KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ
ANABİLİM DALI

DOKTORA TEZİ

İNSANSIZ HAVA ARAÇLARI İÇİN RRT VE YPA TABANLI HİBRİD VE YAPAY SİNİR AĞI DESTEKLİ GENEL YOL PLANLAMASININ GELİŞTİRİLMESİ

AYHAN GÜLTEKİN

KOCAELİ 2024

KOCAELİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ

ANABİLİM DALI

DOKTORA TEZİ

İNSANSIZ HAVA ARAÇLARI İÇİN RRT VE YPA TABANLI HİBRİD VE YAPAY SİNİR AĞI DESTEKLİ GENEL YOL PLANLAMASININ GELİŞTİRİLMESİ

AYHAN GÜLTEKİN

Prof.Dr. Yaşar BECERİKLİ		
Danışman,	**Kocaeli Üniversitesi**	**.....**
Prof. Dr. Cemil ÖZ		
Jüri Üyesi,	**Sakarya Üniversitesi**	**.....**
Prof. Dr. Üyesi Temel KAYIKÇIOĞLU		
Jüri Üyesi,	**Karadeniz Teknik Üniversitesi**	**.....**
Doç. Dr. Orhan AKBULUT		
Jüri Üyesi,	**Kocaeli Üniversitesi**	**.....**
Doç. Dr. Alev MUTLU		
Jüri Üyesi,	**Kocaeli Üniversitesi**	**.....**

Tezin Savunulduğu Tarih: 05.02.2024

ETİK BEYAN VE ARAŞTIRMA FONU DESTEĞİ

Kocaeli Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırladığım bu tez/proje çalışmasında,

- Bu tezin/projenin bana ait, özgün bir çalışma olduğunu,
- Çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalarında bilimsel etik ilke ve kurallara uygun davrandığımı,
- Bu çalışma kapsamında elde edilen tüm veri ve bilgiler için kaynak gösterdiğimi ve bu kaynaklara kaynakçada yer verdiğimi,
- Bu çalışmanın Kocaeli Üniversitesi'nin abone olduğu intihal yazılım programı kullanılarak Fen Bilimleri Enstitüsü'nün belirlemiş olduğu ölçütlere uygun olduğunu,
- Kullanılan verilerde herhangi bir tahrifat yapmadığımı,
- Tezin/Projenin herhangi bir bölümünü bu üniversite veya başka bir üniversitede başka bir tez/proje çalışması olarak sunmadığımı,

beyan ederim.

[X] Bu tez/proje çalışmasının herhangi bir aşaması hiçbir kurum/kuruluş tarafından maddi/alt yapı desteği ile desteklenmemiştir.

[ ] Bu tez/proje çalışması kapsamında üretilen veri ve bilgiler ..... tarafından ..... no'lu proje kapsamında maddi/alt yapı desteği alınarak gerçekleştirilmiştir.

Herhangi bir zamanda, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun saptanması durumunda, ortaya çıkacak tüm ahlaki ve hukuki sonuçları kabul ettiğimi bildiririm.

Ayhan GÜLTEKİN

YAYIMLAMA VE FİKRİ MÜLKİYET HAKLARI

Fen Bilimleri Enstitüsü tarafından onaylanan lisansüstü tezimin/projemin tamamını veya herhangi bir kısmını, basılı ve elektronik formatta arşivleme ve aşağıda belirtilen koşullarla kullanıma açma izninin Kocaeli Üniversitesi'ne verdiğimi beyan ederim. Bu izinle Üniversiteye verilen kullanım hakları dışındaki tüm fikri mülkiyet haklarım bende kalacak, tezimin/projemin tamamının ya da bir bölümünün gelecekteki çalışmalarda (makale, kitap, lisans ve patent vb.) kullanımı bana ait olacaktır.
Tezin/projenin kendi özgün çalışmam olduğunu, başkalarının haklarını ihlal etmediğimi ve tezimin/projenin tek yetkili sahibi olduğumu beyan ve taahhüt ederim. Tezimde yer alan telif hakkı bulunan ve sahiplerinden yazılı izin alınarak kullanılması zorunlu metinlerin yazılı izin alarak kullandığımı ve istenildiğinde suretlerini Üniversiteye teslim etmeyi taahhüt ederim.
Yükseköğretim kurulu tarafından yayınlanan ***"Lisansüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge"*** kapsamında tezim aşağıda belirtilen koşullar haricinde YÖK Ulusal Tez Merkezi/ Kocaeli Üniversitesi Kütüphaneleri Açık Erişim Sisteminde erişime açılır.

☐ Enstitü yönetim kurulu kararı ile tezimin/projemin erişime açılması mezuniyet tarihinden itibaren 2 yıl ertelenmiştir.
☐ Enstitü yönetim kurulu gerekçeli kararı ile tezimin/projemin erişime açılması mezuniyet tarihinden itibaren 6 ay ertelenmiştir.
☒ Tezim/projem ile ilgili gizlilik kararı verilmemiştir.

Ayhan GÜLTEKİN

ÖNSÖZ VE TEŞEKKÜR

Bu tezin ortaya çıkmasında beni her zaman yönlendiren ve cesaretlendiren gerek akademik birikimi ve tecrübesi, gerekse de pozitif kişiliği ile her zaman bana gerekli desteği veren saygıdeğer danışman hocam Prof. Dr. Yaşar BECERİKLİ 'ye teşekkürü bir borç bilirim.

Tez izleme jürimde yer alan değerli hocalarım Prof. Dr. Cemil ÖZ ve Doç. Dr. Orhan AKBULUT'un çalışma kapsamındaki değerli öneri ve görüşleri her zaman çok yararlı olmuştur, bunun için kendilerine saygılarımı ve teşekkürlerimi sunarım.

Ayrıca, tez çalışmam boyunca destekleri sebebiyle değerli arkadaşlarım Dr. Samet DİRİ, Dr. Selçuk ÖĞÜTÇÜ, Yük. Müh. Hamdi YILMAZ, Yük. Müh. Faruk ALTUNTAŞ ve destekleri için Kocaeli Üniversitesi Bilgisayar Araştırma ve Uygulama Merkezi ve Öğrenci İşleri Daire Başkanlığı'ndaki çalışma arkadaşlarıma teşekkür ederim.

Onlara ayırmam gereken zamanı tez çalışmam için ayırmamı büyük bir sabır ile karşılayan ve beni her zaman teşvik eden sevgili eşim Neslihan GÜLTEKİN, çocuklarım Ayşe Duru GÜLTEKİN ve Elif Defne GÜLTEKİN'e teşekkür ederim. Onlar olmadan bu tez asla bitmezdi.

Bunun yanında, annem Zehra GÜLTEKİN'e, babam Yusuf GÜLTEKİN'e, kardeşlerime ve birbirinden değerli çalışma arkadaşlarıma beni her zaman motive ettikleri ve bana bunun uzun, zorlu ama sonunda mutlulukla bitecek bir yolculuk olduğunu hatırlattıkları için teşekkür ederim.

Mart – 2024 Ayhan GÜLTEKİN

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

Φ : Yol planlama
Σ : Seri toplamı
δ : [0, T] → R^3 için sınırlandırılmış fonksiyon tanımı
τ : Sınırlandırılmış δ fonksiyonunda Φ için her bir nokta
φ : Roll(Yana Yatma) Euler açısı
θ : Pitch (Burun aşağı/yukarı hareket) Euler açısı
ψ : Yaw(Yön Değiştirme) Euler açısı

Kısaltmalar

CAN : Controller Area Network (Kontrol Alan Ağı)
CNN : Convolutional Neural Network (Evrişimsel Sinir Ağı)
DNN : Deep Neural Network (Derin Sinir Ağı)
FFNN : Feed-Forward Neural Network (İleri Beslemeli Sinir Ağı)
GA : Genetik Algoritma
GWO : Grey Wolf Optimization (Gri Kurt Optimizasyonu)
İHA : İnsansız Hava Aracı
LİDAR : Light Detection and Ranging (Işıkla algılama ve mesafe ölçme)
LMSE : Logarithmic Mean Squared Error (Logaritmik Ortalama Kare Hatası)
LSTM : Long Short-Term Memory (Uzun-Kısa Süreli Bellek)
MSE : Mean Squared Error (Ortalama Kare Hatası)
P-RRT : Probabilistic Rapidly Exploring Random Tree(Olasılıksal Hızlı Keşif Yapılacak Rasgele Ağaç)
PSO : Particle Swarm Optimization" (Parçacık Sürü Optimizasyonu)
ReLU : Rectified Linear Unit (Doğrultulmuş Doğrusal Birim)
RNN : Recurrent Neural Network (Yinelemeli Sinir Ağı)
RRT : Rapidly Exploring Random Tree (Hızlı Keşif Yapılacak Rasgele Ağaç)
RRT* : Rapidly Exploring Random Tree Star (Hızlı Keşif Yapılacak Rasgele Ağaç Yıldız)
ROS : Robot Operating System (Robot İşletim Sistemi)
T : Trajectory (Gezinge)
TF : TensorFlow
TFLite : TensorFlow Lite
TPU : Tensor Processing Unit (Tensör İşlem Birimi)
WSO : White Sharks Optimization (Beyaz Köpekbalığı Optimizasyonu)
YPA : Yapay Potansiyel Alan
YSA : Yapay Sinir Ağı

İNSANSIZ HAVA ARAÇLARI İÇİN RRT VE YPA TABANLI HİBRİD VE YAPAY SİNİR AĞI DESTEKLİ GENEL YOL PLANLAMASININ GELİŞTİRİLMESİ

ÖZET

İnsansız Hava Araçlarının (İHA) kullanım alanları ve otonom olarak yapabildiği görevlerin sayısı günümüzde oldukça artmıştır. İHA gözetleme, keşif vb. askeri görevlerde çok uzun süredir kullanılmaktadır. Günümüzde İHA, önem ve etkisinin yüksek olmasının fark edilmesiyle askeri görevlerle birlikte farklı sivil alanlarda içinde yoğun olarak kullanılmaya başlanmıştır. İHA'nda çözülmesi gereken farklı problemler arasında en önemli problemlerinden birisi de yol planlama problemidir.

Başlangıç ve hedef nokta arasında uçuş sırasında statik ve dinamik engellerden sakınarak güvenli ve araca ait kinematik kısıtlarının da göz önünde bulundurularak uygun bir yolun bulunması yol planlama yaklaşımın temelini oluşturur. Uygun yol planlamasının yapılması, özellikle karmaşık uçuş alanlarında ve gerçek zamanlı hareket planlaması yapılmasına ihtiyaç duyulduğu durumlarda oldukça kritik bir öneme sahiptir. Yol planlaması probleminin çözümü için problemin doğasına uygun olarak farklı yaklaşım ve algoritmalar geliştirilmiştir.

Tez çalışması kapsamında dört motorlu insansız hava aracı için yol planlamasının yapılması amacıyla MATLAB ortamında Rapidly-exploring Random Tree (RRT) ve Yapay Potansiyel Alan (YPA) algoritmaları ile hibrid bir uygulama geliştirilmiştir. Geliştirilen bu uygulamadan elde edilen sonuçlar ile bir Yapay Sinir Ağı (YSA) eğitimi gerçekleştirilmiştir. Bunun için öncelikle geliştirilen hibrid uygulama üzerinden yapay sinir ağının eğitimi için kullanılacak veri kümesi oluşturulmuş sonrasında bu veri kümesi Tensorflow Kütüphanesi kullanılarak oluşturulan YSA modelinin eğitimi için kullanılmıştır. YSA modelinin gerçek dünya problemlerine uygun olarak uygulama ve testlerinin yapılabilmesi amacıyla farklı uygulama dilleri ve platformlarını birlikte çalışabilirliğini sağlayan robot sistemlerde yoğun olarak kullanılan Robot İşletim Sistemi (Robot Operating System - ROS) kullanılmıştır. ROS ile entegrasyonu kolay olan, birlikte çalışabilme başarım düzeyi oldukça yüksek olan bir Gazebo simülatör aracı tez çalışması kapsamında simülasyon platformu olarak kullanılmıştır.

Geliştirilen algoritmanın performansı ROS ortamında RRT ve YPA hibrid algoritması ile karşılaştırılmıştır. Karşılaştırma sonucunda geliştirilen ve önerilen yöntemin yol uzunluğunun ve çalışma süresinin azaltılması açısından başarılı olduğu görülmüştür.

Anahtar Kelimeler: Hibrid Algoritma, İnsansız Hava Aracı, ROS , Yapay Sinir Ağı, Yol Planlama.

DEVELOPMENT OF GLOBAL PATH PLANNING USING HYBRID AND ARTIFICIAL NEURAL NETWORK BASED ON RRT AND APF FOR UNMANED AERIAL VEHICLES

ABSTRACT

The application fields and the number of autonomous tasks that Unmanned Aerial Vehicles (UAVs) can perform have significantly increased in recent times. UAVs have long been utilized for military tasks such as surveillance and reconnaissance. Nowadays, recognizing the importance and high impact of UAVs, their use has commenced intensively in various civilian fields alongside military missions. Among the different challenges that need to be addressed in UAVs, one of the most critical is the path planning problem.

The basis of the path planning approach is to find an appropriate path that avoids static and dynamic obstacles during flight between the start and destination points, taking into account the kinematic constraints of the vehicle. Effective path planning, especially in complex flight areas and situations where real-time motion planning is necessary, holds critical significance. To solve the path planning problem, various approaches and algorithms have been developed that suit the nature of the problem.

Within the scope of the thesis work, a hybrid application utilizing Rapidly-exploring Random Tree (RRT) and Artificial Potential Field (APF) algorithms has been developed in MATLAB environment for path planning of a quadrotor UAV. The results obtained from this application have been used to train an Artificial Neural Network (ANN). For this purpose, a dataset for training the artificial neural network was initially created using the developed hybrid application, and then this dataset was employed to train the ANN model created with the Tensorflow Library. To enable the application and testing of the ANN model in real-world problems, the Robot Operating System (ROS), widely used in robotic systems for its interoperability across different programming languages and platforms, was utilized. Additionally, a Gazebo simulator, known for its high level of interoperability and integration ease with ROS, was employed as the simulation platform within the thesis work.

The performance of the developed algorithm was compared with the RRT and APF hybrid algorithm in the ROS environment. The comparison revealed that the developed and proposed method was successful in reducing the path length and operation time.

Keywords: Hybrid Algorithm, Unmanned Aerial Vehicles, ROS, Artificial Neural Network, Path Planning.

1.GİRİŞ

Günümüzde, İnsansız Hava Araçlarının (İHA) kullanımı her geçen gün artmaktadır. Bu alandaki araştırmalar genellikle İHA'ların navigasyonu, otonom uçuş kabiliyeti, güvenlik önlemleri, enerji verimliliği ve veri işleme gibi konulara odaklanır. Özellikle, yapay zekâ ve makine öğrenimi gibi ileri teknolojiler, İHA'ların verimliliğini, güvenliğini ve uygulanabilirliğini artırmak için giderek daha fazla kullanılmaktadır.

Arama/kurtarma, gözlem, ortamın haritasının çıkarılması gibi sivil ve askeri görevlerin yerine getirilmesinde sırasında insanın biyolojik ve fiziksel sınırları sebebiyle oluşabilecek kısıtların kaldırılmasında İHA kullanımı önemli bir yer tutmaktadır. İHA'larda çözülmesi gereken önemli problemlerden birisi de uygulanabilir ve etkili bir yol planlaması yapabilmektir.

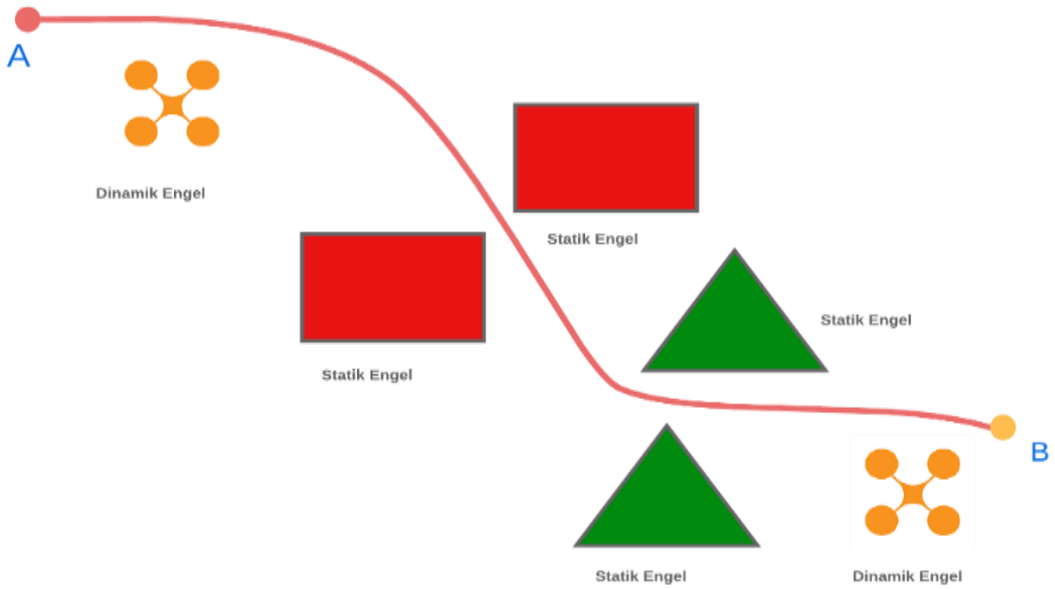
Yol planlama problemi, başlangıç noktasından hedef noktasına belirli amaç fonksiyonlar göz önünde bulundurularak engellerden arındırılmış, araç kinematiklerine en uygun yolun sağlanmasıdır (Anderson, Beard, & McLain, 2005). Yol planlama problemlerinde en önemli sorun, uçuş ortamındaki tepe, doğal vb. engellerin platform üzerindeki sensörler vasıtasıyla tespiti, engellerden arındırılmış yeni bir yol planı yapabilmek, platformların birbiriyle olan olası çarpışmalarını engellemek ve uçuş ortamı içerisinde istenilen hedef rotasından sapmadan gerçek zamanlı olarak yol planlaması yapabilmektir.

İHA'ların 3 boyutlu uzay içerisinde dinamik olarak hem kendi konumlarını belirlemesi hem de başlangıç konumundan bitiş konumuna kadar optimum yol planlamasını yapabilmesi gereklidir (Li ve diğ., 2023). Hedef konum için belirlenen yörünge üzerinde sürekli değişen ve önceden durumları kestirilemeyen engeller için yeni yol planlamasının hızlı ve hatasız olarak yeniden hesaplanması önemli bir zorunluluktur (Yang ve diğ., 2023). Önceden ortamın harita bilgisine sahip olunduğu durumlarda otonom platformun hedef noktaya erişmek için gerekli hesaplamalar başlangıçta yapılabilir. İHA'ların görev yaptığı dış dünya yapısı gereği sürekli değişkenlik gösteren yeni engellerin oluşabileceği bir ortamdır. Bu duruma uyum sağlayabilmek için İHA'lar için gerekli yol planlaması ve modellemelerinin algılayıcılar kullanarak yapılması gerekmektedir. İHA'larda başta kameralar, IMU (Jireskop, İvmeölçer, manyetik alan ölçerden oluşan elektronik ünite) gibi farklı görevleri olan algılayıcılar kullanılabilmektedir. IMU platformun dönme

açısını, açısal hız ve doğrusal ivmeyi hesaplayarak platformun uzaydaki yönelimini bulmaya imkân verir. Çarpışma önleyici ve engel tanıma özelliklerinin oluşturulması İHA optimum yol planlamasının başarımı için hayati öneme sahiptir. Yörünge planının dinamik olması özellikle iteratif olarak çalışan bazı metotlara fazlasıyla işlem yükü getirmektedir. Yol planlama problemlerinin birçoğu zaman ve hesaplama kısıtlarından dolayı NP-hard problemlerdir (Çetin, 2015). İHA'larda yol planlama problemlerinin çözülmesi gezgin satıcı probleminin çözümü ile benzeşmektedir (Zhang ve diğ., 2020). İHA'nın başlangıç noktasından belirlenen bir hedefe ulaşabilmesi için yol oluşturma süreci gezgin satıcı problemine dayanmaktadır. Ancak İHA yol planlama problemleri çok fazla sayıda amaç fonksiyonu barındırması ve yol planlama işleminin üç boyutlu ortamlarda yapılması işlemi çok daha fazla karmaşık hale getirmektedir.

