi, A., 2020, Application of deep learning technique to manage COVID-19 in routine clinical practice using CT images: Results of 10 convolutional neural networks, *Computers in biology and medicine*, 121, 103795.

Aslan, M. F., Durdu, A. ve SABANCI, K., 2017, Shopping Robot That Make Real Time Color Tracking Using Image Processing Techniques, *International Journal of Applied Mathematics Electronics and Computers*, 5 (3), 62-66.

Aslan, M. F., Durdu, A., Yusefi, A., Sabanci, K. ve Sungur, C., 2021a, A tutorial: Mobile robotics, SLAM, bayesian filter, keyframe bundle adjustment and ROS applications, *Robot Operating System (ROS)*, 227-269.

Aslan, M. F., Sabanci, K. ve Durdu, A., 2021b, A CNN-based novel solution for determining the survival status of heart failure patients with clinical record data: numeric to image, *Biomedical Signal Processing and Control*, 68, 102716.

Aslan, M. F., Durdu, A., Sabanci, K., Ropelewska, E. ve Gültekin, S. S., 2022, A Comprehensive Survey of the Recent Studies with UAV for Precision Agriculture in Open Fields and Greenhouses, *Applied Sciences*, 12 (3), 1047.

Ayache, N. ve Faugeras, O. D., 1988, Building, registrating, and fusing noisy visual maps, *The international journal of Robotics Research*, 7 (6), 45-65.

Bailey, T. ve Durrant-Whyte, H., 2006, Simultaneous localization and mapping (SLAM): Part II, *IEEE Robotics & Automation Magazine*, 13 (3), 108-117.

Bak, J.-H., Hwang, S. W., Yoon, J., Park, J. H. ve Park, J.-O., 2019, Collision-free path planning of cable-driven parallel robots in cluttered environments, *Intelligent Service Robotics*, 12 (3), 243-253.

Ban, X., Wang, H., Chen, T., Wang, Y. ve Xiao, Y., 2020, Monocular visual odometry based on depth and optical flow using deep learning, *IEEE Transactions on Instrumentation and Measurement*, 70, 1-19.

Barrau, A. ve Bonnabel, S., 2015, An EKF-SLAM algorithm with consistency properties, *arXiv preprint arXiv:1510.06263*.

Bavle, H., De La Puente, P., How, J. P. ve Campoy, P., 2020, VPS-SLAM: Visual planar semantic SLAM for aerial robotic systems, *IEEE Access*, 8, 60704-60718.

Bay, H., Tuytelaars, T. ve Gool, L. V., 2006, Surf: Speeded up robust features, *European conference on computer vision*, 404-417.

Bency, M. J., Qureshi, A. H. ve Yip, M. C., 2019, Neural path planning: Fixed time, near-optimal path generation via oracle imitation, *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3965-3972.

Blanco-Claraco, J.-L., Moreno-Duenas, F.-A. ve González-Jiménez, J., 2014, The Málaga urban dataset: High-rate stereo and LiDAR in a realistic urban scenario, *The international journal of Robotics Research*, 33 (2), 207-214.

Bloesch, M., Omari, S., Hutter, M. ve Siegwart, R., 2015, Robust visual inertial odometry using a direct EKF-based approach, *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 298-304.

Brossard, M., Bonnabel, S. ve Barrau, A., 2020, Denoising imu gyroscopes with deep learning for open-loop attitude estimation, *IEEE Robotics and Automation Letters*, 5 (3), 4796-4803.

Brunner, M., Brüggemann, B. ve Schulz, D., 2013, Hierarchical rough terrain motion planning using an optimal sampling-based method, *2013 IEEE International Conference on Robotics and Automation*, 5539-5544.

Bueno, M., González-Jorge, H., Martínez-Sánchez, J., Díaz-Vilariño, L. ve Arias, P., 2016, Evaluation of point cloud registration using Monte Carlo method, *Measurement*, 92, 264-270.

Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M. W. ve Siegwart, R., 2016, The EuRoC micro aerial vehicle datasets, *The international journal of Robotics Research*, 35 (10), 1157-1163.

Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I. ve Leonard, J. J., 2016, Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age, *IEEE Transactions on Robotics*, 32 (6), 1309-1332.

Campos, C., Elvira, R., Rodríguez, J. J. G., Montiel, J. M. M. ve Tardós, J. D., 2021a, ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM, *IEEE Transactions on Robotics*, 1-17.

Campos, C., Elvira, R., Rodríguez, J. J. G., Montiel, J. M. M. ve Tardós, J. D., 2021b, ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM, *IEEE Transactions on Robotics*, 37 (6), 1874-1890.

Canziani, A., Paszke, A. ve Culurciello, E., 2016, An analysis of deep neural network models for practical applications, *arXiv preprint arXiv:1605.07678*.

Cao, L., Ling, J. ve Xiao, X., 2020, Study on the influence of image noise on monocular feature-based visual slam based on ffdnet, *Sensors*, 20 (17), 4922.

Chaari, I., Koubaa, A., Bennaceur, H., Ammar, A., Alajlan, M. ve Youssef, H., 2017, Design and performance analysis of global path planning techniques for autonomous mobile robots in grid environments, *International Journal of Advanced Robotic Systems*, 14 (2), 1729881416663663.

Cheein, F. A., Steiner, G., Paina, G. P. ve Carelli, R., 2011, Optimized EIF-SLAM algorithm for precision agriculture mapping based on stems detection, *Computers and Electronics in Agriculture*, 78 (2), 195-207.

Chen, C., Zhu, H., Li, M. ve You, S., 2018, A review of visual-inertial simultaneous localization and mapping from filtering-based and optimization-based perspectives, *Robotics*, 7 (3), 45.

Chen, C., Zhu, H., Wang, L. ve Liu, Y., 2019, A Stereo Visual-Inertial SLAM Approach for Indoor Mobile Robots in Unknown Environments Without Occlusions, *IEEE Access*, 7, 185408-185421.

Chen, C., Wang, B., Lu, C. X., Trigoni, N. ve Markham, A., 2020, A survey on deep learning for localization and mapping: Towards the age of spatial machine intelligence, *arXiv preprint arXiv:2006.12567*.

Chen, Y., Tang, J., Feng, Z., Hakala, T., Hyyppä, J., Zhou, C., Tang, L. ve Li, C., 2017, Possibility of Applying SLAM-Aided LiDAR in Deep Space Exploration, Cham, 239-248.

Chen, Y., 2021, Autonomous Navigation and Planning Technology for Quad-rotors Unmanned Aerial Vehicle (UAV) System, *University Of Technology Sydney*.