1.1.Problemin Tanımı

Yol planlama probleminin çözümü için geliştirilen algoritma ve yöntemlerin temel amacı İHA'nın bir konumdan diğerine olası kısıtları göz önünde bulundurarak hareket etmesi ile başlangıç noktasından hedef noktaya en uygun yolun bulunmasıdır. En uygun yol bulma sürecinde genel olarak hareketli ve sabit engellerden sakınmak, aracın kinematik ve yapısal özelliklerinin dikkat alınması önemlidir. Şekil 1.1'de temel olarak iki nokta arasında yol belirleme süreci gösterilmektedir.



Şekil 1.1. İki nokta arasında temel yol planlama

Genel olarak İHA'nın belirlenen görevi tamamlaması için uçuş ortamında bulunan dinamik ve statik engellerden sakınması beklenir. Statik engeller dağ, tepe, bina vb. olabilir. Dinamik engeller ise İHA hareketi sırasında karşılaşılabilecek diğer uçuş platformlarını ifade etmektedir. İHA üzerindeki sensörler vasıtasıyla güvenli bir mesafeden engelleri tespit etmeli ve yol planlamasını bu duruma uygun yeniden planlayabilmelidir.

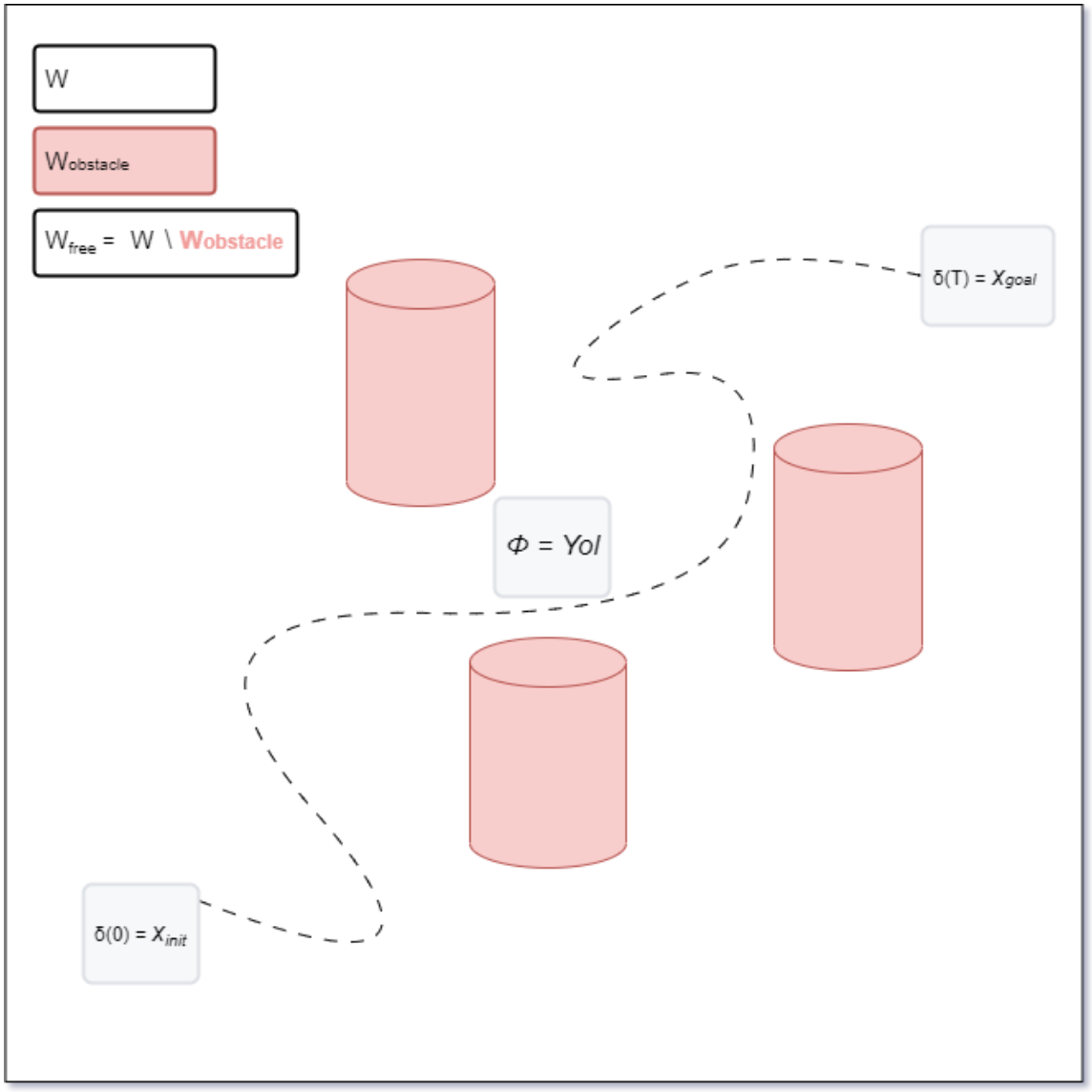
İHA'nın uçuş yaptığı bölge çalışma uzayı olarak adlandırılır ve *w* ile gösterilir. Çalışma uzayı doğası gereği statik ve dinamik engellere sahip olabilir. Çalışma uzayında bulunan engeller w_o ile gösterilmek üzere İHA'nın uçuş bölgesinde serbest olarak hareket edebileceği alan w_{free} ile gösterilir ve Denklem (1.1)'de gösterilen eşitlik ile hesaplanmaktadır.

$$w_{free} = w \setminus \sum_{i}^{n} w_{oi} \tag{1.1}$$

x_{init} ve x_{goal} çalışma uzayı w_{free} alanında başlangıç noktası ve hedef noktası olmak üzere yol planlama problemi $(x_{init}, x_{goal}, w_{free})$ olarak üçlü bir bağlantı olarak tanımlanır. Burada kullanılan " \" sembolü İHA'nın uçuş yaptığı çalışma uzayında bulunan engellerin olmadığı bölgenin elde edilmesi için kullanılmıştır.

δ: $[0, T] \rightarrow R^3$ sınırlanmış bir alan içerisinde verilen bir fonksiyon olarak tanımlanmıştır. Kullanılan bu fonksiyon belirli bir zaman aralığı boyunca oluşturulacak yolu temsil eder. $\delta(0) = x_{init}$, $\delta(T) = x_{goal}$ ve τ, başlangıç ve hedef noktaları arasında bir nokta olmak üzere olmak üzere eğer $\delta(\tau) \in w_{free}$ şartını sağlayan ve tüm τ'ler için Φ süreci mevcut ise $\tau \in [0, T]$ için Φ Yol Planlama olarak tanımlanır (Yang ve diğ., 2014).

Şekil 1.2 'de iki nokta arasında yol belirleme süreci, çalışma uzayı, engeller ve çalışma uzayı içerisinde hareket edilebilecek serbest alanları içerecek şekilde gösterilmiştir. Yol eğrisi, engellerin konumuna ve farklı kısıtlara uygun olarak oluşmaktadır. Araç harekete başlangıç noktasından başladıktan sonra bir sonraki hareket hangi noktaya planlar ve hareket edilecek bir sonraki nokta seçiminde engelle karşılaşma kontrolü yapılır eğer seçilecek noktada engelle karşılaşma durumu oluşursa yeniden yeni bir hareket noktası seçim işlemi yapılır. Bu süreç aracın hedef noktaya ulaşmasına kadar iteratif olarak devam eder.



Şekil 1.2. Uçuş alanı temel gösterimi

Gezinge ve yol planlama tanımları birbiri ile ilişikli ancak içerik olarak farklı kavramlardır. x_{init} *ile* x_{goal} arasında ve w_{free} bölgesinde olmak şartı ile Φ süreci kesikli çizgi veya sürekli bir eğri olabilir. Yol planlamada kullanılan her bir koordinat noktasının t zaman değişkeni ile parametize edilmesi ile yol artık gezinge (T, trajectory) olarak adlandırılır. Gezinge, zamana bağlı olarak birinci ve ikinci türevlerin hız ve ivmenin hesaplanması için kullanılması ile matematiksel olarak ifade edilebilir.

İHA yol planlama problemi, matematiksel olarakda ifade edilebilen ve amacı önceden belirlenmiş, başlangıç ve hedef noktaları arasında kısıtları göz önünde bulundurarak en uygun yolun bulunmasıdır.

İHA yol planlama temel amaçlardan birisi başlangıç ve hedef koordinat arasındaki yolun uzunluğunun minimize yapılmasıdır. Aşağıda minumum yol uzunluğunun amaç fonksiyonun matematiksel gösterimi sunulmuştur. Çalışma kapsamında bu problemin gerçeklenmesi için yeni bir yöntem önerilmiştir. Çalışma kapsamının amaçlarından biri olan yol mesafesinin minimize edilmesi Denklem (1.2) 'de gösterilen eşitlik ile hesaplanmaktadır.

$$Minimize\ J = \sum_{i=1}^{N-1} distance\left(x_{i,}\ x_{i+1}\ \right) \tag{1.2}$$

J = Minimizasyon maliyeti

N = Durumların veya waypointlerin toplam sayısı

x_i = i anındaki İHA'nın pozisyonu

$distance\left(x_{i,}\ x_{i+1}\ \right)$ = sıralı pozisyonlar arası öklid mesafesi

İHA amaç fonksiyonun belirli kısıtlar altında sağlanabilmesi için literatürde kullanılan birçok yöntem ve algoritma mevcuttur. Çalışma kapsamında bu konuda yeni ve etkili bir yöntem olarak yapay sinir ağı yaklaşım kullanılmıştır.

1.2.Yol Planlama Probleminin Kısıtları

Yol planlama problemi aşağıda belirtilen ve problemin kaynağını oluşturan ana kısıtları içerir.

A. Kinematik kısıtlar
B. Çevre kısıtlar
C. Görev tanımlı kısıtlar

1.2.1.Kinematik Kısıtlar

Aracın non-holonomik yani belirli yönlere hareketini serbestçe yapmasını engelleyen kısıtlardır. Minumum dönme yarıçapı, enerji tüketimi, aracın maksimum hızı gibi kısıtlar konum ve hareketin kontrolünü zorlaştırır. Örnek olarak ortalama yükseltisi İHA'nın

kinematik kısıtları sebebiyle yükselebileceği irtifadan daha yüksek çevre şartlarında İHA'nın görev ifası tam olarak yerine getirilemeyecektir.

1.2.2.Çevresel Kısıtlar

Ortamın topografik kısıtlarıdır, doğal (dağ, tepe, vadi, yağmur, kar vb.) ve hareketli (başka bir araç) engeller çevresel kısıtlara örnektir.

1.2.3.Göreve Özgü Kısıtlar

Maksimum kapsama veya belirli bir alan bölgesinin dolaşılması, başlangıçtan hedefe kadar en kısa yoldan gidilmesi, hava fotoğrafçılığı, havadan arazi kontrolü gibi görev kısıtları bu kapsamda değerlendirilmektedir.

1.3.Çalışmanın Amacı

Tez kapsamında geliştirilen yöntemin temel amacı; literatürde aynı amaç için geliştirilmiş çalışmalara özgün bir alternatif oluşturmaktır. Bu amaçla farklı dünya problemlerinde yoğun olarak kullanılan YSA'ları İHA yol planlama probleminin çözümü için tez çalışması kapsamında kullanılmıştır. Yapılan literatür araştırmasında yol planlama probleminin çözümü için YSA yaklaşımın kullanıldığı farklı çalışmaların mevcut olduğu görülmüştür. Ancak çalışmanın bu türdeki çalışmalara göre özgünlüğü, YSA eğitimi için kullanılan veri kümesinin iki farklı geleneksel yol planlama yönteminin hibrid olarak çalıştırılması sonucunda oluşturulması ve oluşturulan veri kümesinin özellik sayısı ve içerik yönünden güçlü olmasıdır.

Tez çalışmasındaki temel amaçlardan bir diğeride özgün yöntemle oluşturulan veri kümesinin kullanımı ile başarımı yüksek bir YSA modeli elde etmektir. Ayrıca bundan sonra mevcut problemin çözümü için YSA kullanılarak yapılacak çalışmalar için önemli bir kaynak oluşturulmasının sağlanmasıda çalışma kapsamında istenen amaçlardan birisidir. YSA eğitiminde kullanılan modellerin gelişmişlik düzeyinin artmasıyla birlikte daha karmaşık ortamlar içinde yol planlama yaklaşımları geliştirilebilecektir. YSA doğası gereği sürekli olarak öğrenme stratejisini geliştirerek girdi ve çıktılar arasındaki hatayı azaltma yönünde hareket eder. Ayrıca uyarlanabilir yapısı ile farklı ortamlarda kullanılabilir.

1.4.Literatür

Yol planlama problemi yapısı itibariyle NP-hard bir problemdir. NP-hard problemler, belirli bir model altında hesaplanabilir tüm problemleri ifade eden hesaplama sınıf kümesini içerisinde bulunur. Hesaplama sınıf kümesine giren problemler çözülebilir veya hesaplanabilir olarak ifade edilir. Ancak bu kümede bulunan problemlerin hesaplama ve çözüm süreleri problemin zorluk düzeyine göre farklı olabilmektedir. Yol planlama probleminin bu sınıfa girmesindeki en büyük sebeplerden birisi de uçuş ortamının komplekslik düzeyinin ve kısıtların artması ile başlangıç ve hedef noktası arasında optimal yolun bulunması için gerekli sürenin üssel olarak artmasıdır. Problemin çözümü için klasik teknikler, sezgisel, meta-sezgisel ve makina öğrenmesi yaklaşımları sıklıkla kullanılmaktadır.

Geleneksel olarak kullanılan yol planlama algoritmaları, analitik olarak çözümler sunsa da çözüme ulaşma sürecinde özellikle ortamın haritasının önceden bilinmesinin gerekliliği ve konum bilgisinin problemin çözümünde önemli olması gibi nedenlerden özellikle kompleks ve engellerin dinamik olarak değişebildiği ortamlarda performansları yetersiz kalabilmektedir. Son yıllarda YSA'ları, kullanım alanlarının genişlemesi, farklı problemler için kullanılabilme avantajları ve problem çözme başarım oranlarının yüksek olmaları ile İHA yol planlama probleminin çözümü içinde kullanılmaya başlanmışlardır. Öğrenme sürecinin etkin olarak kullanılması, hata oranın öğrenme sürecinde sürekli iyileştirilmesi, YSA'nın temelinde insan beyninin herhangi bir süreci öğrenmesi ile aynı prensiplerin olması ve gerçek dünya problemlerinin doğasında bulunan doğrusal olmayan problemlerin çözümünde de çok rahat kullanılabilmesi gibi avantajları YSA tabanlı yol planlama algoritmalarının kullanımını artırmaktadır.

Yapay Potansiyel Alan (YPA) yaklaşımı ilk olarak Khatib (1986), tarafından ortaya atılmıştır. Khatip sunmuş olduğu çalışmayla; robotun kuvvet alanı etkisiyle hareket ettiğini, ulaşılacak noktanın çekici kuvvet ve engellerin itici kuvvetleri oluşturduğunu ve robotun yönünü ve hızını bütün bu kuvvetlerin oluşturduğu bileşke kuvvete göre belirleyebileceğini açıklamıştır (Khatip, 1986).

Sujit & Beard (2009) sunmuş oldukları çalışmada çoklu İHA yol planlama probleminin çözümü için Parçacık Sürü Optimizasyonu (PSO) algoritması kullanmışlardır. Statik ve

hareketli engellerin olduğu ortamda PSO algoritmasının hesaplama zamanı açısından önemli geliştirmeler sağladığını belirtmişlerdir (Sujit & Beard, 2009).

Karaman & Frazzoli (2011), sunmuş oldukları çalışma ile olasılıksal duruma uygun olarak Rapidly Exploring Random Tree (RRT) algoritmasının optimum yol belirleme yüzdesinin artırımı için Rapidly Exploring Random Tree Star (RRT*) ve Probabilistic Roadmap Star (PRM*) algoritmasını geliştirmişlerdir. Bu algoritma ile olasılıksal sonsuzluk durumunun detaylı analizini yapmışlardır. Ayrıca yol içerisindeki rastgele noktaların sayısı arttıkça çözüme ulaşma maliyetinin asimptotik davranışını incelemişlerdir (Karaman & Frazzoli, 2011).

Qureshi ve diğ., (2014), sunmuş oldukları çalışma ile RRT* algoritmasının yüksek hesaplama maliyetinin engellenmesi ve optimal yola ulaşılması için potansiyel fonksiyon tabanlı P-RRT* (Potential Based-RRT*) yaklaşımını geliştirmişlerdir. Bu yaklaşım YPA yönteminin lokal minumum problemi gibi temel bazı problemlerine çözüm olmuş ve aynı zamanda gerçek zamanlı İHA yol planlaması uygulaması geliştirilebilmesi içinde temel bir kaynak oluşturmuştur (Qureshi ve diğ., 2014).

Özalp ve diğ., (2014) sunmuş oldukları çalışmada, aracın kinematik kısıtlarını, uçuş alanının fiziksel şartlarınıda göz önünde bulundurarak İHA optimum yol planlama probleminin çözümü için genetik algoritmadan (GA) yararlanmışlardır. Uçuş alanı bilgisi için NASA tarafından sağlanan 3D uydu verilerini kullanmışlardır. Aldıkları sonuçlara göre çoklu İHA kullanımının hem zaman hemde performans açısından çok daha iyi sonuç verdiğini belirlemişlerdir, çalışma kapsamında global yol planlama süreci GA tarafından yapılmıştır (Özalp ve diğ., 2014).

Eriksson (2015), sunmuş olduğu çalışmada RRT, RRT* ve RRT-u algoritmalarını ayrı ayrı inceleyip değerlendirmiştir. Çalışmasında RRT-u algoritmasının diğer iki algoritmaya göre dinamik ortamlar için çok daha fazla başarılı sonuçlar verdiğini tespit etmiştir (Erikson, 2015).

Bai ve diğ., (2018), YPA yaklaşımının geliştirilmesi için YPA içerisinde tanımlanan boylamsal faktör kavramını ortaya atmışlardır. Ayrıca bu çalışma ile lokal minumum

problemini çözümü için yeni bir yaklaşım önermişlerdir. Çalışma kapsamında B-spline enterpolasyon metodu ile YPA performansını artırmışlardır (Bai ve diğ., 2018).

Chen ve diğ., (2018), sunmuş oldukları çalışmada yol planlama probleminin çözümü için A* algoritmasını kullanmışlardır. Sunulan çalışmada 2D ızgara ile belirlenmiş ortamlarda algoritmanın performansı ölçülmüştür. Performans ölçümünde yakıt maliyeti gibi kısıtlar göz önünde bulundurulmuştur. Çalışma sonucunda elde edilen sonuçlara göre A* algoritmasını özellikle maliyet optimizasyonu ve yol uzunluğunun azaltılması noktasında önemli geliştirmenin olduğunu belirtmişlerdir (Chen ve diğ., 2018).

Ciu & Wang (2021), sunmuş oldukları çalışmalarında çok katmanlı yol planlama algoritması tabanlı pekiştirmeli öğrenimi modelini kullanmışlardır. Çalışmalarında global ve yerel bilgileri tutacak şekilde iki katman kullanmışlardır. Her iki katmanda engelden sakınacak biçimde uyumlu çalışacak biçimde tasarlanmıştır. Ayrıca gerçek zamanlı olarak oluşturulan yol için B-Spline yöntemi kullanılarak yolun yumuşatılması sağlanmıştır (Ciu & Wang, 2021).