Chiang, K.-W., Hou, H., Niu, X. ve El-Sheimy, N., 2004, Improving the positioning accuracy of DGPS/MEMS IMU integrated systems utilizing cascade de-noising algorithm, *Proceedings of the 17th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS 2004)*, 809-818.

Chiu, C.-Y. ve Sastry, S. S., 2021, Simultaneous Localization and Mapping: A Rapprochement of Filtering and Optimization-Based Approaches, *University of California*.

Choi, S. Y. ve Cha, D., 2019, Unmanned aerial vehicles using machine learning for autonomous flight; state-of-the-art, *Advanced Robotics*, 33 (6), 265-277.

Chollet, F., 2017, Xception: Deep learning with depthwise separable convolutions, *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1251-1258.

Clark, R., Wang, S., Wen, H., Markham, A. ve Trigoni, N., 2017, Vinet: Visual-inertial odometry as a sequence-to-sequence learning problem, *Proceedings of the AAAI Conference on Artificial Intelligence*.

Clement, L. E., Peretroukhin, V., Lambert, J. ve Kelly, J., 2015, The Battle for Filter Supremacy: A Comparative Study of the Multi-State Constraint Kalman Filter and the Sliding Window Filter, *2015 12th Conference on Computer and Robot Vision*, 23-30.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S. ve Schiele, B., 2016, The cityscapes dataset for semantic urban scene understanding, *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3213-3223.

Costante, G. ve Ciarfuglia, T. A., 2018, LS-VO: Learning Dense Optical Subspace for Robust Visual Odometry Estimation, *IEEE Robotics and Automation Letters*, 3 (3), 1735-1742.

Couturier, A. ve Akhloufi, M. A., 2021, A review on absolute visual localization for UAV, *Robotics and Autonomous Systems*, 135, 103666.

Craig, J. J., 2009, Introduction to robotics: mechanics and control, 3/E, Pearson Education India, p.

Das, A. ve Dubbelman, G., 2018, An experimental study on relative and absolute pose graph fusion for vehicle localization, *2018 IEEE Intelligent Vehicles Symposium (IV)*, 630-635.

Davis, L., 1991, Handbook of genetic algorithms.

Davison, A. J., 1999, Mobile robot navigation using active vision, *Advances in Scientific Philosophy Essays in Honour of*.

Davison, A. J., Reid, I. D., Molton, N. D. ve Stasse, O., 2007, MonoSLAM: Real-time single camera SLAM, *IEEE transactions on pattern analysis and machine intelligence*, 29 (6), 1052-1067.

Dellaert, F. ve Kaess, M., 2006, Square Root SAM: Simultaneous localization and mapping via square root information smoothing, *The international journal of Robotics Research*, 25 (12), 1181-1203.

Delmerico, J. ve Scaramuzza, D., 2018, A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots, *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2502-2509.

Depaola, R., Chimento, C., Anderson, M. L., Brink, K. ve Willis, A., 2018, Uav navigation with computer vision–flight testing a novel visual odometry technique, *2018 AIAA Guidance, Navigation, and Control Conference*, 2102.

Dhruv, P. ve Naskar, S., 2020, Image Classification Using Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN): A Review, Singapore, 367-381.

Di Paola, D., Milella, A., Cicirelli, G. ve Distante, A., 2010, An autonomous mobile robotic system for surveillance of indoor environments, *International Journal of Advanced Robotic Systems*, 7 (1), 8.

Dijkstra, E. W., 1959, A note on two problems in connexion with graphs, *Numerische mathematik*, 1 (1), 269-271.

Ding, Y., Sohn, J. H., Kawczynski, M. G., Trivedi, H., Harnish, R., Jenkins, N. W., Lituiev, D., Copeland, T. P., Aboian, M. S. ve Mari Aparici, C., 2019, A deep learning model to predict a diagnosis of Alzheimer disease by using 18F-FDG PET of the brain, *Radiology*, 290 (2), 456-464.

Dissanayake, M. G., Newman, P., Clark, S., Durrant-Whyte, H. F. ve Csorba, M., 2001, A solution to the simultaneous localization and map building (SLAM) problem, *IEEE Transactions on robotics and automation*, 17 (3), 229-241.

Dong, N., Zhao, L., Wu, C. H. ve Chang, J. F., 2020, Inception v3 based cervical cell classification combined with artificially extracted features, *Applied Soft Computing*, 93, 106311.

Dorigo, M., Maniezzo, V. ve Colorni, A., 1996, Ant system: optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26 (1), 29-41.

Dowling, L., Poblete, T., Hook, I., Tang, H., Tan, Y., Glenn, W. ve Unnithan, R. R., 2018, Accurate indoor mapping using an autonomous unmanned aerial vehicle (UAV), *arXiv preprint arXiv:1808.01940*.

Duan, C., Junginger, S., Huang, J., Jin, K. ve Thurow, K., 2019, Deep learning for visual SLAM in transportation robotics: a review, *Transportation Safety and Environment*, 1 (3), 177-184.

Durrant-Whyte, H. ve Bailey, T., 2006, Simultaneous localization and mapping: part I, *IEEE Robotics & Automation Magazine*, 13 (2), 99-110.

Durrant-Whyte, H. F., 1988, Uncertain geometry in robotics, *IEEE Journal on Robotics and Automation*, 4 (1), 23-31.

Elmokadem, T. ve Savkin, A. V., 2021, Towards Fully Autonomous UAVs: A Survey, *Sensors*, 21 (18), 6223.

Engel, J., Sturm, J. ve Cremers, D., 2013, Semi-dense visual odometry for a monocular camera, *Proceedings of the IEEE international conference on computer vision*, 1449-1456.

Engel, J., Schöps, T. ve Cremers, D., 2014, LSD-SLAM: Large-Scale Direct Monocular SLAM, Cham, 834-849.

Faessler, M., Fontana, F., Forster, C., Mueggler, E., Pizzoli, M. ve Scaramuzza, D., 2016, Autonomous, vision-based flight and live dense 3D mapping with a quadrotor micro aerial vehicle, *Journal of Field Robotics*, 33 (4), 431-450.

Faust, A., Palunko, I., Cruz, P., Fierro, R. ve Tapia, L., 2017, Automated aerial suspended cargo delivery through reinforcement learning, *Artificial Intelligence*, 247, 381-398.

Ferrera, M., Eudes, A., Moras, J., Sanfourche, M. ve Besnerais, G. L., 2021, OV^2SLAM: A Fully Online and Versatile Visual SLAM for Real-Time Applications, *IEEE Robotics and Automation Letters*, 6 (2), 1399-1406.

Flores-Caballero, G., Rodríguez-Molina, A., Aldape-Pérez, M. ve Villarreal-Cervantes, M. G., 2020, Optimized path-planning in continuous spaces for unmanned aerial vehicles using meta-heuristics, *IEEE Access*, 8, 176774-176788.

Forster, C., Pizzoli, M. ve Scaramuzza, D., 2014, SVO: Fast semi-direct monocular visual odometry, *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 15-22.

Forster, C., Carlone, L., Dellaert, F. ve Scaramuzza, D., 2016a, On-Manifold Preintegration for Real-Time Visual--Inertial Odometry, *IEEE Transactions on Robotics*, 33 (1), 1-21.

Forster, C., Zhang, Z., Gassner, M., Werlberger, M. ve Scaramuzza, D., 2016b, SVO: Semidirect visual odometry for monocular and multicamera systems, *IEEE Transactions on Robotics*, 33 (2), 249-265.

Fraser, C., 2018, SLAM, SfM and photogrammetry: What's in a name, *Proceedings of the ISPRS Technical Comission II: Symposium*.

Gammell, J. D., Srinivasa, S. S. ve Barfoot, T. D., 2014, Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic, *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2997-3004.

Geiger, A., Lenz, P., Stiller, C. ve Urtasun, R., 2013, Vision meets robotics: The kitti dataset, *The international journal of Robotics Research*, 32 (11), 1231-1237.

Gers, F. A., Schraudolph, N. N. ve Schmidhuber, J., 2002, Learning precise timing with LSTM recurrent networks, *Journal of machine learning research*, 3 (Aug), 115-143.

Gogul, I. ve Kumar, V. S., 2017, Flower species recognition system using convolution neural networks and transfer learning, *2017 fourth international conference on signal processing, communication and networking (ICSCN)*, 1-6.

Gomez-Ojeda, R., Moreno, F.-A., Zuniga-Noël, D., Scaramuzza, D. ve Gonzalez-Jimenez, J., 2019, PL-SLAM: A stereo SLAM system through the combination of points and line segments, *IEEE Transactions on Robotics*, 35 (3), 734-746.

Graves, A. ve Schmidhuber, J., 2005, Framewise phoneme classification with bidirectional LSTM and other neural network architectures, *Neural networks*, 18 (5-6), 602-610.

Grisetti, G., Kummerle, R., Stachniss, C. ve Burgard, W., 2010, A tutorial on graph-based SLAM, *IEEE Intelligent Transportation Systems Magazine*, 2 (4), 31-43.

Gul, F., Rahiman, W. ve Nazli Alhady, S. S., 2019, A comprehensive study for robot navigation techniques, *Cogent Engineering*, 6 (1), 1632046.

Guo, S., Zhang, X., Zheng, Y. ve Du, Y., 2020, An autonomous path planning model for unmanned ships based on deep reinforcement learning, *Sensors*, 20 (2), 426.

Gupta, L., Jain, R. ve Vaszkun, G., 2016, Survey of Important Issues in UAV Communication Networks, *IEEE Communications Surveys & Tutorials*, 18 (2), 1123-1152.

Gurturk, M., Yusefi, A., Aslan, M. F., Soycan, M., Durdu, A. ve Masiero, A., 2021, The YTU dataset and recurrent neural network based visual-inertial odometry, *Measurement*, 184, 109878.

Han, J., 2019, An efficient approach to 3D path planning, *Information Sciences*, 478, 318-330.

Han, L., Lin, Y., Du, G. ve Lian, S., 2019, Deepvio: Self-supervised deep learning of monocular visual inertial odometry using 3d geometric constraints, *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 6906-6913.

Han, S. ve Xi, Z., 2020, Dynamic Scene Semantics SLAM Based on Semantic Segmentation, *IEEE Access*, 8, 43563-43570.

Hart, P. E., Nilsson, N. J. ve Raphael, B., 1968, A formal basis for the heuristic determination of minimum cost paths, *IEEE transactions on Systems Science and Cybernetics*, 4 (2), 100-107.

He, K., Zhang, X., Ren, S. ve Sun, J., 2016, Deep residual learning for image recognition, *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770-778.

He, M., Zhu, C., Huang, Q., Ren, B. ve Liu, J., 2020, A review of monocular visual odometry, *The Visual Computer*, 36 (5), 1053-1065.

Heo, S., Cha, J. ve Park, C. G., 2019, EKF-Based Visual Inertial Navigation Using Sliding Window Nonlinear Optimization, *IEEE Transactions on Intelligent Transportation Systems*, 20 (7), 2470-2479.

Hochreiter, S. ve Schmidhuber, J., 1997, Long Short-Term Memory, *Neural Computation*, 9 (8), 1735-1780.

Hong, E. ve Lim, J., 2018, Visual-Inertial Odometry with Robust Initialization and Online Scale Estimation, *Sensors*, 18 (12), 4287.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M. ve Adam, H., 2017, Mobilenets: Efficient convolutional neural networks for mobile vision applications, *arXiv preprint arXiv:1704.04861*.

Huang, B., Zhao, J. ve Liu, J., 2019, A Survey of Simultaneous Localization and Mapping, *arXiv preprint arXiv:1909.05214*.

Hui, T.-W., Tang, X. ve Loy, C. C., 2018, Liteflownet: A lightweight convolutional neural network for optical flow estimation, *Proceedings of the IEEE conference on computer vision and pattern recognition*, 8981-8989.

Hutter, M., Gehring, C., Lauber, A., Gunther, F., Bellicoso, C. D., Tsounis, V., Fankhauser, P., Diethelm, R., Bachmann, S. ve Blösch, M., 2017, ANYmal-toward legged robots for harsh environments, *Advanced Robotics*, 31 (17), 918-931.

Jang, D., Yoo, J., Son, C. Y., Kim, D. ve Kim, H. J., 2020, Multi-Robot Active Sensing and Environmental Model Learning With Distributed Gaussian Process, *IEEE Robotics and Automation Letters*, 5 (4), 5905-5912.

Jeong, I.-B., Lee, S.-J. ve Kim, J.-H., 2019, Quick-RRT*: Triangular inequality-based implementation of RRT* with improved initial solution and convergence rate, *Expert Systems with Applications*, 123, 82-90.

Jiang, G., Yin, L., Jin, S., Tian, C., Ma, X. ve Ou, Y., 2019, A simultaneous localization and mapping (SLAM) framework for 2.5 D map building based on low-cost LiDAR and vision fusion, *Applied Sciences*, 9 (10), 2105.