Feng ve diğ., (2021), sunmuş oldukları engel pozisyonu tahmin ve yapay potansiyel alan yaklaşımından oluşan hibrid yöntemlerinde markov zincirleri temelli gezinge değerlendirme algoritması ile dinamik engelleri önceden tahmin etme ve sonucunda local minima, local oscillationdan arınmış belirsiz ortamlarda güvenli yol planlama için yeni bir yaklaşım ortaya koymuşlardır (Feng ve diğ., 2021).

Cao ve diğ., (2022), sunmuş oldukları çalışmada 3D uzayda Rapidly-exploring Random Tree–Connect (RRT-Connect) ve YPA method algoritmasını hibrid olarak kullanarak İHA'ların yol bulma ve dinamik olarak engelden sakınma süreçlerini geliştirmişlerdir. Sonuç olarakda algoritma ile belirlenen yol üzerinde B-Spline uygulayarak daha yumuşak bir yol bulmuşlardır (Cao ve diğ., 2022).

Wu ve diğ., (2022), sunmuş oldukları çalışmada derin sinir ağları kullanarak gerçek zamanlı olarak kompleks ortamlarda yol planlama problemi çözümü için yeni bir model geliştirmişlerdir ve çalışmalarında verimlilik, başarım oranı ve oluşturulan yolun kalitesi açısından önemli iyileştirmeler olduğunu belirtmişlerdir (Wu ve diğ., 2022).

Wu ve diğ., (2023), sunmuş oldukları yeni yaklaşım ile ortamın küçük parçalara ayrılması ve kompleksiliğinin azaltılması sonucunda Çoklu İHA iş birliği ile yol planlama probleminin çözümü için eş-evrim algoritmasını kullanmışlardır. Bunun için öncelikle yol planlama problemini analiz etmiş ve modellemişlerdir oluşturdukları modele kısıt olarak İHA arasındaki ilişki parametrelerini eklemişlerdir (Wu ve diğ., 2023).

Matlekovic ve diğ., (2023), sunmuş oldukları çalışmada yol planlama probleminin çözümü için Min–Max K-Chinese Postman Probleminin (MM K-CPP) farklı bir uygulamasını matematiksel olarak formüle etmişlerdir. Uyguladıkları yöntem ile genel yol planlama problemi için birden fazla İHA kullanıp ve her bir İHA'nın ayrıştırma yapılmış yolun her bir segmentini minumum zaman diliminde dolaşmasını incelemişlerdir (Matlekovic ve diğ., 2023).

Li ve diğ., (2023), sunmuş oldukları geliştirmiş grey wolf optimizer (GWO) algoritması ile yol planlama probleminin optimal çözümü için gri kurtların avlanma ve sosyal davranış becerilerine benzer iteratif arama tabanlı yöntem geliştirmişlerdir. Geliştirdikleri yöntem içerisinde kullandıkları ağırlık fonksiyonları ile multi-modal fonksiyon, autoregressive fonksiyon ve sigmoid fonksiyonlarla yöntemin geleneksel yönetmelere göre daha iyi sonuç verdiğini gözlemlemişleridir (Li ve diğ., 2023).

Jamshidi ve diğ., (2023), sunmuş oldukları çalışmada İHA yol problemi çözümü için controller area network (CAN) kullanarak herhangi bir host bilgisayar kullanmadan çoklu microcontroller yapısı içerisinde GA uygulamasını paralel ve tekli olarak uygulamış ve sonucunda paralel uygulamanın daha iyi sonuç verdiğini gözlemişlerdir (Jamshidi ve diğ., 2023).

Bashir ve diğ., (2023), sunmuş oldukları çalışmada yol planlama probleminin çözümü için disk tabanlı belirsizlik modelini kullanmışlardır. Çevresel koşullara dikkate alınarak engelleri dikdörtgen olarak modellemişler ve bu engeller çevresinde tanımlanan noktalar üzerinden yol oluşturmuşlardır. Çalışmaları sonucunda uygulamanın eş zamanlı uygulamalarda kullanılabileceğini belirtmişlerdir (Bashir ve diğ., 2023).

Zhang ve diğ., (2023), sunmuş oldukları çalışmada insansız hava aracı yol planlama probleminin çözümü için geliştirilmiş White Sharks Optimization (WSO) algoritması

kullanmışlardır. Öncelikle uçuş alanını verimli bir şekilde keşfetmek için 3D alan arazi ortamının ve kısıtlama fonksiyonu oluşturmak için arazi matrisi kullanılır ayrıca algoritmanın etkinliğinin artırılması amacıyla çoklu yörünge araması, doğrusal olmayan yakınsama faktörü çalışmada kullanılmıştır (Zhang ve diğ., 2023).

1.5. İHA Temel Kavramlar

Genel olarak İHA'nın başlangıç noktasından hedef noktasına ulaşma durumu İHA'nın anlık koordinatı ile hedef noktasının koordinatı arasındaki mesafenin önceden belirlenen sabit bir değerden küçük olma durumu ile kontrol edilir. Bu durum matematiksel olarak $\|x^{iha} - x^{hedef}\| \leq \varepsilon$ şeklinde ifade edilir. İHA oldukça hızlı hareket eden bir platformdur, üzerinde bulunan farklı sensörler vasıtasıyla uçuş ortamının bilgileri alınabilmektedir. Kinematik, çevresel, görev tanımlı kısıtlar, sensörler tarafından algılanan verinin bazı durumlarda işlenmesinin zor olması ve belirsizliği gibi durumlarda yol planlama sürecinin oldukça hızlı olarak yenilenebilmesi gerekebilmektedir. Çalışma kapsamında İHA platformunun araç kinematiklerinin yol planlama kapsamında yapılması gereken manevralara uygun olduğu varsayılacaktır. Örnek olarak engelden sakınmak için gerekli olan dönme manevrasını minimum dönme yarıçapından bağımsız yapabildiği kabul edilecektir. Çalışma kapsamında İHA genel yol planlaması problemi için geliştirilen yöntemin farklı algoritmaların beraber kullanımı ile başarımı yüksek bir sonuç alınması hedeflenmektedir. İHA platformunun uçuş rotasında hareketini engelleyecek nesneler olabileceği gibi hareketini serbest olarak yapabileceği ortamlar bulunabilmektedir.

1.5.1.Terminoloji

Goerzen ve diğ. (2014) tarafından aşağıda belirtilen tanımlamalar yapılmıştır (Goerzen ve diğ., 2014).

Araç Nesnesi *:* Uçuş ortamında hareket etme yeteneğine sahip ve genel olarak iki veya üç boyutlu uzay içerisinde pozisyon ve yönlenme vektörü ile tanımlanan araç.

Yeryüzü Nesnesi *:* Aracın içinde bulunduğu fiziksel ortamdır. Araç üç boyutlu öklit uzayında olabileceği gibi üç boyutlu uzay içine gömülü iki boyutlu bir yüzeyde de olabilir.

Konfigürasyon : Konfigürasyon rigid olarak aracın şeklinin vektörler ile tanımlanması 6 dereceli (3 adet hareket edebilecek koordinat noktası ve 3 adet yönelim noktası) serbestlik modeli ile gösterilmesidir.

Konfigürasyon Uzayı : Olası bütün araç konfigürasyonları kümesidir. C-space olarak adlandırılır.

Serbestlik Derecesi : Her bir konfigürasyonun temsil edilmesi için gerekli koordinat sayısı.

Yeryüzü Konfigürasyonu : Serbest uzay ve engel uzayı olarak ikiye bölünür. Uzayda bu iki bölge birbirinden ayrı tanımlanır.

Serbest Uzay : Aracın herhangi bir engelle karşılaşmadan serbest olarak hareket edebildiği bölgedir.

Yol ve Yörünge *:* Yol konfigürasyon uzayında başlangıç noktasından hedef noktasından çizilen bir eğridir. Yörünge ise belirlenen yola zaman parametresinin eklenmesi ile bulunur. Araç belirli bir (t) anı sonrasında hangi noktada olduğunu belirler.

Yol planlama aşaması temelde genel yol planlaması ve yerel yol planlaması olmak üzere iki gruba ayrılır;

Genel yol planlaması; durağan uçuş ortamları için uygundur, genel olarak gerçek zamanlı hesaplamalara çok da uygun değildir. RRT algoritması genel yol planlamasına uygun bir algoritmadır.

Yerel yol planlaması; Her bir iterasyonda uçuş ortamının belirli bir bölgesinin işleme alınarak yerel bölge için optimum yolun bulunması ve hedefe doğru yönelim sürecinin devam ettirilmesidir. YPA yaklaşımı lokal olarak yol planlaması yapabilen bir yöntemdir.

1.6.Tez Organizasyonu

Tez çalışmasının;

İkinci bölümünde, yol planlama probleminin çözümünde kullanılan algoritma ve yöntemlerle ilgili bilgiler verilmiştir.

Üçüncü bölümünde, geliştirilen hibrid yöntemin çalıştırılması sonucunda oluşturulan veri kümesi kullanılarak eğitim süreci tamamlanan yapay sinir ağı modeli ile ilgili bilgilendirmeler yer almaktadır.

Dördüncü bölümünde; çalışma kapsamında kullanılan yazılım platformları (ROS, Gazebo, PX4) ilgili detaylı bilgiler verilmiştir.

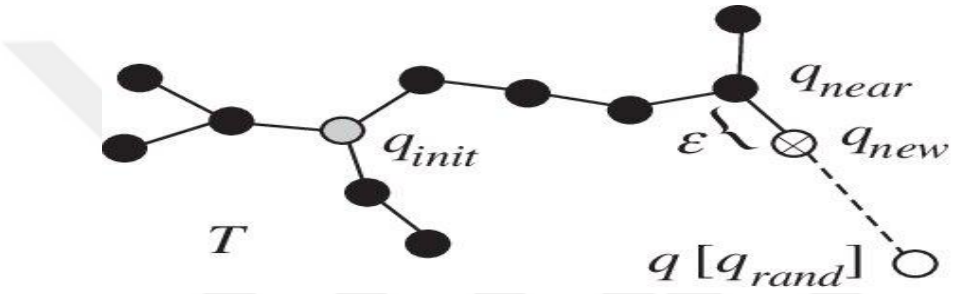
Beşinci bölümünde, geliştirilen yöntem ile ilgili detaylı bilgiler verilmiştir.

Altıncı bölümünde; ise bu tez çalışması kapsamında elde edilen deneysel sonuçlar, bulgular ve sonraki çalışmalar için öneri ve yorumlara yer almaktadır.

2.YOL PLANLAMA YÖNTEMLERİ

2.1.Random Exploring Rapid Tree (RRT)

RRT algoritmasını grafiksel olarak Şekil 2.1'deki gibi tanımlayabiliriz. İlk olarak başlangıç noktası seçilir, daha sonra rastgele olarak bir nokta alınır ve ağaç yapısı içerisindeki en yakın nokta rastgele seçilen noktaya göre bulunur, bu iki nokta arasında engellerle karşılaşmayacak şekilde yeni bir nokta seçilir ve bu yeni nokta ağaca eklenir ve iterasyona hedef noktaya gelene kadar devam edilir. RRT algoritması oldukça hızlı çalışana bir algoritmadır, ancak çoğu zaman optimum sonucu vermeyebilir.



Şekil 2.1. RRT algoritması genel gösterimi (Yoshida, 2005)

RRT algoritmasına ait sözde kod ise Tablo 2.1'de gösterilmektedir.

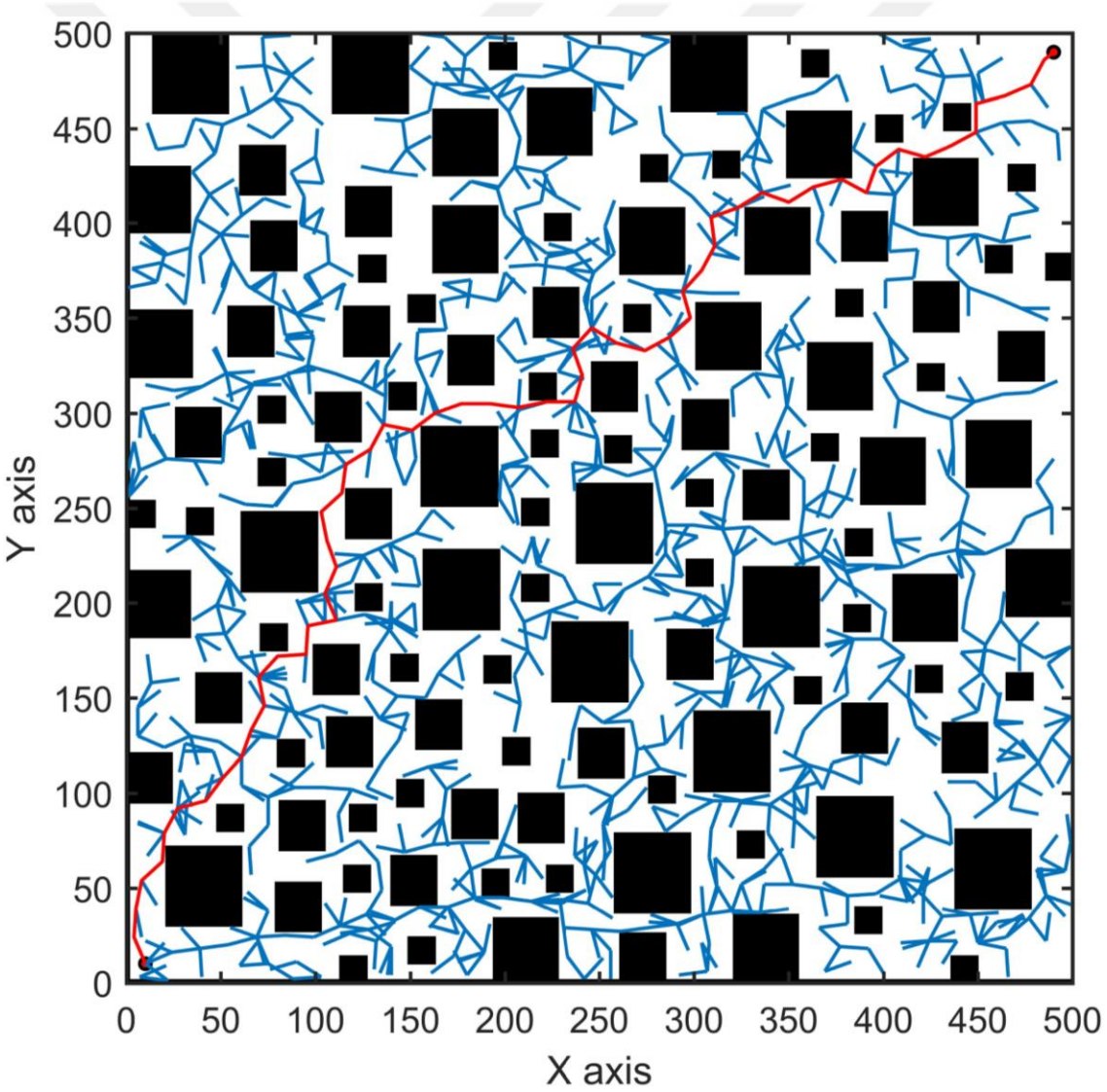
Tablo 2.1. Temel RRT algoritmasının sözde kodu

Algoritma 1. Temel RRT algoritmasının sözde kodu

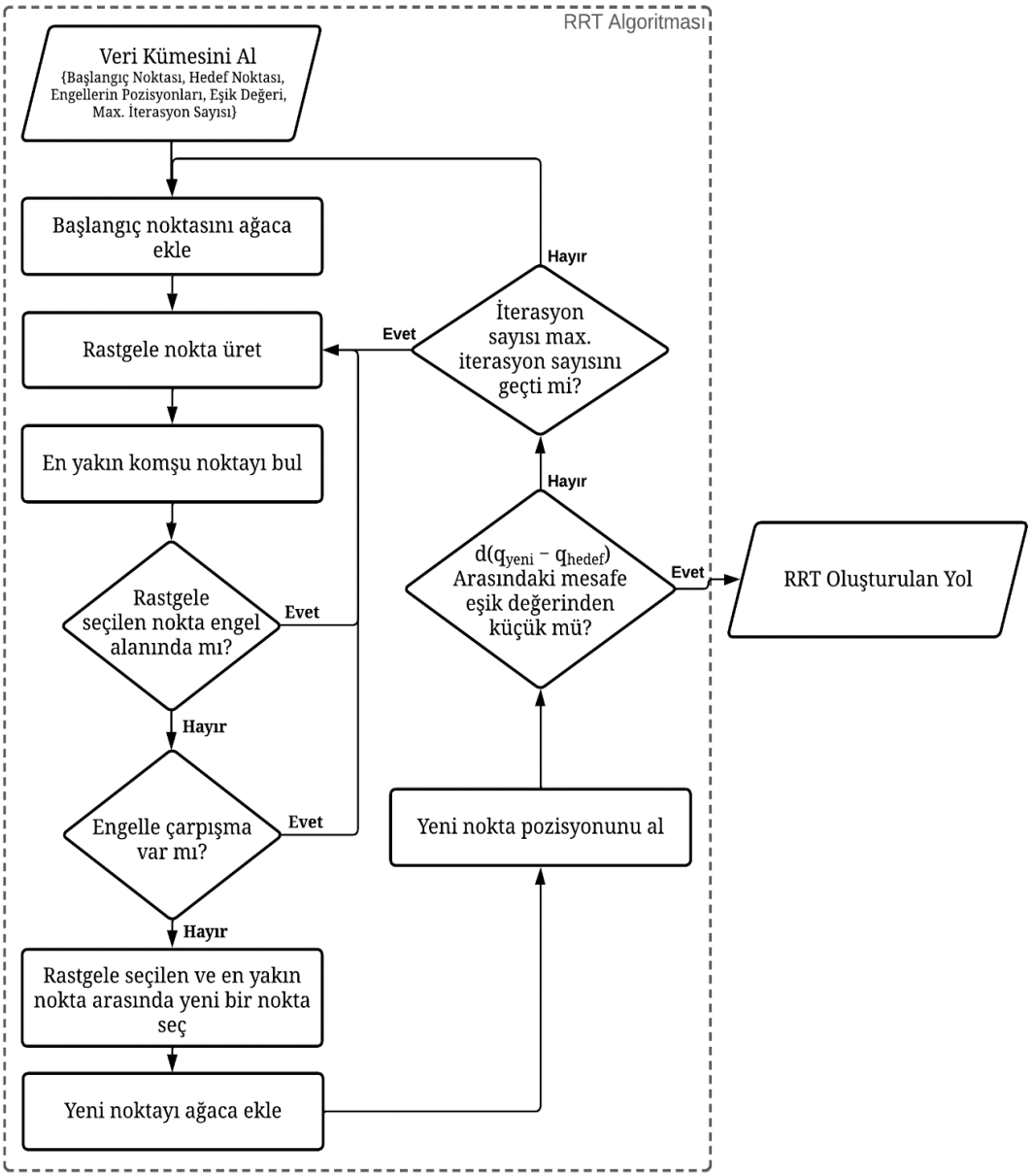
Girdi: *Başlangıç Parametreleri* q_{start}, q_{goal}, *engel, Örnekleme Sayısı K, Adım Büyüklüğü* δ
Çıktı: *RRT Graf G, Yol*

```
düğüm = q_start ;
for i = 1 to K do
        q_rand = Yeni_Örnek();
        q_near ← En_Yakın_Düğüm(düğüm, q_rand );
        q_new ← Yeni_Düğüm_Belirle(q_near , q_rand , δ);
  if Engel_Çarpışma(q_new ) == False then
        düğümler.ekle(q_new );
        return Tekrarla;
  if Uzaklık_Hesapla(q_new − q_goal ) < Error then
        return Hedef;
  else
        return Devam;
final ;
return Graf;
```

RRT temel olarak yukarıda belirtilen akışa göre çalışır. Öncelikle başlangıç, hedef ve engeller konfigürasyon uzayında tanımlanır ve rastgele olarak oluşturulacak nokta sayısı belirlenir. Rastgele belirlenen nokta ile yol ağacındaki en nokta arasında belirlenen mesafede seçilen nokta eğer engel ile karşılaşma durum yok ise ağaca eklenir. Bu iterasyon hedef noktaya belli bir mesafeye gelen kadar tekrarlanır. RRT algoritmasının değişik bir türü olarak RRT* algoritması geliştirilmiştir (Karaman & Frazzoli, 2011). RRT* algoritması hedefe giderken daha düz çizgiler oluşturur. RRT algoritmasına yeni bir adım daha ekleyerek optimal yol için öncelikle optimal yol üzerindeki ana düğümleri kontrol eder. Zaman karmaşıklığı RRT algoritması ile aynı olsa da çalışma zamanı açısından avantaj sağlar.