Jiang, J., Yuan, J., Zhang, X. ve Zhang, X., 2021a, DVIO: An Optimization-Based Tightly Coupled Direct Visual-Inertial Odometry, *IEEE Transactions on Industrial Electronics*, 68 (11), 11212-11222.

Jiang, P., Cheng, Y., Yi, J. ve Liu, J., 2021b, An efficient constrained global optimization algorithm with a clustering-assisted multiobjective infill criterion using Gaussian process regression for expensive problems, *Information Sciences*, 569, 728-745.

Jiao, J., Jiao, J., Mo, Y., Liu, W. ve Deng, Z., 2018, Magicvo: End-to-end monocular visual odometry through deep bi-directional recurrent convolutional neural network, *arXiv preprint arXiv:1811.10964*.

Jiao, Y., Shi, G. ve Tran, T. D., 2021, Optical Flow Estimation via Motion Feature Recovery, *arXiv preprint arXiv:2101.06333*.

Kaess, M., Williams, S., Indelman, V., Roberts, R., Leonard, J. J. ve Dellaert, F., 2012, Concurrent filtering and smoothing, *2012 15th International Conference on Information Fusion*, 1300-1307.

Kaiser, J., Martinelli, A., Fontana, F. ve Scaramuzza, D., 2016, Simultaneous state initialization and gyroscope bias calibration in visual inertial aided navigation, *IEEE Robotics and Automation Letters*, 2 (1), 18-25.

Kang, H. ve Park, F., 2015, Motion optimization using Gaussian process dynamical models, *Multibody System Dynamics*, 34 (4), 307-325.

Karaim, M., Noureldin, A. ve Karamat, T. B., 2019, Low-cost IMU Data Denoising using Savitzky-Golay Filters, *2019 International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, 1-5.

Karaman, S. ve Frazzoli, E., 2011, Sampling-based algorithms for optimal motion planning, *The international journal of Robotics Research*, 30 (7), 846-894.

Karur, K., Sharma, N., Dharmatti, C. ve Siegel, J. E., 2021, A Survey of Path Planning Algorithms for Mobile Robots, *Vehicles*, 3 (3), 448-468.

Kavraki, L. E., Svestka, P., Latombe, J.-C. ve Overmars, M. H., 1996, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, *IEEE Transactions on robotics and automation*, 12 (4), 566-580.

Kennedy, J. ve Eberhart, R., 1995, Particle swarm optimization, *Proceedings of ICNN'95-international conference on neural networks*, 1942-1948.

Khaddour, M., Shidlovskiy, S., Shashev, D. ve Mondal, M., 2021, Survey of Denoising Methods for Inertial Sensor Measurements, *2021 International Conference on Information Technology (ICIT)*, 787-790.

Khatib, O., 1986, Real-time obstacle avoidance for manipulators and mobile robots, In: Autonomous robot vehicles, Eds: Springer, p. 396-404.

Kiani, F., Seyyedabbasi, A., Aliyev, R., Gulle, M. U., Basyildiz, H. ve Shah, M. A., 2021, Adapted-RRT: novel hybrid method to solve three-dimensional path planning problem using sampling and metaheuristic-based algorithms, *Neural Computing and Applications*, 1-31.

Kim, D., Lee, J. ve Yoon, S.-e., 2014, Cloud RRT∗: Sampling cloud based RRT∗, *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2519-2526.

Kim, J.-E., Nam, N.-E., Shim, J.-S., Jung, Y.-H., Cho, B.-H. ve Hwang, J. J., 2020, Transfer Learning via Deep Neural Networks for Implant Fixture System Classification Using Periapical Radiographs, *Journal of Clinical Medicine*, 9 (4), 1117.

Kim, Y.-G., Kim, H.-K., Lee, S.-G. ve Lee, K.-D., 2006, Ubiquitous home security robot based on sensor network, *2006 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, 700-704.

Kim, Y., Jung, H., Min, D. ve Sohn, K., 2018, Deep monocular depth estimation via integration of global and local predictions, *IEEE transactions on Image Processing*, 27 (8), 4131-4144.

Kingma, D. P. ve Ba, J., 2014, Adam: A method for stochastic optimization, *arXiv preprint arXiv:1412.6980*.

KIZRAK, M. A. ve BOLAT, B., 2018, Derin öğrenme ile kalabalık analizi üzerine detaylı bir araştırma, *Bilişim Teknolojileri Dergisi*, 11 (3), 263-286.

Klein, G. ve Murray, D., 2007, Parallel Tracking and Mapping for Small AR Workspaces, *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, 225-234.

Koenig, N. ve Howard, A., 2004, Design and use paradigms for gazebo, an open-source multi-robot simulator, *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, 2149-2154.

Krizhevsky, A., Sutskever, I. ve Hinton, G. E., 2012, Imagenet classification with deep convolutional neural networks, *Advances in neural information processing systems*, 25, 1097-1105.

Kumar, A., Sarkar, S. ve Pradhan, C., 2020, Malaria Disease Detection Using CNN Technique with SGD, RMSprop and ADAM Optimizers, In: Deep Learning Techniques for Biomedical and Health Informatics, Eds: Dash, S., Acharya, B. R., Mittal, M., Abraham, A. ve Kelemen, A., *Cham*: Springer International Publishing, p. 211-230.

Kurt, F., 2018, Evrişimli Sinir Ağlarında Hiper Parametrelerin Etkisinin İncelenmesi, *Hacettepe Üniversitesi*.

Lakshmanan, A. K., Mohan, R. E., Ramalingam, B., Le, A. V., Veerajagadeshwar, P., Tiwari, K. ve Ilyas, M., 2020, Complete coverage path planning using reinforcement learning for tetromino based cleaning and maintenance robot, *Automation in Construction*, 112, 103078.

LaValle, S. M., 1998, Rapidly-exploring random trees: A new tool for path planning.

Le Gentil, C., Vidal-Calleja, T. ve Huang, S., 2020, Gaussian process preintegration for inertial-aided state estimation, *IEEE Robotics and Automation Letters*, 5 (2), 2108-2114.

LeNail, A., 2019, NN-SVG: Publication-Ready Neural Network Architecture Schematics, *J. Open Source Softw.*, 4 (33), 747.

Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R. ve Furgale, P., 2015, Keyframe-based visual–inertial odometry using nonlinear optimization, *The international journal of Robotics Research*, 34 (3), 314-334.

Li, A., Ruan, X., Huang, J., Zhu, X. ve Wang, F., 2019, Review of vision-based Simultaneous Localization and Mapping, *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, 117-123.