Şekil 2.2. Engelli ortamda RRT ağaçları ve belirlenen yol (Wang ve diğ., 2020)

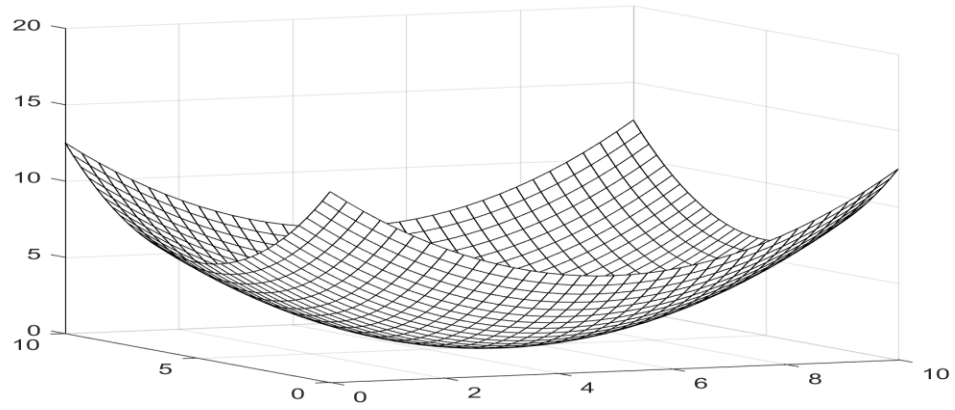


Şekil 2.3. RRT algoritması akış şeması (Gültekin ve diğ., 2023)

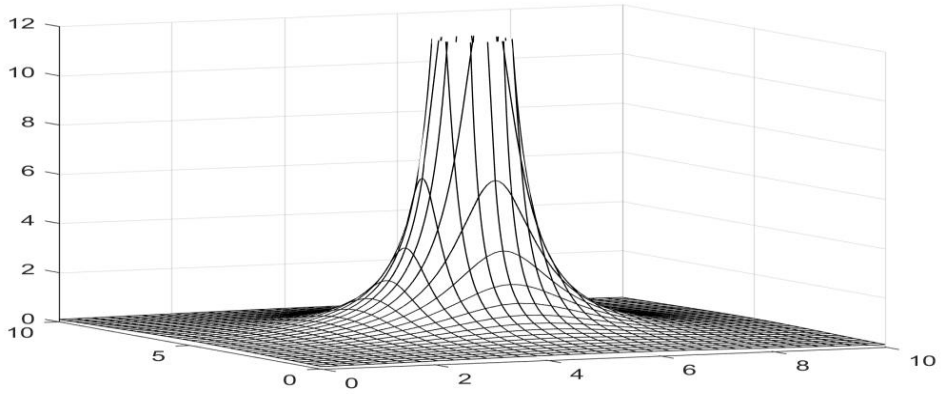
2.2.Yapay Potansiyel Alan (YPA)

YPA yaklaşımı İHA başta olmak üzere otonom harekete edebilen araçlarada uygulanabilen vektör tabanlı yaklaşımlardandır. Bu yaklaşım ilk olarak mobil robotun potansiyel alan içerisinde hareketinin çekici ve itici kuvvetlere nasıl gösterileceğini açıklayan Khatip (1986), tarafından ortaya atılmıştır (Khatip, 1986). YPA'da, yol planlama problemlerinde kullanılan geometri tabanlı yaklaşımlarına benzer olarak uçuş ortamı grid (ızgara) olarak temsil edilir. Ancak YPA tüm uçuş alanında her bir grid

hücresinden bağımsız olarak İHA'ya etki eden tüm iten ve çeken kuvvetleri aynı anda değerlendirir. YPA temel olarak İHA benzeri otonom hareket edebilen bir platformun yüksek potansiyel alandan düşük potansiyele doğru hareket ettiğini ve hareket sırasında çarpışma engelleme ve çarpışmadan sakınma manevralarının yapılabildiğini modellemektedir. Şekil 2.4. ve Şekil 2.5 'de global çeken ve iten potansiyel kuvvet alan gösterimleri verilmiştir (Yang ve diğ., 2023)



Şekil 2.4. Global çeken potansiyel kuvvet



Şekil 2.5. Global iten potansiyel kuvvet

Yüklü bir parçacığın elektrik alanda hareketine benzer olarak İHA'nın yüksek potansiyel alandan düşük potansiyel alana doğru hareket ettiği varsayılır. Hedef düşük potansiyel , engeller ise düşük potansiyel alanları temsil eder.

YPA metodu çeken ve iten kuvvetlerin etkisi ile oluşan bileşke kuvvet doğrultusunda robotun hareketinin hedef koordinata doğru yönelmesini sağlayan ve dinamik yol planlama yaklaşımlarında sıklıkla kullanılan önemli bir metottur. YPA metodu düşük hesaplama maliyeti ve sürekli hedef koordinata doğru yönelim sağlar.

Çeken potansiyel alan $U_{att}(q)$ ve bu alanın gradyan hesaplaması ile oluşan çeken potansiyel kuvvet $F_{att}(q)$ ile ifade edilmiş ve Denklem (2.1) ve Denklem (2.2) 'de gösterilen eşitliklerle ile hesaplanmaktadır.

$$U_{att}(q) = \frac{1}{2} k_{att} \|q - q_g\|^2 \tag{2.1}$$

$$F_{att}(q) = -\nabla U_{att}(q) = -k_{att} \|q - q_g\| \tag{2.2}$$

İfadede gösterilen k_{att} yerçekimsel kazanç kuvveti, $\|q - q_g\|^2$, q noktasından hedef noktaya olan öklit mesafesini tanımlar, hedef noktaya yaklaştıkça çeken potansiyel kuvvet değeri azalır.

İten potansiyel alan $U_{rep}(q)$ ve bu alanın gradient hesaplaması ile oluşan çeken potansiyel kuvvet $F_{rep}(q)$ ile ifade edilmiş ve Denklem (2.3) ve Denklem (2.4) 'de gösterilen eşitliklerle ile hesaplanmaktadır.

$$U_{rep}(q) = \begin{cases} \frac{1}{2} k_{rep} \left(\frac{1}{\|q - q_{obs}\|} - \frac{1}{d_0} \right)^2, \|q - q_{obs}\| \leq d_0 \\ 0, \|q - q_{obs}\| > d_0 \end{cases} \tag{2.3}$$

$$F_{rep}(q) = \begin{cases} k_{rep} \left(\frac{1}{\|q - q_{obs}\|} - \frac{1}{d_0} \right), \frac{1}{\|q - q_{obs}\|^2}, \|q - q_{obs}\| \leq d_0 \\ 0, \qquad \|q - q_{obs}\| > d_0 \end{cases} \tag{2.4}$$

İfadede gösterilen k_{rep} itme kazanç kuvveti, $\|q - q_g\|^2$, q noktasından engele olan öklit mesafesini tanımlar.

Yukarıda verilenlere göre potansiyel alan Denklem (2.5) ve potansiyel kuvvet birleşik Denklem (2.6) 'da gösterilen eşitliklerle ile hesaplanmaktadır.

$$U_{tol}(q) = U_{att}(q) + U_{rep}(q) \tag{2.5}$$

$$F_{tol}(q) = F_{att}(q) + \sum_{i=1}^{n} F_{rep,i}(q) \tag{2.6}$$

YPA'nın düşük hesaplama maliyeti ve hedef noktaya doğru sürekli yönelim, yerel yol planlama çok fazla kullanılan bir yöntem olma gibi avantajlara sahip olsada yerel minimum tuzağı, ulaşılamayan hedef ve kolaylıkla salınım durumunun oluşması gibi dezavantajlarıda mevcuttur (Shin & Kim, 2021) . YPA özellikle literatürde yer alan diğer yol planlama algoritmaları ile hibrid olarak çalıştığında, dezavantajları ortadan kalkabilmektedir. YPA yaklaşımı dinamik uçuş ortamlarında anlık oluşabilecek engellerden sakınmak için yerel yol planlama yaklaşımı olarakda kullanılmaktadır. YPA vektör tabanlı çalışan bir yöntemdir. Çeken ve iten kuvvetlerin bileşkesi alınarak sürekli hedef koordinata doğru yönelim yapılması sağlanır.

YPA yönteminin temelinde gradyan vektör hesaplaması vardır. Gradyan vektör belirlenen bir nokta için en hızlı artış yönünü ve büyüklüğünü belirler. Optimizasyon problemlerinde sıklıkla kullanılır. Gradyan tanımı kısmi türevle ile ilgilidir. Örneğin fonksiyonun x ve y değişkenlerine göre alınan türevi vektör olarak gradyan vektörü oluşturur. Gradyan vektörün yönünün pozitif veya negatif olması, artan ya da azalan bir eğime sahip olduğunu gösterir. Eğer yön pozitif fonksiyonun artan bir eğime sahip olduğunu, negatif ise azalan bir eğime sahip olduğunu gösterir. Artış yapılan nokta bir çukur veya yerel maksimum noktası, azalışın olduğu nokta çukur veya yerel minumum noktası olabilir. İHA yol planlamada gradyan vektörler YPA yöntemi içerisinde kullanılır. İHA yol planlamada uçuş güvenliği, maliyetlerin azaltılması önemli kavramlardır ve bu kavramlar açısında optimizasyon yapılması gerekir. Gradyan vektörleri uçuş alanı içerisinde bulunan yükseklik değerlerinden türetilebilir. İHA'nın başta kinematik ve diğer kısıtları sebebiyle uçuş verimliliği uçuş alanının topografik durumuna bağlıdır. Uçuş ortamı çok fazla engebeli veya dağlık olabilir. YPA, skaler bir alan üzerinden gradient hesaplaması sonrasında İHA platformunun hareketinin modellenebildiği vektör alanın oluşmasını sağlar. Vektör alanının şiddet ve en yüksek değişimin yönü ile İHA'nın bir sonraki adımda hareket bilgisini oluşturur. YPA yol planlama yaklaşımında başarımın artırılması için vektörel alan içerisinde engel ve hedef noktanın modellenmesi gerekmektedir. Mümkün mertebede modellemenin gerçek dünya şartlarına en uygun olarak yapılması beklenir ancak buradaki en büyük kısıt modellemedeki karmaşıklık düzeyi arttıkça işlem maliyetinin de artmasıdır. YPA yaklaşımında sabit engeller olduğu gibi uçuş ortamındaki diğer hareketli hava araçları da engel olarak tanımlanabilir.

2.3.Yol Planlama Yöntemleri

Literatürde yol planlama problemine çözüm getiren çok sayıda ve farklı algoritma ve yöntem mevcuttur. Tablo 2.2'de bu yöntemlerde temel olarak kullanılan bazı algoritmalar gösterilmiştir.

Tablo 2.2. Literatürde yaygın olarak yol planlama algoritmaların karşılaştırılması

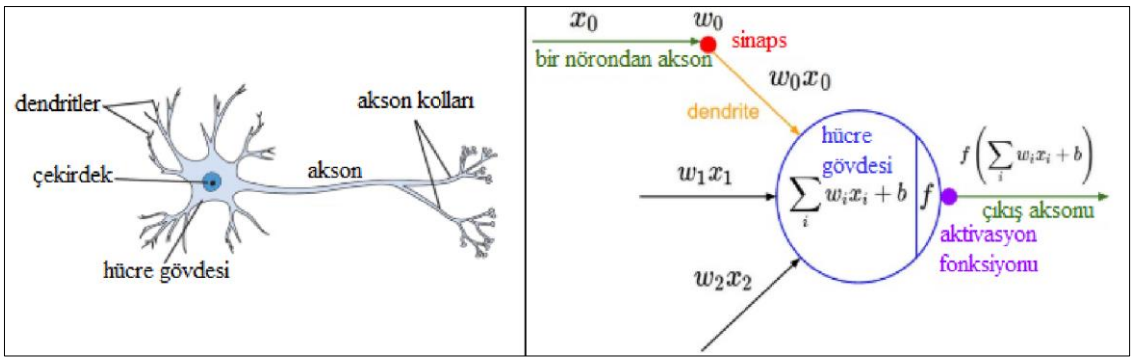
Algoritma	Ref.	Avantajları	Dezavantajları	Kategori
RRT	**LaValle, 1998**	Kolay uygulama, yol bulma garantisi	Her zaman optimum sonucu bulamama	Klasik
YPA	**Khatip, 1986**	Hızlı yakınsama , düşük işlem zamanı	Yerel minumum tuzağı, düşük verimlilik	Klasik
Dijkstra	**Dijkstra, 1959**	Kolay uygulama	Dinamik ortamlar için uygun değil	Klasik
A*	**Hart ve diğ., 1968**	Optimal sonuca yakın sonuç, ayarlanabilir sezgisel fonksiyon	Yüksek seviyede hesaplama maliyeti, bellek kullanımı	Sezgisel
GA	**Holland, 1975**	Farklı problemlere uygulanabilirlik, paralelleştirmeye uygun yapısı	Büyük ve karmaşık problemler için yavaşlık, sınırlı bellek kullanımı	Meta-Sezgisel
PSO	**Kennedy & Eberhart, 1995**	Kolay uygulama, hızlı sonuç, farklı problemlere uygulanabilirlik,	Her zaman istenilen sonuca erişememe, yanı parametrelerle farklı sonuçların oluşabilmesi	Meta-Sezgisel
Grey Wolf	**Mirjalili ve diğ., 2014**	Basitlik, az parametre, hızlı yakınsama	Probleme bağlılık, parametrelerin uygun biçimde ayarlanma zorunluluğu	Meta-Sezgisel
Önerilen Yaklaşım		Öğrenme ve geliştirmeye dayalı yapısı, geleneksel yöntemlere göre uyarlanabilir yol oluşturma	Gerçek zamanlı uygulama ve dinamik ortamlar için uygun olmaması	YSA/Hibrid

3.YAPAY SİNİR AĞI MODELİ

Bu tez çalışması kapsamında RRT ve YPA yöntemlerinin birlikte çalıştırılması ile yol veri kümesi oluşturulmuştur. Oluşturulan bu veri kümesi YSA modeli eğitimi için kullanılmıştır. Böylece YSA modeli eğitilerek yol oluşturmada kullanılacak noktalarının YSA ile seçilmesi sağlanmıştır. Literatür araştırmalarında geleneksel yöntemlerle oluşturulan ve YSA modelinin eğitimi için veri kümesinde farklı özelliklerinin kullanıldığı bir yöntemin olmadığı görülmüştür. Eğitimde kullanılacak ve bu yönde geliştirilen yapay sinir ağı modeli için Google tarafından oluşturulmuş Tensorflow kütüphanesi (URL-1). kullanılmıştır.

3.1.Yapay Sinir Ağı Nedir?

YSA, insan biyolojik sinir sistemlerinin matematiksel olarak modellemesi ile geliştirilmiştir. İlk defa Warren McCulloch ve Walter Pitts tarafından matematiksel olarak modellenmiştir (McCulloch & Pitts, 1943). YSA'ların öğrenme üzerine olan altyapısının yıllar içinde geliştirilmesi ile başta görüntü ve ses tanıma, doğal dil işleme, sağlık, finans ve askeri uygulamalar olmak üzere farklı alanlarda çok yoğun olarak kullanılmasına yol açmıştır. YSA'nın en basit formu 1957 yılında Frank Rosenblatt tarafından oluşturulmuş perceptron modelidir. Bu model çok sayıda girişe ağırlık uygulayarak ve aktivasyon fonksiyonundan geçirerek çıkışa 0 veya 1 değerlerini vermektedir (F. Rosenblatt, 1957). YSA'lar model içinde kullanılan katman sayılarının artması, öğrenme için farklı algoritmaların geliştirilmesi ve nöron çıkışlarını belirleyen aktivasyon fonksiyonlarının çeşitlenmesi ile günümüzde yapay zekâ teknolojilerinin geliştirmesinde önemli bir yer tutmaktadır. Derin öğrenme, çok katmanlı sinir ağlarının kullanıldığı bir sinir ağı modelidir. Tez çalışması kapsamında da derin sinir ağı modeli olan İleri Besleme Yapay Sinir Ağı (FFNN) kullanılmıştır. Görüntü tanıma uygulamalarında Evrişimli Sinir Ağları (Convolutional Neural Networks - CNNs), ses tanıma, doğal dil işleme gibi uygulamalarda Recurrent Sinir Ağları (Recurrent Neural Networks - RNNs) nöron ağları istenilen çıktıların alınabilmesi için sıklıkla kullanılan sinir ağlarıdır. Şekil 3.1'de biyolojik sinir hücresi ve YSA hücresinin örnek modelleri birlikte gösterilmektedir.



Şekil 3.1. Yapay sinir hücresinin modellenmesi (Cantemir, 2019)

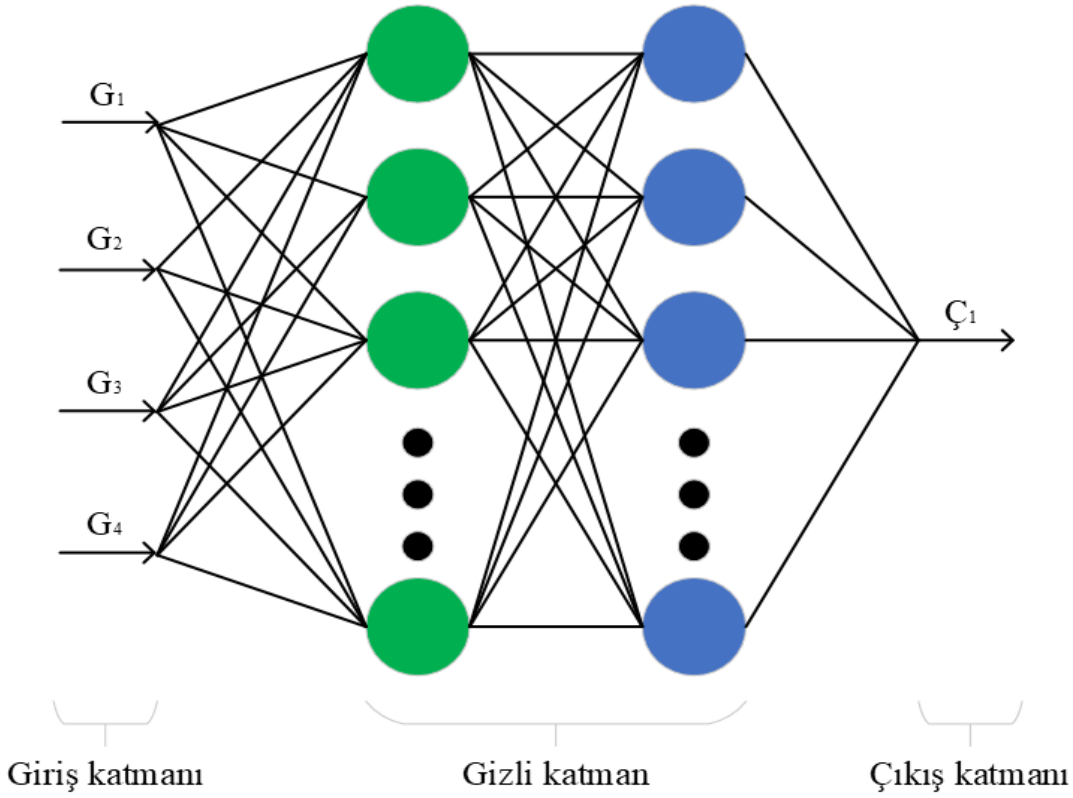
YSA'da, hücre çıkışı Denklem (3.1)'de gösterilen eşitlik ile hesaplanmaktadır. (Wang ve diğ., 2019). YSA'larda her bir nöron bir önceki katmandaki diğer nöronlara bağlıdır. Nöronlara başlangıçta rastgele ağırlık ve eşik değerleri atanır. Öğrenme sürecinde ağırlıklar ve eşik değerleri sürekli değiştirilerek hedeflenen çıkış değerlerine ulaşılması sağlanır. Sonuçta YSA tarafından belirlenen çıktıların doğruluk değerlerini tanımlar.

$$y_j^n = f\left(\sum_{i=1} w_{ji}^n \, x_i^{n-1} + \beta_j^n\right) \tag{3.1}$$

Buradaki , *y* çıkış değerini temsil ederken, *f(x)* aktivasyon fonksiyonunu, *x* giriş nöronunu, *w*, $(n-1)$'inci katmandaki *i*'den n'inci katmandaki *j* nöronuna ağırlıkları ve β eşiğini (bias) ifade etmektedir. Giriş verileri ağırlıklandırılarak toplayıcıya yönlendirilir ve burada bias değeri eklenerek elde edilen bu veriler doğrusal olmayan bir aktivasyon fonksiyonundan geçirilir ve ardından çıkışa ulaştırılır.