Li, C., 2020, Towards End-to-End Learning of Monocular Visual-Inertial Odometry with an Extended Kalman Filter, *University of Toronto (Canada)*.

Li, C. ve Waslander, S. L., 2020, Towards End-to-end Learning of Visual Inertial Odometry with an EKF, *2020 17th Conference on Computer and Robot Vision (CRV)*, 190-197.

Li, R., Wang, S., Long, Z. ve Gu, D., 2018, Undeepvo: Monocular visual odometry through unsupervised deep learning, *2018 IEEE international conference on robotics and automation (ICRA)*, 7286-7291.

Li, Y., Wei, W., Gao, Y., Wang, D. ve Fan, Z., 2020, PQ-RRT*: An improved path planning algorithm for mobile robots, *Expert Systems with Applications*, 152, 113425.

Liao, B., Wan, F., Hua, Y., Ma, R., Zhu, S. ve Qing, X., 2021, F-RRT*: An improved path planning algorithm with improved initial solution and convergence rate, *Expert Systems with Applications*, 184, 115457.

Liu, G. ve Guo, J., 2019, Bidirectional LSTM with attention mechanism and convolutional layer for text classification, *Neurocomputing*, 337, 325-338.

Liu, Y., Dai, H.-N., Wang, Q., Shukla, M. K. ve Imran, M., 2020, Unmanned aerial vehicle for internet of everything: Opportunities and challenges, *Computer communications*, 155, 66-83.

López, E., García, S., Barea, R., Bergasa, L. M., Molinos, E. J., Arroyo, R., Romera, E. ve Pardo, S., 2017, A multi-sensorial simultaneous localization and mapping (SLAM) system for low-cost micro aerial vehicles in GPS-denied environments, *Sensors*, 17 (4), 802.

Lowe, D. G., 2004, Distinctive image features from scale-invariant keypoints, *International journal of computer vision*, 60 (2), 91-110.

Lu, F. ve Milios, E., 1997, Globally consistent range scan alignment for environment mapping, *Autonomous Robots*, 4 (4), 333-349.

Lynen, S., Achtelik, M. W., Weiss, S., Chli, M. ve Siegwart, R., 2013, A robust and modular multi-sensor fusion approach applied to mav navigation, *2013 IEEE/RSJ international conference on intelligent robots and systems*, 3923-3929.

Mahdianpari, M., Salehi, B., Rezaee, M., Mohammadimanesh, F. ve Zhang, Y., 2018, Very Deep Convolutional Neural Networks for Complex Land Cover Mapping Using Multispectral Remote Sensing Imagery, *Remote Sensing*, 10 (7), 1119.

Mansur, S., Habib, M., Pratama, G. N. P., Cahyadi, A. I. ve Ardiyanto, I., 2017, Real time monocular visual odometry using optical flow: study on navigation of quadrotors UAV, *2017 3rd International Conference on Science and Technology-Computer (ICST)*, 122-126.

Mashayekhi, R., Idris, M. Y. I., Anisi, M. H., Ahmedy, I. ve Ali, I., 2020, Informed RRT*-connect: An asymptotically optimal single-query path planning method, *IEEE Access*, 8, 19842-19852.

Maurović, I., Seder, M., Lenac, K. ve Petrović, I., 2017, Path planning for active SLAM based on the D* algorithm with negative edge weights, *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48 (8), 1321-1331.

Mishra, J. P., Singh, K. ve Chaudhary, H., 2021, A hybrid noise removal algorithm for MEMS sensors, *Materials Today: Proceedings*.

Montemerlo, M., Thrun, S., Koller, D. ve Wegbreit, B., 2002, FastSLAM: A factored solution to the simultaneous localization and mapping problem, *Aaai/iaai*, 593598.

Mostafa, M., Zahran, S., Moussa, A., El-Sheimy, N. ve Sesay, A., 2018, Radar and Visual Odometry Integrated System Aided Navigation for UAVS in GNSS Denied Environment, *Sensors*, 18 (9), 2776.

Mourikis, A. I. ve Roumeliotis, S. I., 2007, A multi-state constraint Kalman filter for vision-aided inertial navigation, *Proceedings 2007 IEEE International Conference on Robotics and Automation*, 3565-3572.

Munguia, R., Trujillo, J.-C., Guerra, E. ve Grau, A., 2022, A Hybrid Visual-Based SLAM Architecture: Local Filter-Based SLAM with KeyFrame-Based Global Mapping, *Sensors*, 22 (1), 210.

Munguía, R., Urzua, S., Bolea, Y. ve Grau, A., 2016, Vision-based SLAM system for unmanned aerial vehicles, *Sensors*, 16 (3), 372.

Mur-Artal, R., Montiel, J. M. M. ve Tardós, J. D., 2015, ORB-SLAM: A Versatile and Accurate Monocular SLAM System, *IEEE Transactions on Robotics*, 31 (5), 1147-1163.

Mur-Artal, R. ve Tardós, J. D., 2017a, Visual-inertial monocular SLAM with map reuse, *IEEE Robotics and Automation Letters*, 2 (2), 796-803.

Mur-Artal, R. ve Tardós, J. D., 2017b, Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras, *IEEE Transactions on Robotics*, 33 (5), 1255-1262.

Nasir, J., Islam, F., Malik, U., Ayaz, Y., Hasan, O., Khan, M. ve Muhammad, M. S., 2013, RRT*-SMART: A rapid convergence implementation of RRT, *International Journal of Advanced Robotic Systems*, 10 (7), 299.

Niloy, M. A. K., Shama, A., Chakrabortty, R. K., Ryan, M. J., Badal, F. R., Tasneem, Z., Ahamed, M. H., Moyeen, S. I., Das, S. K., Ali, M. F., Islam, M. R. ve Saha, D. K., 2021, Critical Design and Control Issues of Indoor Autonomous Mobile Robots: A Review, *IEEE Access*, 9, 35338-35370.

Noack, M. M., Doerk, G. S., Li, R., Streit, J. K., Vaia, R. A., Yager, K. G. ve Fukuto, M., 2020, Autonomous materials discovery driven by Gaussian process regression with inhomogeneous measurement noise and anisotropic kernels, *Scientific Reports*, 10 (1), 17663.

Noreen, I., Khan, A. ve Habib, Z., 2016, Optimal path planning using RRT* based approaches: a survey and future directions, *Int. J. Adv. Comput. Sci. Appl*, 7 (11), 97-107.

Noreen, I., Khan, A., Ryu, H., Doh, N. L. ve Habib, Z., 2018, Optimal path planning in cluttered environment using RRT*-AB, *Intelligent Service Robotics*, 11 (1), 41-52.

Olgun, M., Onarcan, A. O., Özkan, K., Işik, Ş., Sezer, O., Özgişi, K., Ayter, N. G., Başçiftçi, Z. B., Ardiç, M. ve Koyuncu, O., 2016, Wheat grain classification by using dense SIFT features with SVM classifier, *Computers and Electronics in Agriculture*, 122, 185-190.

Osco, L. P., Junior, J. M., Ramos, A. P. M., de Castro Jorge, L. A., Fatholahi, S. N., de Andrade Silva, J., Matsubara, E. T., Pistori, H., Gonçalves, W. N. ve Li, J., 2021, A review on deep learning in UAV remote sensing, *International Journal of Applied Earth Observation and Geoinformation*, 102, 102456.

Pal, M. ve Deswal, S., 2010, Modelling pile capacity using Gaussian process regression, *Computers and Geotechnics*, 37 (7), 942-947.

Palunko, I., Cruz, P. ve Fierro, R., 2012, Agile Load Transportation : Safe and Efficient Load Manipulation with Aerial Robots, *IEEE Robotics & Automation Magazine*, 19 (3), 69-79.

Pandey, T., Pena, D., Byrne, J. ve Moloney, D., 2021, Leveraging deep learning for visual odometry using optical flow, *Sensors*, 21 (4), 1313.

Piao, J.-C. ve Kim, S.-D., 2017, Adaptive monocular visual–inertial SLAM for real-time augmented reality applications in mobile devices, *Sensors*, 17 (11), 2567.

Price, K., Storn, R. M. ve Lampinen, J. A., 2006, Differential evolution: a practical approach to global optimization, Springer Science & Business Media, p.

Pumarola, A., Vakhitov, A., Agudo, A., Sanfeliu, A. ve Moreno-Noguer, F., 2017, PL-SLAM: Real-time monocular visual SLAM with points and lines, *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 4503-4508.

Qin, H., Meng, Z., Meng, W., Chen, X., Sun, H., Lin, F. ve Ang, M. H., 2019, Autonomous Exploration and Mapping System Using Heterogeneous UAVs and UGVs in GPS-Denied Environments, *IEEE Transactions on Vehicular Technology*, 68 (2), 1339-1350.

Qin, T., Li, P. ve Shen, S., 2018, VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator, *IEEE Transactions on Robotics*, 34 (4), 1004-1020.

Qu, C., Gai, W., Zhong, M. ve Zhang, J., 2020, A novel reinforcement learning based grey wolf optimizer algorithm for unmanned aerial vehicles (UAVs) path planning, *Applied soft computing*, 89, 106099.

Quan, M., Piao, S., Tan, M. ve Huang, S.-S., 2019a, Accurate monocular visual-inertial SLAM using a map-assisted EKF approach, *IEEE Access*, 7, 34289-34300.

Quan, M., Piao, S., Tan, M. ve Huang, S., 2019b, Accurate Monocular Visual-Inertial SLAM Using a Map-Assisted EKF Approach, *IEEE Access*, 7, 34289-34300.

Qureshi, A. H. ve Ayaz, Y., 2016, Potential functions based sampling heuristic for optimal path planning, *Autonomous Robots*, 40 (6), 1079-1093.

Qureshi, A. H. ve Yip, M. C., 2018, Deeply informed neural sampling for robot motion planning, *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 6582-6588.

Qureshi, A. H., Simeonov, A., Bency, M. J. ve Yip, M. C., 2019, Motion planning networks, *2019 International Conference on Robotics and Automation (ICRA)*, 2118-2124.

Qureshi, A. H., Miao, Y., Simeonov, A. ve Yip, M. C., 2020, Motion planning networks: Bridging the gap between learning-based and classical motion planners, *IEEE Transactions on Robotics*, 37 (1), 48-66.

Qureshi, A. H., Miao, Y., Simeonov, A. ve Yip, M. C., 2021, Motion Planning Networks: Bridging the Gap Between Learning-Based and Classical Motion Planners, *IEEE Transactions on Robotics*, 37 (1), 48-66.

Ren, Z., Wang, L. ve Bi, L., 2019, Robust GICP-Based 3D LiDAR SLAM for Underground Mining Environment, *Sensors*, 19 (13), 2915.

Richter, P. ve Toledano-Ayala, M., 2015, Revisiting Gaussian Process Regression Modeling for Localization in Wireless Sensor Networks, *Sensors*, 15 (9), 22587-22615.

Roberge, V., Tarbouchi, M. ve Labonté, G., 2018, Fast genetic algorithm path planner for fixed-wing military UAV using GPU, *IEEE Transactions on Aerospace and Electronic Systems*, 54 (5), 2105-2117.

ROS, 2020, https://www.ros.org

Rosinol, A., Abate, M., Chang, Y. ve Carlone, L., 2020, Kimera: an open-source library for real-time metric-semantic localization and mapping, *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 1689-1696.

Rossi, G., Tanteri, L., Tofani, V., Vannocci, P., Moretti, S. ve Casagli, N., 2018, Multitemporal UAV surveys for landslide mapping and characterization, *Landslides*, 15 (5), 1045-1052.

Rosten, E. ve Drummond, T., 2006, Machine learning for high-speed corner detection, *European conference on computer vision*, 430-443.

Rubio, F., Valero, F. ve Llopis-Albert, C., 2019, A review of mobile robots: Concepts, methods, theoretical framework, and applications, *International Journal of Advanced Robotic Systems*, 16 (2), 1729881419839596.

Rublee, E., Rabaud, V., Konolige, K. ve Bradski, G., 2011, ORB: An efficient alternative to SIFT or SURF, *2011 International conference on computer vision*, 2564-2571.

Ruggiero, F., Lippiello, V. ve Ollero, A., 2018, Aerial manipulation: A literature review, *IEEE Robotics and Automation Letters*, 3 (3), 1957-1964.

RYZE, 2020, Tello, https://www.ryzerobotics.com/tello/specs:

Sabanci, K. ve Balci, S., 2020, Development of an expression for the output voltage ripple of the DC-DC boost converter circuits by using particle swarm optimization algorithm, *Measurement*, 158, 107694.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. ve Chen, L.-C., 2018, Mobilenetv2: Inverted residuals and linear bottlenecks, *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4510-4520.

Sansebastiano, E. ve del Pobil, A. P., 2021, The Relationship between "C-Space","Heuristic Methods", and "Sampling Based Planner", In: Motion Planning, Eds: IntechOpen, p.