Bias, sinir ağlarında kullanılan bir parametredir ve genellikle her katmandaki her bir nöron için ayrı bir bias değeri bulunur. Bias, sinir ağının öğrenme kapasitesini artıran ve modelin daha iyi genelleme yapmasını sağlayan bir bileşendir ve genel olarak modelin esnekliğini, öğrenme yeteneğini artırmak, modelin başlangıç değerlerini ayarlamak, asimetri ve çeşitliliği artırmak için kullanılır.

YSA modeli olan İleri Besleme Yapay Sinir Ağı (FFNN) genel yapısı Şekil 3.2'de gösterilmektedir.



Şekil 3.2. YSA örnek ağ modeli

YSA'lar, katmanlar arasındaki bağlantı şekillerine göre FFNN ve Geri Beslemeli Ağlar (Recurrent Neural Networks veya RNNs) olmak üzere iki ana kategoride incelenir. İleri Beslemeli Ağlar, bir katmandaki hücre çıkışlarının bir sonraki katmandaki hücrelere girdi olarak aktarıldığı bir yapıya sahiptir ve bu tür ağlar, doğrusal olmayan sabit işlevleri yerine getirir. RNN ise çıktıları önceki katmanlardan gelen çıktılar veya o andaki durumlarla beslenen bir sinir ağı modelidir. Bu geri besleme mekanizması, geçmiş bilgilerin mevcut hesaplamalara dahil edilmesine ve zaman içinde değişen dinamik ilişkilerin modellenmesine olanak sağlar. Bu tür ağlar, geçmiş deneyim ve bilgi kullanarak gelecekteki çıktıları tahmin etmek veya mevcut girdilere bağlı olarak davranışları öngörmek gibi problemleri çözmek için kullanılır. Bu ağ yapısında, ağın yapısına ve çözümlemek istediği problemin durumuna bağlı olarak Ortalama Karesel Hata (Mean Square Error - MSE), Logaritmik Karesel Hata (Logarithmic Mean Square Error - LMSE), Çapraz Entropi gibi çeşitli hata ölçüm yöntemleri kullanılabilir. Çapraz entropi, sınıflandırma problemlerinde YSA'nın çıkış değerleri ile beklenen değerler arasındaki olasılıksal farkın hesaplanmasında yaygın olarak kullanılır. Çapraz entropi Denklem (3.2)'de gösterilen eşitlik ile hesaplanmaktadır. Hesaplanan olasılık değeri ile

beklenen olasılık değerleri arasındaki farkın artması, çapraz entropi kaybının arttığını gösterir. İdeal durumda, bu farkın 0 olması hedeflenir (Koşan ve diğ., 2019; Z. Zhang & Sabuncu, 2018).

$$H(p,q) = -\sum_{x \in X} p(x)\, log\, q(x) \tag{3.2}$$

Bu denklem temelde p ve q olasılık dağılımları sırayla gerçek ve tahmin değerlerden oluşan olasılık dağılımları olmak üzere olmak üzere bu iki olasılık dağılımının birbirine ne kadar benzemediğini ölçer.

Burada;

H(p, q): Dağılımların çapraz entropisi
p(x): p her bir sınıfın gerçek olasılık dağılımı
q(x): q her bir sınıfın tahmin edilen olasılık dağılımı
x: Olasılık sınıfı olmak üzere;

Hesaplanan çapraz entropi ne kadar düşük ise model tahminide o kadar başarılı olur. Başarılı bir model tahmini için çapraz entropi değerinin mümkün olduğunca minimize edilmesi istenir.

Çalışma kapsamında kayıp fonksiyonu olarak MSE kullanılmıştır. Bu fonksiyonun kullanımı ile model tahminlerinin gerçek değerlerden ne kadar farklı olduğu hesaplanır. MSE değeri ne kadar düşükse model tahminlerinin o kadar iyi olur. MSE, Denklem (3.3)'de gösterilen eşitlik ile hesaplanmaktadır.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{3.3}$$

Burada:

n veri noktalarının toplam sayısını ifade eder.
y_i gerçek değerleri ifade eder.
$\hat{y}_i$ tahmin edilen değerleri ifade eder.

YSA'lar, karmaşık verilerin sınıflandırılması, tahminlerin yapılması, örüntü tanıma, dil işleme, resim tanıma, işaretlerin tanınması, öneri sistemleri, nesne tespiti, kontrol sistemleri gibi geniş yelpazedeki alanlarda etkili bir şekilde kullanılabilirler. Ancak

başarılı sonuçlar elde edebilmeleri için doğru veri kümeleriyle eğitilmeleri gerekmektedir. Bununla birlikte, YSA'ların yaygın kullanım alanları aşağıdaki gibi gruplanabilir:

1. Doğal Dil İşleme: YSA'lar, doğal dildeki kelime ve cümle yapılarını öğrenme süreci sırasında tanıyabilme yeteneğine sahip olmalarıyla doğal dil işleme alanında yaygın olarak kullanılır.
2. Ses Tanıma: Ses tanıma sistemlerinde de YSA'lar sıkça kullanılır. Bir kişinin konuşmasını tanıyarak, söylenen kelimeleri veya cümleleri yazıya dönüştürebilirler.
3. Örüntü Tanıma: Nesneleri ve desenleri tanımlamak için YSA'lar kullanılır; bu nedenle robotik ve güvenlik sistemleri gibi alanlarda sıklıkla tercih edilir.
4. Kontrol Sistemleri: YSA'lar kontrol sistemlerinde de kullanılabilir. Örneğin, bir basınç tankını yönetmek veya bir robotun kontrolünü sağlamak için kullanılabilirler. Ayrıca, YSA'ların desteği ile kontrol sistemlerinin performansı artırılabilir.
5. Öneri ve Tahmin Sistemleri: Uygulama alanlarına bağlı olarak, YSA'lar kullanıcılara özel öneriler ve tahminler sunabilirler.
6. Finansal Tahminler: YSA'lar, borsa fiyatlarındaki değişimleri veya finansal verilerin tahmin edilmesi gibi alanlarda da etkili bir biçimde kullanılabilirler.

YSA'lar, öğrenme sürecinde veri kümelerindeki giriş-çıkış ilişkilerini tanımlar ve bu ilişkileri kullanarak yeni verileri sınıflandırabilir veya tahminler yapabilirler. Örneğin, bir görüntü tanıma YSA'sı, bir görüntünün içerdiği desenleri ve özellikleri analiz ederek görüntüyü sınıflandırabilir. Bu sebeple, YSA'lar yapay zekâ ve makine öğrenmesi alanlarında yaygın olarak kullanılır. Esnek ve adaptif yapıları sayesinde, geleneksel programlama yaklaşımlarına göre özellikle büyük ve karmaşık veri kümeleri üzerinde işlem yapmak için tercih edilirler.

Ayrıca, YSA'lar insan beyninin işleyişini taklit ettikleri için, eğitildikleri amaç doğrultusunda insan benzeri öğrenme ve karar verme yeteneklerine sahip olabilirler. YSA'ların eğitim sürecinde genellikle belirli adımlar bulunmaktadır. YSA modelinin geliştirilmesi bu adım sırası önemlidir. Bu adımların sırasıyla YSA eğitim sürecindeki temel aşamaları temsil ettiğini görebiliyoruz:

1. Veri Toplama: Eğitim için gerekli verilerin toplanması ve düzenlenmesi.
2. Veri Önişleme: Verilerin temizlenmesi, normalleştirilmesi, dönüştürülmesi veya düzenlenmesi gibi ön işleme adımlarının uygulanması.
3. Model Tasarımı: YSA mimarisinin oluşturulması; katman sayısı, türü, aktivasyon fonksiyonları gibi parametrelerin belirlenmesi.
4. Model Eğitimi: Verilerin YSA'na verilerek ağırlık ve bias değerlerinin hesaplanması ve bu değerlerin hata fonksiyonuna göre güncellenmesi işlemi.
5. Model Doğrulama: Eğitilmiş YSA modelinin, ayrı bir doğrulama veri kümesinde test edilmesi ve performansının ölçülmesi.
6. Model Ayarlaması: Modelin performansının değerlendirilmesi ve bazı parametreler ile öğrenme oranı, optimizasyon algoritması gibi hiper parametrelerin ayarlanması, ardından modelin yeniden eğitilmesi.
7. Model Dağıtımı: Eğitilmiş YSA modelinin kullanıma hazır hale getirilmesi ve gerçek dünya uygulamalarında kullanılabilir hale gelmesi.

Bu adımlar eğitilmek istenilen modele ve modelin iyileştirilmesine bağlı olarak tekrarlanabilir veya özelleştirilebilir. YSA'lar, Evrişimli Sinir Ağları (CNN), Yinelemeli Sinir Ağları (RNN), Derin Sinir Ağları (DNN), Uzun-Kısa Süreli Bellekli Sinir Ağları (LSTM) gibi farklı özelleştirilmiş modellere sahiptir. Her model türü, özel görevler için özel olarak tasarlanmış olup farklı yapısal özelliklere sahiptir. Bu farklı yapılar, belirli görevler için daha etkili olabilir ve öğrenme süreçlerini daha verimli hale getirebilir. Örneğin, CNN, görüntü tanıma gibi görevlerde etkilidir; RNN, sıralı veriler gibi zaman serisi verilerde ve doğal dil işleme alanında kullanılırken; LSTM, zaman serisi verilerindeki uzun vadeli bağımlılıkları modellemek için tercih edilir. Bu özelleştirilmiş modeller, belirli görevlere uygun yapısal ve işlevsel özellikler sunarlar.

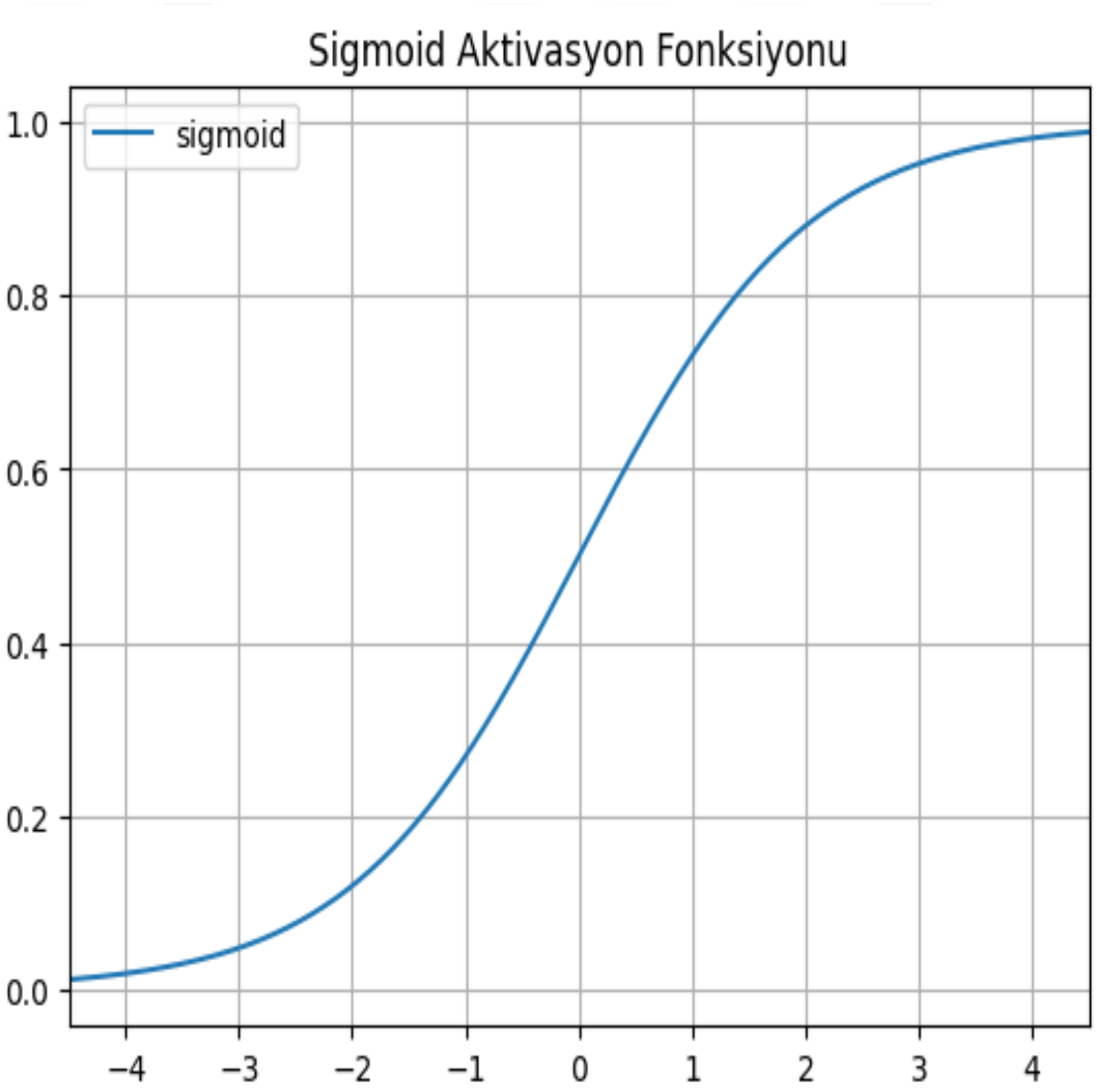
3.2.Aktivasyon Fonksiyonları

YSA'ların kullanıldığı alanlara uygun olarak farklı aktivasyon fonksiyonları kullanılmaktadır. Aktivasyon fonksiyonları belirli bir eşik değeri ve nöronun aldığı toplam değere göre bir çıktı değeri verir. Literatürde yaygın olarak Sigmoid, Hiperbolik Tanjant(tanh), Rectified Linear Unit (Relu) ve Softmax aktivasyon fonksiyonları kullanılmaktadır. (Sharma ve diğ., 2020).

3.2.1.Sigmoid Aktivasyon Fonksiyonu

YSA'da kullanılan aktivasyon fonksiyonları arasında en yaygın olarak kullanılan fonksiyondur. Bu fonksiyon giriş değeri olarak aldığı değeri 0 ile 1 arasında normalize etmektedir. *S(x)*, *x* girişine denk gelen çıkış ve *e* Euler sayısı olmak üzere Fonksiyon Denklem (3.4)'deki gösterilen eşitlik ile hesaplanmakta ve Şekil 3.3'de gösterilmektedir (Sharma ve diğ., 2020).

$$S(x) = \frac{1}{1 + e^{-x}} \tag{3.4}$$

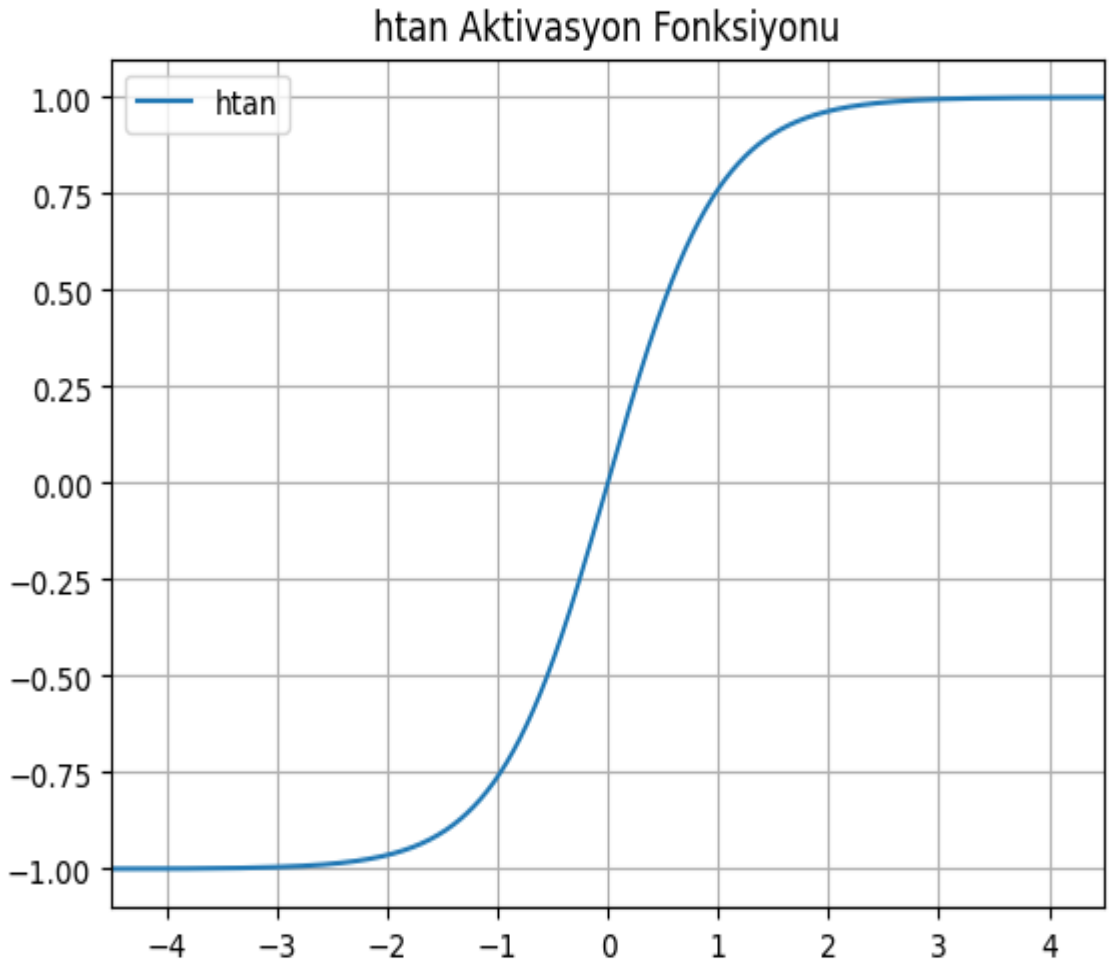


Şekil 3.3. Sigmoid aktivasyon fonksiyonu (Sharma ve diğ., 2020)

3.2.2.Hiperbolik Tanjant Aktivasyon Fonksiyonu

Hiperbolik Tanjant (tanh) aktivasyon fonksiyonu da YSA'da yaygın olarak kullanılan aktivasyon fonksiyonlarındandır. Sigmoid aktivasyon fonksiyonuna benzer bir şekilde çalışmaktadır. Hiperbolik Tanjant fonksiyonunun Sigmoid aktivasyon fonksiyonundan farkı giriş değeri olarak aldığı değeri -1 ile 1 aralığında normalize etmesidir. *Tanh(x), x* girişine denk gelen hiperbolik tanjant aktivasyon fonksiyonu ve *e* Euler sayısı olmak üzere Fonksiyon Denklem (3.5)'deki gösterilen eşitlik ile hesaplanmakta ve Şekil 3.4'te gösterilmektedir (Sharma ve diğ., 2020).

$$tanh\,(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{3.5}$$

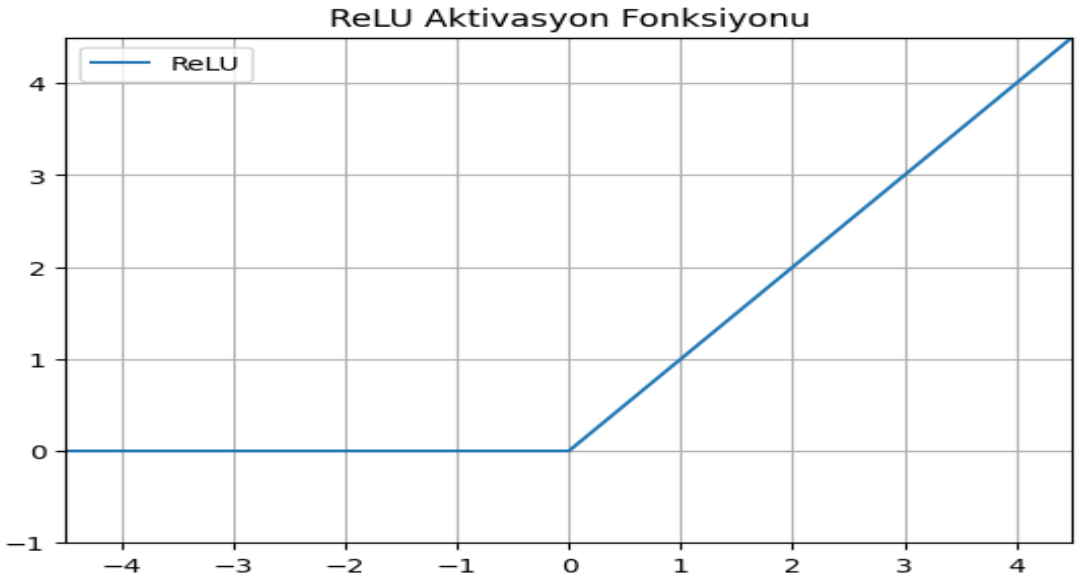


Şekil 3.4. Hiperbolik tanjant fonksiyonu (Sharma ve diğ., 2020)

3.2.3.ReLU (Rectified Linear Unit) Aktivasyon Fonksiyonu

Rectified Linear Unit (ReLU) aktivasyon fonksiyonu ağdaki tüm nöronların aktive olmasını engelleyerek hesaplama yükünü azaltmak ve ağın daha hızlı çalışmasını sağlamak gerektiğinde yaygın olarak kullanılmaktadır. ReLU fonksiyonu üretilen çıkış değerini 0 ile giriş değeri olarak verilen değer aralığında normalize etmektedir. Bu fonksiyon, giriş değeri x pozitifse, x değerini çıkış olarak verir; eğer x negatifse, sıfır değerini çıkış olarak verir. Yani, ReLU fonksiyonu, negatif değerlere sıfır çıkış üretirken, pozitif değerleri doğrudan ileterek sıfırın altındaki değerlere olan duyarlılığı azaltır Fonksiyon Denklem (3.6)'daki gösterilen eşitlik ile hesaplanmakta ve Şekil 3.5'te gösterilmektedir (Sharma ve diğ., 2020).

$$Relu\,(x) = (0, x) \tag{3.6}$$



Şekil 3.5. Relu fonksiyonu (Sharma ve diğ., 2020)

3.2.4.Softmax Aktivasyon Fonksiyonu

YSA'da ikiden fazla sınıflandırma yapılması gerekli olan problemlerde yaygın olarak kullanılmaktadır. Giriş değeri olarak almış olduğu değerin hangi sınıfa ait olduğunun olasılığını hesaplamaktadır. Softmax fonksiyonu bir vektörün elemanlarını 0 ile 1 aralığında olacak şekilde normalleştirir ve bu değerlerin toplamı 1 olacak şekilde ölçekler. Çıkış katmanı aktivasyon fonksiyonu olarak genellikle derin öğrenme modellerinde

kullanılmaktadır. Fonksiyon, Denklem (3.7)'deki gösterilen eşitlik ile hesaplanmakta (Sharma ve diğ., 2020) ve Şekil 3.6'da gösterilmektedir

$$\vartheta(\text{m. sınıf}) = \frac{e^{W_m A}}{\sum_{i=1}^{N} e^{W_i A}} \tag{3.7}$$

θ(m, s): Softmax fonksiyonunun *m'inci* sınıfa ait çıkış değeri.
m: Sınıf endeksi (1'den *N* kadar).

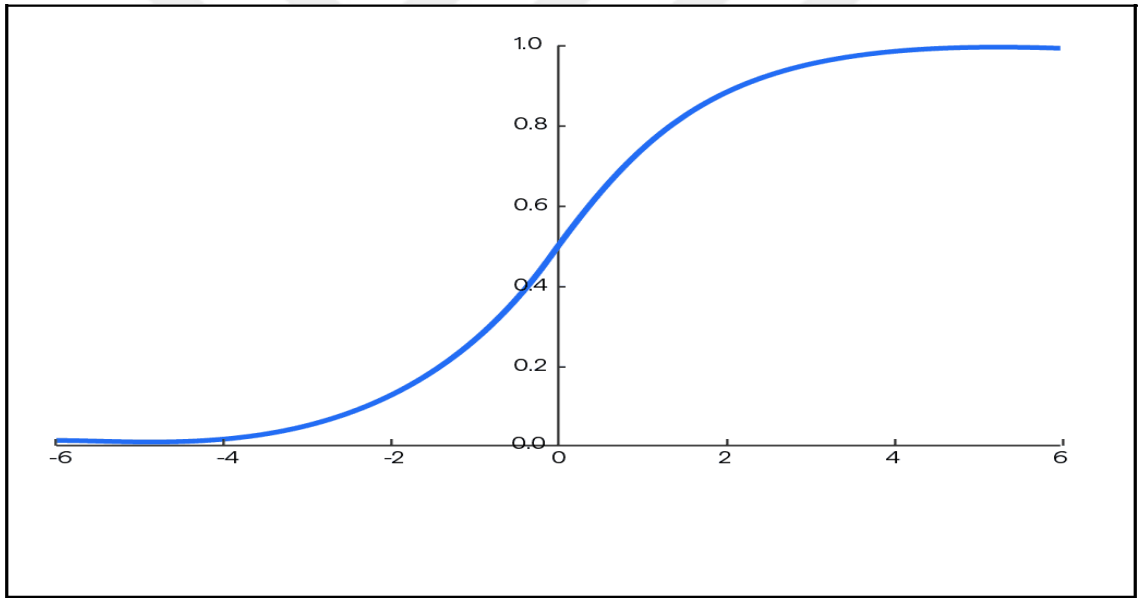
s: Giriş vektörü veya aktivasyon vektörü.

e: Euler sayısı (yaklaşık olarak 2.71828).

W_m*:* Sınıf *m* için ağırlık vektörü.

A: Giriş vektörü veya aktivasyon vektörü ile çarpılan ağırlık matrisi.

$\sum_{i=1}^{N} e^{W_i A}$ Tüm sınıfların ağırlıklı toplamı.



Şekil 3.6. Softmax Aktivasyon Fonksiyonu

3.3.Yapay Sinir Ağı Modeli Geliştirmede Kullanılan Yazılım Kütüphaneleri

YSA, derin öğrenme modelleri geliştirmek ve eğitmek için bir dizi yazılım kütüphanesi kullanılır. Bu kütüphaneler, karmaşık sinir ağı yapılarını tasarlamak, eğitmek ve değerlendirmek için gerekli olan çeşitli fonksiyonları içerir. Başlıca yapay sinir ağı yazılım kütüphaneleri arasında TensorFlow, Keras, PyTorch, Theano, Caffe ve MXNet gibi isimler bulunmaktadır. Çalışma kapsamında YSA modelinin geliştirilmesi ve eğitilmesi süreçlerinde TensorFlow, Keras kütüphaneleri kullanılmıştır.

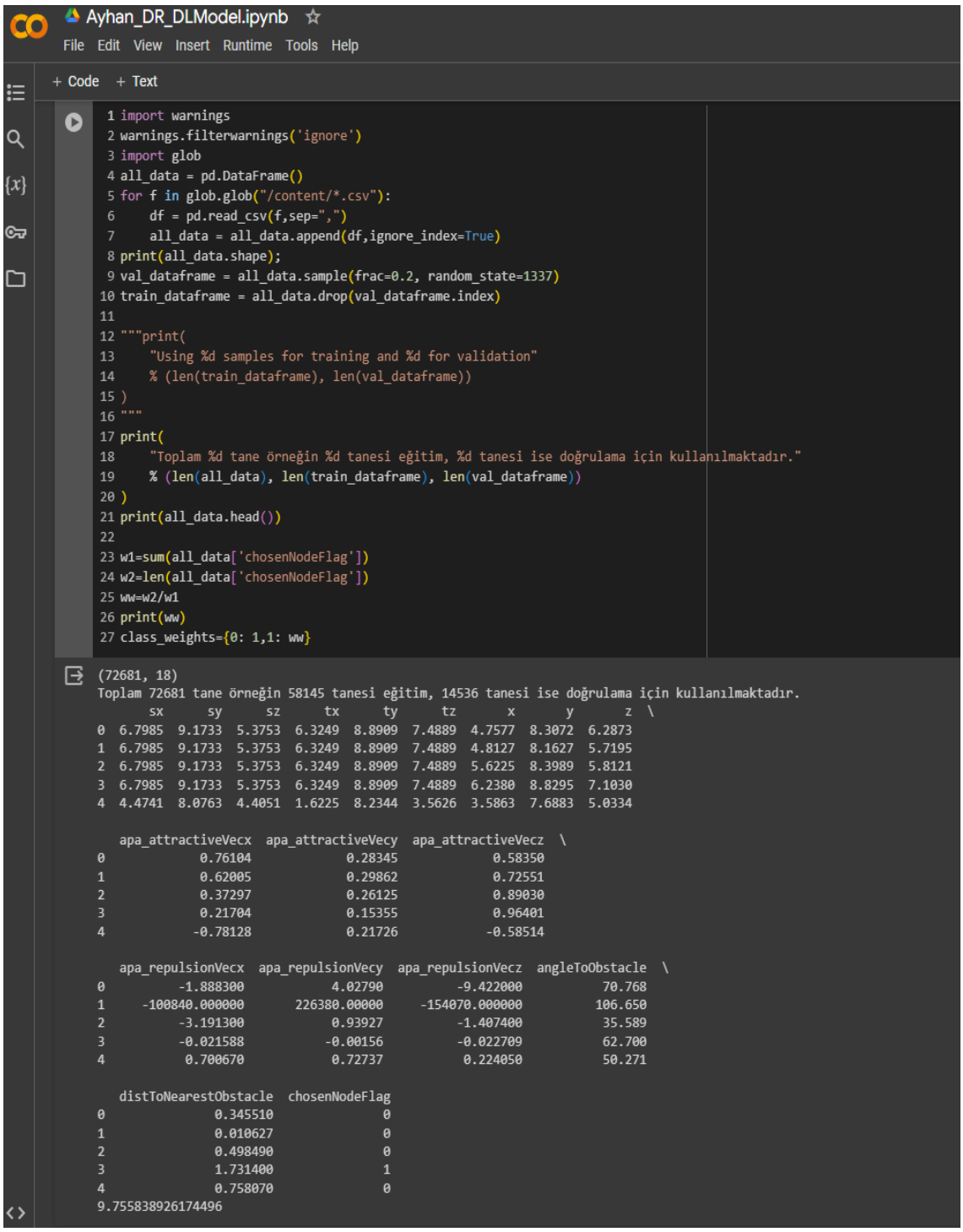
3.3.1.TensorFlow

TensorFlow (TF), Google tarafından geliştirilen bir makine öğrenimi ve derin öğrenme kütüphanesidir ve özellikle Theano'nun yerini alarak, geniş bir kullanım alanına sahiptir. Bu açık kaynaklı kütüphane, derin öğrenme modellerinin yanı sıra pekiştirmeli öğrenme, yapay sinir ağları ve çeşitli makine öğrenimi modellerini tasarlama, eğitme ve tahmin yapma süreçlerini desteklemektedir. TensorFlow'un esnek yapısı, görüntü işleme, doğal dil işleme, ses işleme ve öngörü analizi gibi çok çeşitli uygulama alanlarında etkili bir şekilde kullanılmasına olanak tanımaktadır.

TensorFlow'un temel felsefesi, hesaplamaları grafiklere ayırarak, modelin katmanları, işlevleri ve veri arasındaki ilişkileri bir "veri akış grafiği" şeklinde göstermektir. Bu grafik tabanlı hesaplama yöntemi, paralel işlemleri etkili bir şekilde gerçekleştirmeyi ve donanım hızlandırıcılar gibi kaynakları kullanarak işlem gücünü artırmayı mümkün kılar. Bu özellikler, büyük ölçekli ve karmaşık makine öğrenimi modellerinin etkili bir şekilde eğitilmesini ve uygulanmasını sağlar. TensorFlow, özellikle büyük ölçekli ve karmaşık yapay zekâ (AI) projeleri için tasarlanmıştır. İlk olarak 2015 yılında yayınlanan bu kütüphane, araştırmacılara, mühendislere ve geliştiricilere geniş bir yapay zekâ uygulama yelpazesi oluşturma imkânı sunmaktadır. Aynı zamanda TPU (Tensor Processing Unit), özellikle yapay zekâ (AI) ve derin öğrenme (deep learning) uygulamalarında kullanılmak üzere tasarlanmış bir çeşit işlemcidir. Google tarafından geliştirilen TPU'lar, özellikle TensorFlow ve diğer derin öğrenme çerçeveleriyle uyumlu bir şekilde çalışmak üzere tasarlanmıştır.

TensorFlow, C/C++ temellidir ve bir Python API aracılığıyla kullanıcılarına erişim sağlar. Ayrıca, Java, C++, Python ve Go gibi farklı programlama dillerini destekler ve bulut tabanlı geliştirme ortamlarında, özellikle Google Colab gibi platformlarda, kullanılabilmektedir. Java API'si şu anda deneysel destektedir ve mevcut sürümlerde kararlı kabul edilmemektedir, bu nedenle Java ve Scala geliştiricileri için şu aşamada tam olarak uygun olmayabilir (URL-1). Şekil 3.6'da Google Colab üzerinde çalışan ve pandas kütüphanesi kullanarak verilen bir CSV dosyasını işleyen ve önişlemden geçiren TensorFlow kodu örneği gösterilmektedir. Aynı zamanda bu kod örneğinde sınıf ağırlıklarının nasıl hesaplanacağı ve oranlarının ne olacağı ilgili kod parçası mevcuttur. Eğitim ve doğrulama veri kümesinin boyutları **len(train_dataframe)**,

len(val_dataframe) komutları ile belirlenir. Sınıf ağırlıkları (**class_weights**) belirlenir. Burada, sınıf 0 için ağırlık 1, sınıf 1 için ise **ww** değeri kullanılır. **glob** modülü, belirli bir deseni karşılayan dosyaları aramak için kullanılır. Bu örnekte, "/content/" dizinindeki tüm CSV dosyalarını seçmek için **glob.glob("/content/*.csv")** kullanılmıştır.



Şekil 3.7. Colab üzerinden Python dili ile tensörleri yeniden şekillendiren kod örneği

3.3.2.Keras

Keras, yüksek seviyeli bir yapay sinir ağı API'si olup, özellikle TensorFlow, Theano ve Microsoft Cognitive Toolkit (CNTK) gibi altta yatan düşük seviyeli kütüphanelerle entegrasyonu sağlayan açık kaynaklı bir Python kütüphanesidir. François Chollet tarafından geliştirilen Keras, hızlı prototipleme ve kullanım kolaylığı sağlamak amacıyla tasarlanmıştır (URL-2). Keras'ın temel özellikleri şunlardır:

Modüler ve Basitleştirilmiş Arabirim: Keras, modüler bir yapısı olan ve kullanıcılara basitleştirilmiş bir arayüz sunan bir kütüphanedir. Model oluşturma, katmanları ekleme, derin öğrenme modellerini eğitme ve değerlendirme işlemleri, Keras API'sini kullanarak oldukça basit bir şekilde gerçekleştirilebilir.

Evrensel Entegrasyon: Keras, altta yatan işlem kütüphanesi olarak TensorFlow'u kullanarak varsayılan olarak entegre edilmiştir. Ancak, Theano veya CNTK gibi diğer popüler derin öğrenme kütüphaneleri ile de uyumludur. Bu, Keras'ın esnek ve evrensel bir yapıya sahip olmasını sağlar.

Modüler Katmanlar: Keras, farklı tipte sinir ağı katmanları içeren geniş bir modüler katman kütüphanesine sahiptir. Yoğun (dense), konvolüsyonel (convolutional), reküran (recurrent), normalleştirme, düşey (embedding) ve diğer katman türleri gibi çeşitli katmanlar, model tasarımını destekler.

Uyumlu Optimizasyon ve Kayıp Fonksiyonları: Keras, çeşitli optimizasyon algoritmaları (SGD, Adam, RMSprop vb.) ve kayıp fonksiyonları ile uyumludur. Bu, kullanıcıların belirli bir görev için en uygun olanları seçmelerine olanak tanır.

Eğitim ve Değerlendirme Araçları: Keras, model eğitimi ve performans değerlendirmesi için kolay kullanımlı araçlar sağlar. Eğitim sırasında kayıp fonksiyonu ve metriklerin izlenmesi, aynı zamanda erken durma gibi eğitim sırasında kullanılan özelliklere sahiptir.

3.4.Yapay Sinir Ağı Modeli Eğitimi

Tez çalışması kapsamında RRT ve YPA algoritmalarının hibrid olarak kullanımı oluşturulan veri kümesi modelin eğitim için kullanılmıştır. Hibrid algoritmada başlangıç noktasından itibaren ilk olarak RRT algoritması çalıştırmış ve hedefe doğru yönelim için

rastgele seçilen noktanın YPA ile iten ve çeken kuvvetleri vektörel olarak hesaplanmıştır. Literatürde eksik olduğu tespit edilen bir Yapay zekâ modeli kullanılarak YSA modelini eğitimi geçekleştirilmiştir. Eğitilen bu YSA modeli kullanılarak elden edilen veriler ROS kütüphanesi kullanılarak geliştirilen Python kodu üzerinden Gazebo simülasyon ortamına uçuş simülasyonu için gönderilmiştir.

3.4.1.Veri Kümesi

Çalışma kapsamında YSA modelinin eğitimi için 10x10, 25x25 ve 50x50 harita üzerinden oluşturulan yol veri kümeleri kullanılmıştır. Oluşturulan her bir veri kümesi için YSA modeli eğitilmiştir. Tablo 3.1 'de hibrid algoritma tarafından üretilen veri kümesi örneği verilmiştir.

Tablo 3.1. Hibrid algoritma tarafından üretilen veri kümesi örneği

Start			**Pose**			**Target**			**Artificial Potential Field**			**Obstacle**			**X**	**Y**	**Z**
S_x	S_y	S_z	P_x	P_y	P_z	T_x	T_y	T_z	Att_x	Att_Y	Att_z	Rep_x	Rep_y	Rep_z			
4.1	1.9	7.8	3.85	4.66	2.6	2.7	8.7	0.8	-0.2	0.8	-0.3	1.37	2.69	-1.93	153	0.5	1
4.1	1.9	7.8	4.7	3.4	3.2	2.7	8.7	0.8	-0.32	0.86	-0.38	-0.66	-1.33	0.8	38.35	0.6	0
4.1	1.9	7.8	4.9	3.6	3.1	2.7	8.7	0.8	-0.37	0.84	-0.37	-0.16	-0.28	0.17	8.85	1.0	0
4.1	1.9	7.8	4.7	3.4	2.4	2.7	8.7	0.8	-0.33	0.9	-0.26	-0.06	-0.14	0.17	69.19	1.2	0
4.1	1.9	7.8	3.2	6.9	1.7	2.7	8.7	0.8	-0.23	0.88	-0.41	1.07	-1.01	0.13	34.3	0.7	1
4.1	1.9	7.8	3.6	2.9	2.4	2.7	8.7	0.8	-0.15	0.95	-0.26	-0.04	-0.13	0.07	126.7	1.2	0
4.1	1.9	7.8	2.6	6.5	3.6	2.7	8.7	0.8	0.03	0.6	-.079	23.53	-38.1	18.75	11.36	0.2	0

X = AngleToObstacle (Engelle yapılan açı)

Y = DistToNearestObstacle (En yakın engelle yapılan açı)

Z = ChosenNodeFlag (Noktanın yol içinde seçilme durumu)

Tensorflow içerisinde veriler modele girdi olarak verilmeden önce normalizasyon işlemi yapılır. TensorFlow'da veri normalizasyonu, genellikle girdi verilerinin belirli bir aralığa ölçeklenmesi işlemidir. Normalizasyon, modelin daha hızlı eğitilmesine ve daha iyi sonuçlar elde etmesine yardımcı olabilir.