Saputra, M. R. U., de Gusmao, P. P., Lu, C. X., Almalioglu, Y., Rosa, S., Chen, C., Wahlström, J., Wang, W., Markham, A. ve Trigoni, N., 2020, Deeptio: A deep thermal-inertial odometry with visual hallucination, *IEEE Robotics and Automation Letters*, 5 (2), 1672-1679.

Sasaki, Y. ve Nitta, J., 2017, Long-term demonstration experiment of autonomous mobile robot in a science museum, *2017 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS)*, 304-310.

Savitzky, A. ve Golay, M. J., 1964, Smoothing and differentiation of data by simplified least squares procedures, *Analytical chemistry*, 36 (8), 1627-1639.

Scaramuzza, D. ve Fraundorfer, F., 2011, Visual odometry [tutorial], *IEEE Robotics & Automation Magazine*, 18 (4), 80-92.

Scaramuzza, D. ve Zhang, Z., 2019, Visual-inertial odometry of aerial robots, *arXiv preprint arXiv:1906.03289*.

Schafer, R. W., 2011, What is a Savitzky-Golay filter?[lecture notes], *IEEE Signal processing magazine*, 28 (4), 111-117.

Schulz, E., Speekenbrink, M. ve Krause, A., 2018, A tutorial on Gaussian process regression: Modelling, exploring, and exploiting functions, *Journal of Mathematical Psychology*, 85, 1-16.

Se, S., Lowe, D. ve Little, J., 2002, Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks, *The international journal of Robotics Research*, 21 (8), 735-758.

Servières, M., Renaudin, V., Dupuis, A. ve Antigny, N., 2021, Visual and visual-inertial slam: State of the art, classification, and experimental benchmarking, *Journal of Sensors*, 2021.

Sherstinsky, A., 2020, Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network, *Physica D: Nonlinear Phenomena*, 404, 132306.

Siegwart, R., Nourbakhsh, I. R. ve Scaramuzza, D., 2011, Introduction to autonomous mobile robots, MIT press, p.

Simonyan, K. ve Zisserman, A., 2014, Very deep convolutional networks for large-scale image recognition, *arXiv preprint arXiv:1409.1556*.

Smith, R., Self, M. ve Cheeseman, P., 1990, Estimating uncertain spatial relationships in robotics, In: Autonomous robot vehicles, Eds: Springer, p. 167-193.

Smith, R. C. ve Cheeseman, P., 1986, On the representation and estimation of spatial uncertainty, *The international journal of Robotics Research*, 5 (4), 56-68.

Spero, D. J. ve Jarvis, R. A., 2007, A review of robotic SLAM.

Spurny, V., Pritzl, V., Walter, V., Petrlik, M., Baca, T., Stepan, P., Zaitlik, D. ve Saska, M., 2021, Autonomous firefighting inside buildings by an unmanned aerial vehicle, *IEEE Access*, 9, 15872-15890.

Stachniss, C., Leonard, J. J. ve Thrun, S., 2016, Simultaneous localization and mapping, In: Springer Handbook of Robotics, Eds: Springer, p. 1153-1176.

Stentz, A., 1997, Optimal and efficient path planning for partially known environments, In: Intelligent unmanned ground vehicles, Eds: Springer, p. 203-220.

Strasdat, H., Montiel, J. ve Davison, A. J., 2010, Real-time monocular SLAM: Why filter?, *2010 IEEE International Conference on Robotics and Automation*, 2657-2664.

Strasdat, H., Montiel, J. M. ve Davison, A. J., 2012, Visual SLAM: why filter?, *Image and Vision Computing*, 30 (2), 65-77.

Stumberg, L. V., Usenko, V. ve Cremers, D., 2018, Direct Sparse Visual-Inertial Odometry Using Dynamic Marginalization, *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2510-2517.

Sturm, J., Engelhard, N., Endres, F., Burgard, W. ve Cremers, D., 2012, A benchmark for the evaluation of RGB-D SLAM systems, *2012 IEEE/RSJ international conference on intelligent robots and systems*, 573-580.

Sun, D., Yang, X., Liu, M.-Y. ve Kautz, J., 2018, Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume, *Proceedings of the IEEE conference on computer vision and pattern recognition*, 8934-8943.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. ve Rabinovich, A., 2015, Going deeper with convolutions, *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1-9.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. ve Wojna, Z., 2016, Rethinking the inception architecture for computer vision, *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2818-2826.

Şırlancı, M., 2021, Malicious code detection: run trace analysis by LSTM, *Middle East Technical University*.

Tai, L. ve Liu, M., 2016, Deep-learning in mobile robotics-from perception to control systems: A survey on why and why not, *arXiv preprint arXiv:1612.07139*.

Tang, J., 2020, Deep Learning Assisted Visual Odometry, Doctoral Thesis, *KTH Royal Institute of Technology*.

Tang, Y., Qin, L., Li, X., Chew, C. ve Zhu, J., 2017, A frog-inspired swimming robot based on dielectric elastomer actuators, *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2403-2408.

Thrun, S., Koller, D., Ghahramani, Z., Durrant-Whyte, H. ve Ng, A. Y., 2004, Simultaneous mapping and localization with sparse extended information filters: Theory and initial results, In: Algorithmic Foundations of Robotics V, Eds: Springer, p. 363-380.

Thrun, S. ve Leonard, J. J., 2008, Simultaneous Localization and Mapping, In: Springer Handbook of Robotics, Eds: Siciliano, B. ve Khatib, O., *Berlin, Heidelberg*: Springer Berlin Heidelberg, p. 871-889.

Tieleman, T. ve Hinton, G., 2012, Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude, *COURSERA: Neural networks for machine learning*, 4 (2), 26-31.

Toraman, S., Alakus, T. B. ve Turkoglu, I., 2020, Convolutional capsnet: A novel artificial neural network approach to detect COVID-19 disease from X-ray images using capsule networks, *Chaos, Solitons & Fractals*, 140, 110122.

Trabes, E. ve Jordan, M. A., 2017, A Node-Based Method for SLAM Navigation in Self-Similar Underwater Environments: A Case Study, *Robotics*, 6 (4), 29.

Triggs, B., McLauchlan, P. F., Hartley, R. I. ve Fitzgibbon, A. W., 1999, Bundle adjustment—a modern synthesis, *International workshop on vision algorithms*, 298-372.

Trujillo, J.-C., Munguia, R., Guerra, E. ve Grau, A., 2018, Cooperative monocular-based SLAM for multi-UAV systems in GPS-denied environments, *Sensors*, 18 (5), 1351.

Unlersen, M. F., Balci, S., Aslan, M. F. ve Sabanci, K., 2021, The Speed Estimation via BiLSTM-Based Network of a BLDC Motor Drive for Fan Applications, *Arabian Journal for Science and Engineering*.