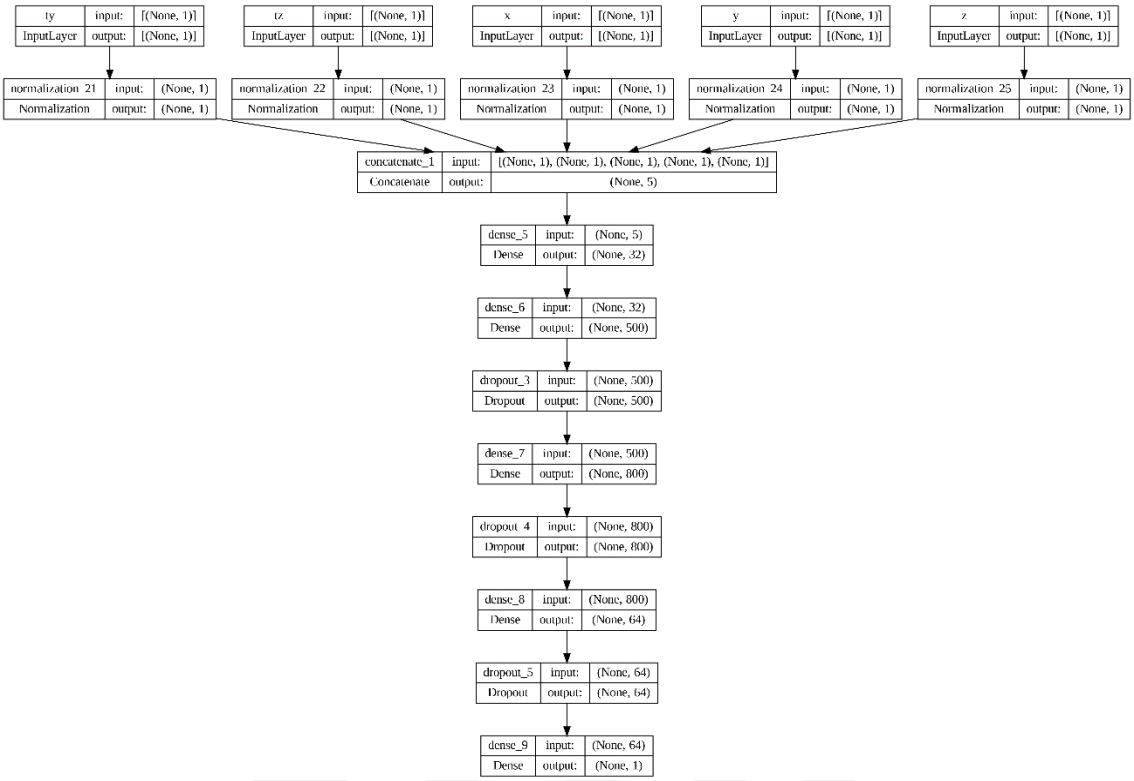
3.4.2.Model Eğitimi

Modelin eğitimi için uygulama geliştirme kolaylığı açısından Google Colab platformu üzerinden Python kullanılmıştır. Eğitim işlemi; TF, Keras, Numpy ve Pandas kütüphaneleri kullanılarak gerçekleştirilmiştir.

Hibrid uygulama sonra oluşturulan veri kümesi kullanılarak YSA modelinin eğitimi için Python dili, Tensorflow, Keras, Numpy ve Pandas kütüphanesi kullanılmıştır. Eğitim için öncelikle veri kümesi ortama yüklenir. Veri kümeleri toplamda 10x10 harita için 11060, 25x25 harita için 6115, , 50x50 harita için 6443, adet örnekten oluşmaktadır. Oluşturulan örnek sayılarının, tüm harita ortamlarında yol oluşturma iterasyon sayısının aynı olmasına rağmen farklı sayıda oluşması haritaların büyüklüğünün farklı olması ve daha küçük harita boyutları için hibrid dataset oluşturma algoritmasının daha fazla yol oluşturmasıdır. Yöntemde verilerin %80'i eğitim, %20'side test için ayrılmıştır. Bu veri çerçeveleri uygun veri kümeleri haline dönüştürülerek 8'erli gruplara ayrılmıştır. Bu girdilerin tamamı eğitimin başarımını artırmak için normalize edilmiştir ve giriş katmanına uygulanacak biçimde bir dizi haline getirilmiştir. Eğitilen model 4 katmadan meydana gelen ve 500 döngü boyunca eğitimi gerçekleştirilmiş ileri beslemeli yapay sinir ağıdır. Çıkış katmanı hariç aktivasyon fonksiyonu olarak bu modelde daha iyi sonuçlar verdiğinden ReLU kullanılmış ve her bir katmanda öğrenmeyi tamamlayamadığı için atılma oranı 0,5 olarak seçilmiştir. Çıkış katmanında ise sigmoid aktivasyon fonksiyonu kullanılmıştır. Yoğunluk; ilk katmanda 32, ikinci katmanda 500, üçüncü katmanda 800, dördündü katmanda 64, çıkış katmanında ise 1 olarak belirlenmiştir. Modelin derlemesi sırasında ağırlıklandırma için az bellek gereksinimi bulunan, hesaplama açısından verimli

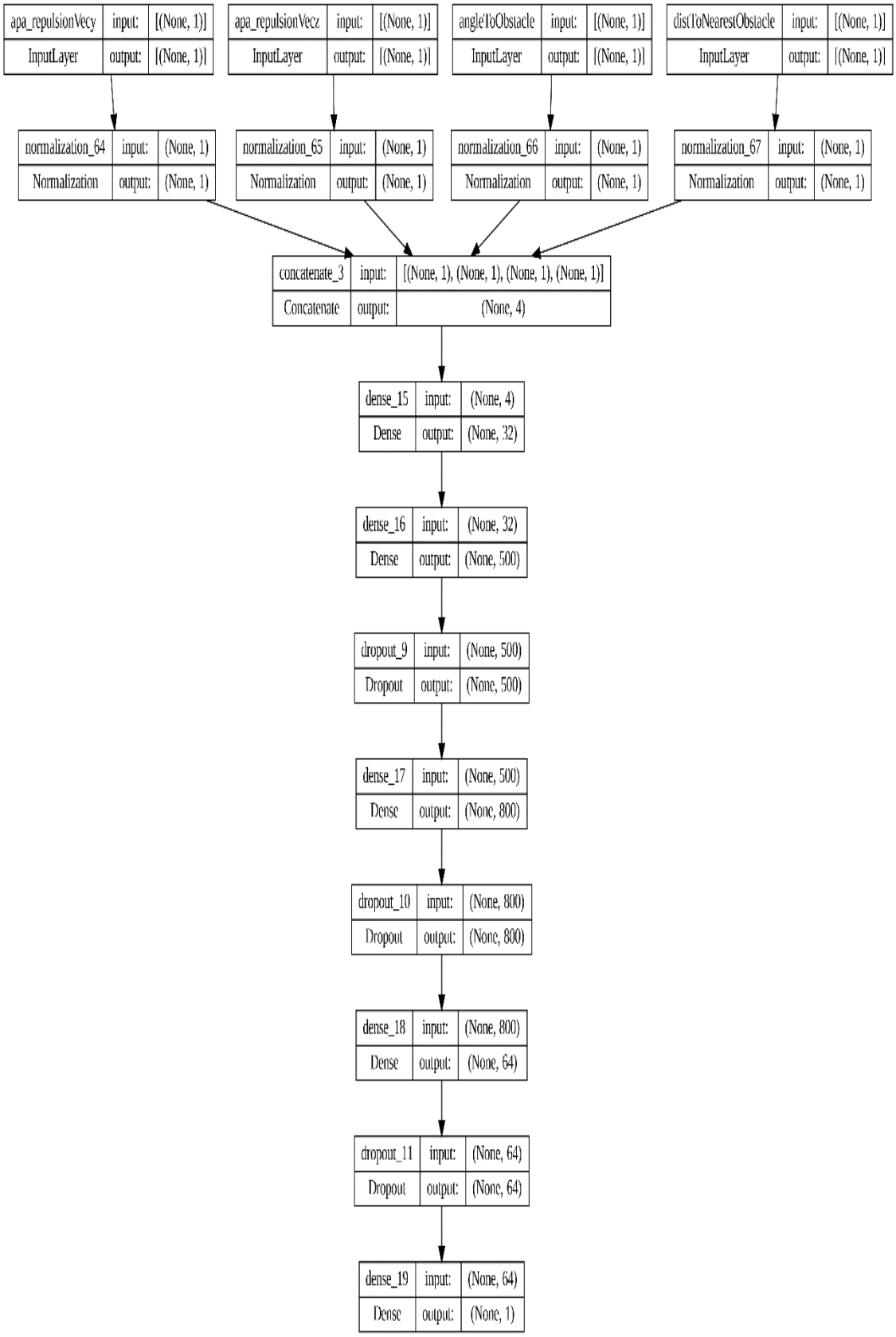
olan, gradyanların ve kare gradyanların üstel hareketli ortalamasını hesaplayan adam (Kingma & Ba, 2014) algoritması kullanılmıştır. Hata fonksiyonu olarak da MSE kullanılmıştır. Modelin iyileşmesini sağlamak amacıyla doğrulama kaybı monitörize edilmiş ve en iyileşen eğitimlerin sonuçları saklanmıştır. Modelin eğitimi 500 devir boyunca yapılmıştır. Eğitilen ağ modeli giriş değişkenlerinin fazla sayıda olması sebebiyle parça parça Şekil 3.8, Şekil 3.9, Şekil 3.10 ve Şekil 3.11'de gösterilmiştir.

Şekil 3.8. Eğitilen ağın modeli – 1



Şekil 3.9. Eğitilen ağın modeli – 2

Şekil 3.10. Eğitilen ağın modeli – 3

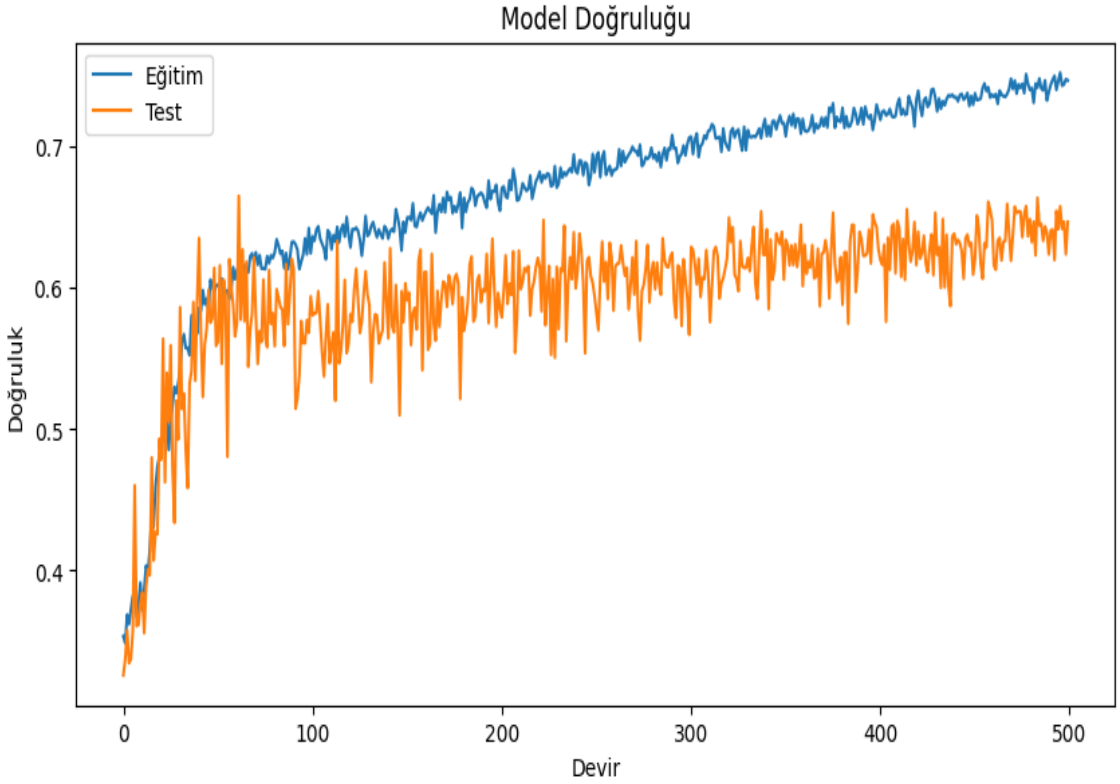


Şekil 3.11. Eğitilen ağın modeli – 4

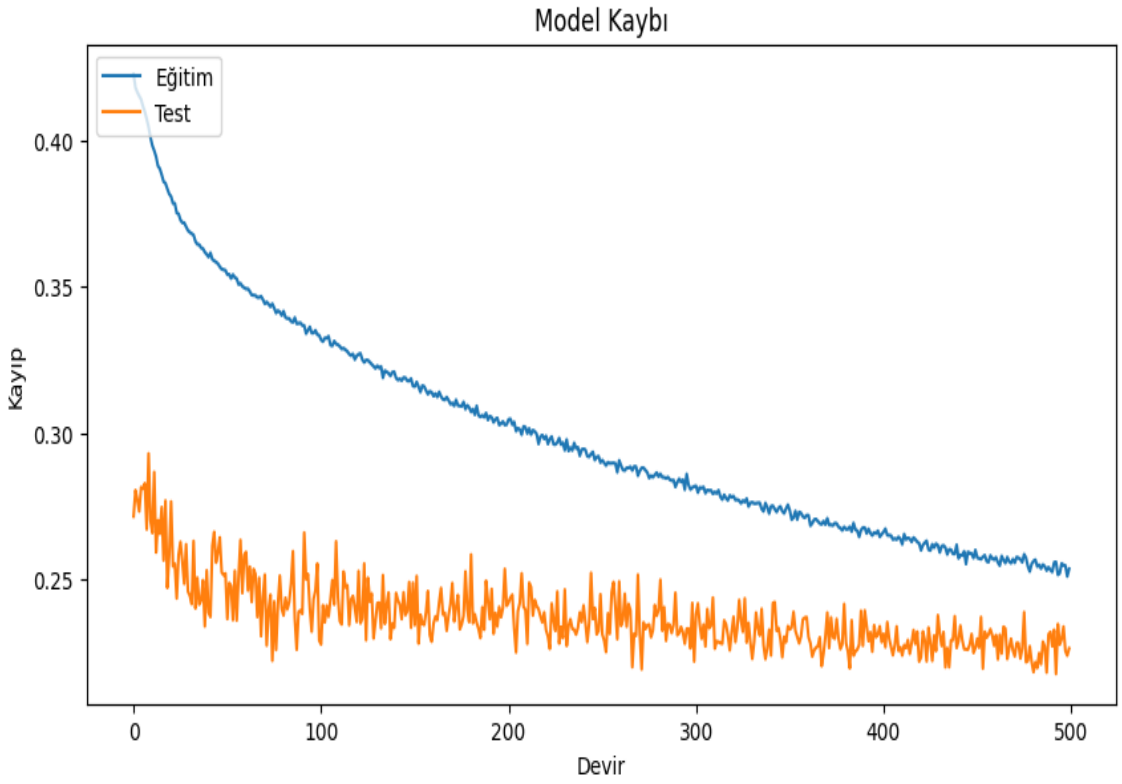
Tablo 3.2. Eğitilmiş önerilen modelin farklı metrikler altında başarım değerleri

Harita Büyüklüğü [m]	K-Fold	Ortalama Doğruluk Değeri	Ortalama Geçerleme Doğruluk Değeri	Ortalama Kayıp Değeri	Ortalama Geçerleme Kayıp Değeri	Ortalama F1 Skor Değeri	Ortalama Geçerleme F1 Skor Değeri	Ortalama Devir Süresi
Harita-1 10 x 10	K-Fold = 5	0.6641	0.6155	0.3002	0.2372	0.5554	0.4663	1.1149
	K-Fold = 10	0.6684	0.6241	0.3008	0.2342	0.5557	0.4761	1.2327
	K-Fold = 20	0.6674	0.6311	0.3026	0.2314	0.5545	0.4770	2.2377
Harita-2 25 x 25	K-Fold = 5	0.7129	0.6393	0.2419	0.2335	0.6817	0.5937	0.6720
	K-Fold = 10	0.7120	0.6412	0.2435	0.2314	0.6803	0.5975	0.6574
	K-Fold = 20	0.7055	0.6454	0.2457	0.2267	0.6764	0.6068	2.7814
Harita-3 50 x 50	K-Fold = 5	0.7224	0.6677	0.2320	0.2173	0.7036	0.6382	0.6910
	K-Fold = 10	0.7210	0.6661	0.2338	0.2172	0.7030	0.6417	0.7988
	K-Fold = 20	0.7204	0.6627	0.2341	0.2177	0.7027	0.6378	0.8112

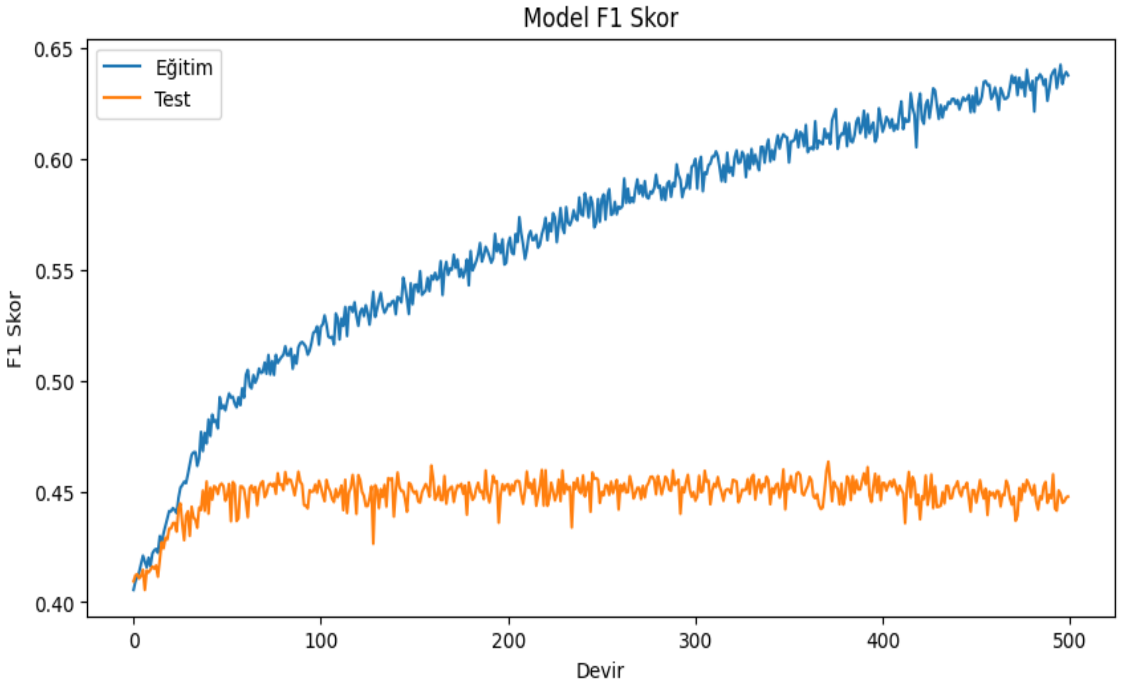
Modelin eğitimi sırasındaki 10 x 10 haritaya ait K-fold = 5 için eğitim ve test doğruluk değerleri Şekil 3.12'deki grafikte, eğitim ve test kayıp değerleri Şekil 3.13'deki grafikte, eğitim ve test f1 skor değerleri Şekil 3.14'deki grafikte gösterilmektedir.