Varshosaz, M., Afary, A., Mojaradi, B., Saadatseresht, M. ve Ghanbari Parmehr, E., 2020, Spoofing detection of civilian UAVs using visual odometry, *ISPRS International Journal of Geo-Information*, 9 (1), 6.

Wang, J., Chi, W., Li, C., Wang, C. ve Meng, M. Q.-H., 2020a, Neural RRT*: Learning-based optimal path planning, *IEEE Transactions on Automation Science and Engineering*, 17 (4), 1748-1758.

Wang, J., Chi, W., Li, C., Wang, C. ve Meng, M. Q. H., 2020b, Neural RRT*: Learning-Based Optimal Path Planning, *IEEE Transactions on Automation Science and Engineering*, 17 (4), 1748-1758.

Wang, J. ve Gao, F., 2021, Improved visual inertial odometry based on deep learning, *Journal of Physics: Conference Series*, 012016.

Wang, K., Ma, S., Chen, J., Ren, F. ve Lu, J., 2020c, Approaches challenges and applications for deep visual odometry toward to complicated and emerging areas, *IEEE Transactions on Cognitive and Developmental Systems*.

Wang, S., Clark, R., Wen, H. ve Trigoni, N., 2017, Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks, *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2043-2050.

Wang, S., Clark, R., Wen, H. ve Trigoni, N., 2018, End-to-end, sequence-to-sequence probabilistic visual odometry through deep neural networks, *The international journal of Robotics Research*, 37 (4-5), 513-542.

Wang, Z., Zhang, Q., Li, J., Zhang, S. ve Liu, J., 2019, A computationally efficient semantic slam solution for dynamic scenes, *Remote Sensing*, 11 (11), 1363.

Wang, Z., Wu, Z., Si, L., Tong, K. ve Tan, C., 2021, A novel path planning method of mobile robots based on an improved bat algorithm, *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 235 (16), 3071-3086.

Wen, S., Zhao, Y., Yuan, X., Wang, Z., Zhang, D. ve Manfredi, L., 2020, Path planning for active SLAM based on deep reinforcement learning under unknown environments, *Intelligent Service Robotics*, 1-10.

Wen, S., Li, P., Zhao, Y., Zhang, H., Sun, F. ve Wang, Z., 2021, Semantic visual SLAM in dynamic environment, *Autonomous Robots*, 1-12.

Weytjens, H. ve De Weerdt, J., 2020, Process outcome prediction: CNN vs. LSTM (with attention), *International Conference on Business Process Management*, 321-333.

Williams, C. K. ve Rasmussen, C. E., 1996, Gaussian processes for regression.

Williams, C. K. ve Rasmussen, C. E., 2006, Gaussian processes for machine learning, 3, MIT press Cambridge, MA, p.

Williams, H., Browne, W. N. ve Carnegie, D. A., 2017, Learned action slam: Sharing slam through learned path planning information between heterogeneous robotic platforms, *Applied soft computing*, 50, 313-326.

Wu, K., Zhang, T., Su, D., Huang, S. ve Dissanayake, G., 2017, An invariant-EKF VINS algorithm for improving consistency, *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1578-1585.

Xiao, L., Wang, J., Qiu, X., Rong, Z. ve Zou, X., 2019, Dynamic-SLAM: Semantic monocular visual localization and mapping based on deep learning in dynamic environment, *Robotics and Autonomous Systems*, 117, 1-16.

Yamashita, R., Nishio, M., Do, R. K. G. ve Togashi, K., 2018, Convolutional neural networks: an overview and application in radiology, *Insights into imaging*, 9 (4), 611-629.

Yang, L., Qi, J., Song, D., Xiao, J., Han, J. ve Xia, Y., 2016, Survey of robot 3D path planning algorithms, *Journal of Control Science and Engineering*, 2016.

Yang, N., Stumberg, L. v., Wang, R. ve Cremers, D., 2020, D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1281-1292.

Yang, S., Wei, N., Jeon, S., Bencatel, R. ve Girard, A., 2017, Real-time optimal path planning and wind estimation using Gaussian process regression for precision airdrop, *2017 American Control Conference (ACC)*, 2582-2587.

Ye, H., Huang, W., Huang, S., Cui, B., Dong, Y., Guo, A., Ren, Y. ve Jin, Y., 2020, Recognition of banana fusarium wilt based on UAV remote sensing, *Remote Sensing*, 12 (6), 938.

Yousif, K., Bab-Hadiashar, A. ve Hoseinnezhad, R., 2015, An overview to visual odometry and visual SLAM: Applications to mobile robotics, *Intelligent Industrial Systems*, 1 (4), 289-311.

Yurttakal, A. H., 2019, Dinamik kontrastlı manyetik rezonans görüntülerinde bilgisayarlı meme kanseri sınıflandırması, *Kırıkkale Üniversitesi*.

Yusefi, A., Durdu, A., Aslan, M. F. ve Sungur, C., 2021, LSTM and Filter Based Comparison Analysis for Indoor Global Localization in UAVs, *IEEE Access*, 9, 10054-10069.

Zhang, H.-y., Lin, W.-m. ve Chen, A.-x., 2018a, Path planning for the mobile robot: A review, *Symmetry*, 10 (10), 450.

Zhang, J., Ou, Y., Jiang, G. ve Zhou, Y., 2016, An approach to restaurant service robot SLAM, *2016 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2122-2127.

Zhang, L., Wei, L., Shen, P., Wei, W., Zhu, G. ve Song, J., 2018b, Semantic SLAM based on object detection and improved octomap, *IEEE Access*, 6, 75545-75559.

Zhang, Y. ve Wang, S., 2021, LSPP: A Novel Path Planning Algorithm Based on Perceiving Line Segment Feature, *IEEE Sensors Journal*.

Zhang, Z., Xu, C., Yang, J., Tai, Y. ve Chen, L., 2018c, Deep hierarchical guidance and regularization learning for end-to-end depth estimation, *Pattern Recognition*, 83, 430-442.

Zhao, B., Huang, Y., Wei, H. ve Hu, X., 2021, Ego-Motion Estimation Using Recurrent Convolutional Neural Networks through Optical Flow Learning, *Electronics*, 10 (3), 222.

Zhao, Y., Zheng, Z. ve Liu, Y., 2018, Survey on computational-intelligence-based UAV path planning, *Knowledge-Based Systems*, 158, 54-64.

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H. ve He, Q., 2020, A comprehensive survey on transfer learning, *Proceedings of the IEEE*, 109 (1), 43-76.