Şekil 3.12. Eğitilen modelin K-fold = 5 için doğruluk grafiği (10 x10)

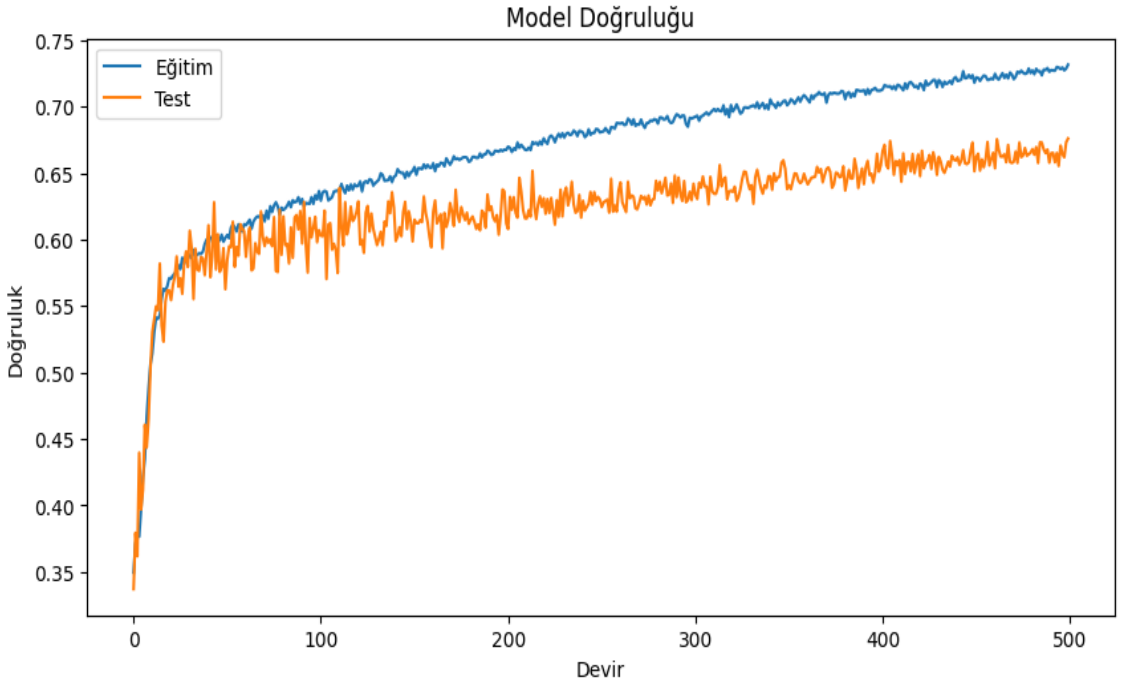


Şekil 3.13. Eğitilen modelin K-fold = 5 için kayıp grafiği (10 x10)

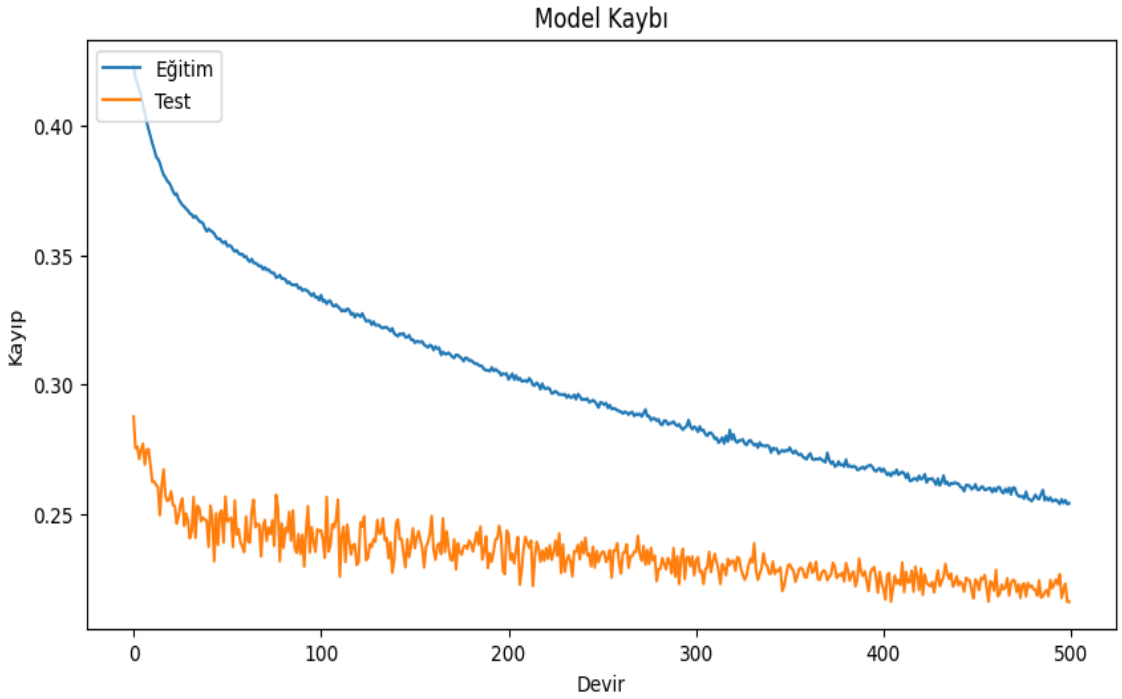


Şekil 3.14. Eğitilen modelin K-fold = 5 için f1 skor grafiği (10 x 10)

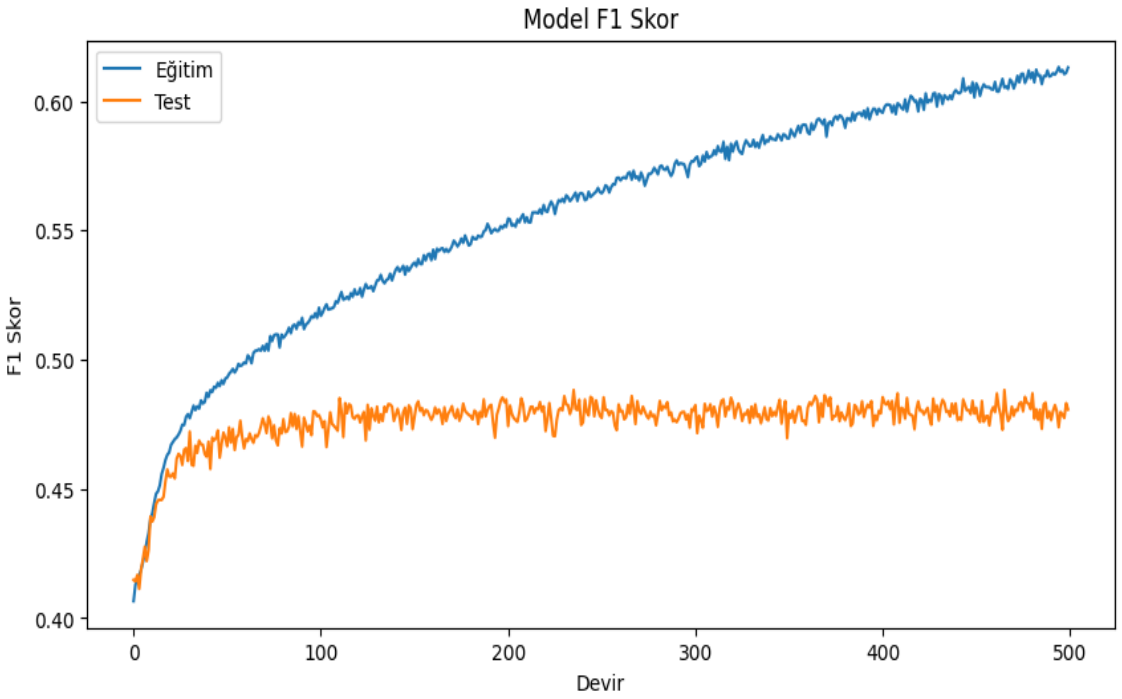
Modelin eğitimi sırasındaki 10 x 10 haritaya ait K-fold = 10 için eğitim ve test doğruluk değerleri Şekil 3.15'deki grafikte, eğitim ve test kayıp değerleri Şekil 3.16'deki grafikte, eğitim ve test f1 skor değerleri Şekil 3.17'deki grafikte gösterilmektedir.



Şekil 3.15. Eğitilen modelin K-fold = 10 için doğruluk grafiği (10 x10)

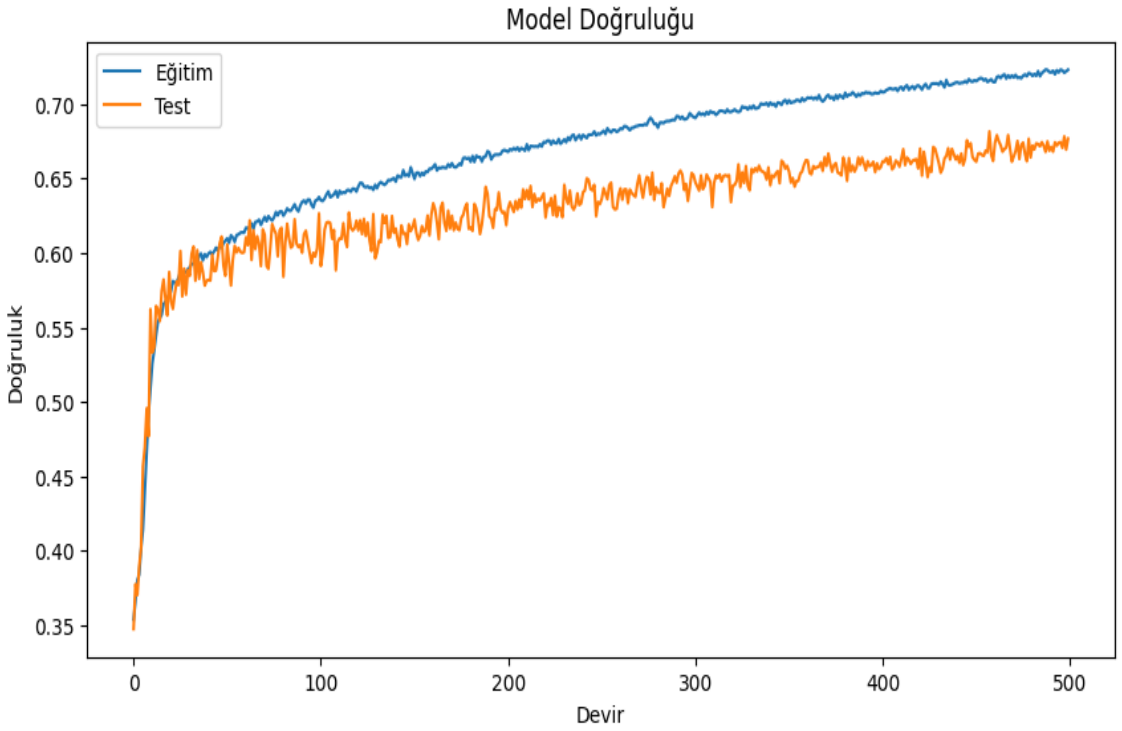


Şekil 3.16. Eğitilen modelin K-fold = 10 için kayıp grafiği (10 x10)

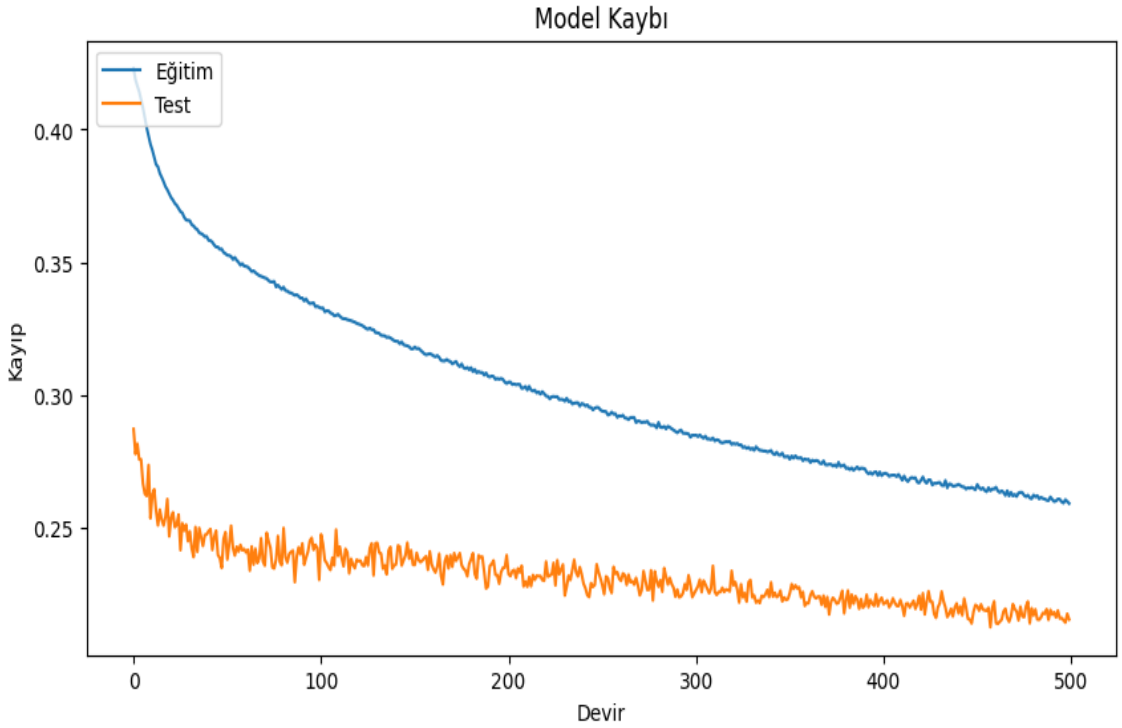


Şekil 3.17. Eğitilen modelin K-fold = 10 için f1 skor grafiği (10 x 10)

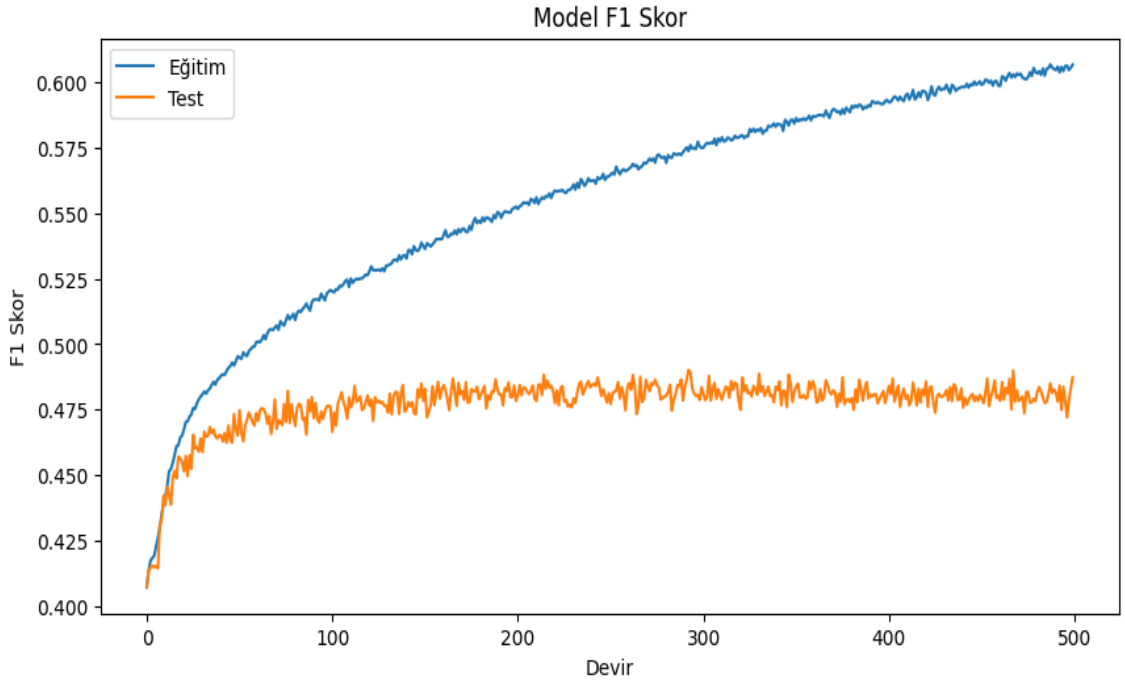
Modelin eğitimi sırasındaki 10 x 10 haritaya ait K-fold = 20 için eğitim ve test doğruluk değerleri Şekil 3.18'deki grafikte, eğitim ve test kayıp değerleri Şekil 3.19'daki grafikte, eğitim ve test f1 skor değerleri Şekil 3.20'deki grafikte gösterilmektedir.



Şekil 3.18. Eğitilen modelin K-fold = 20 için doğruluk grafiği (10 x10)

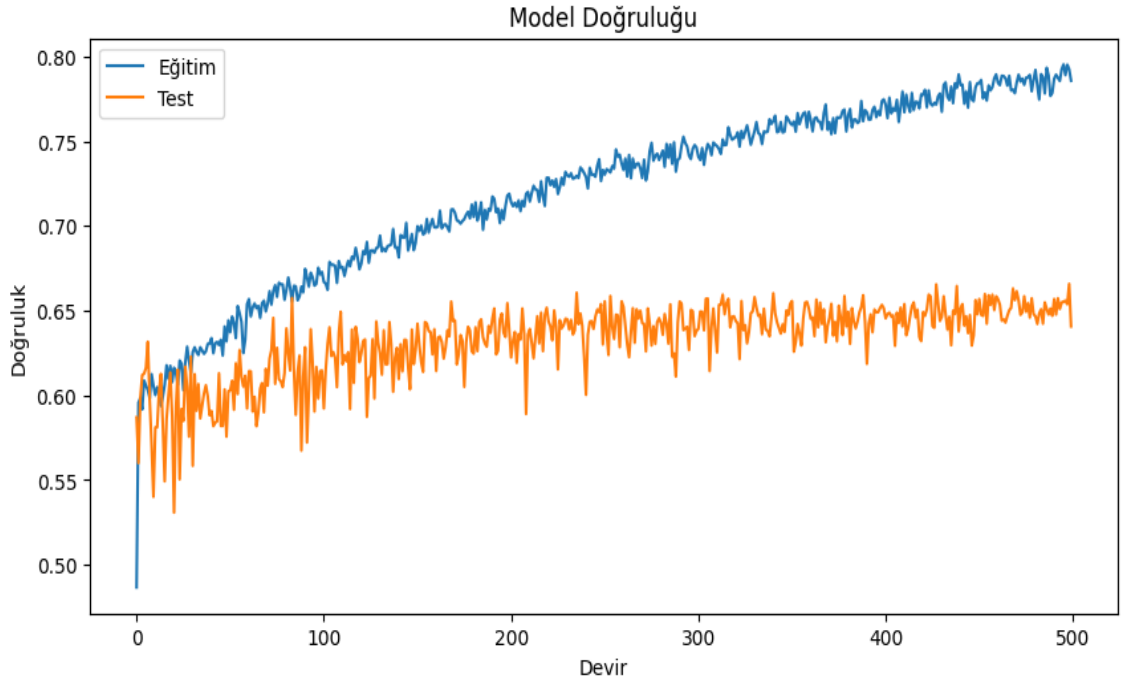


Şekil 3.19. Eğitilen modelin K-fold = 20 için kayıp grafiği (10 x10)

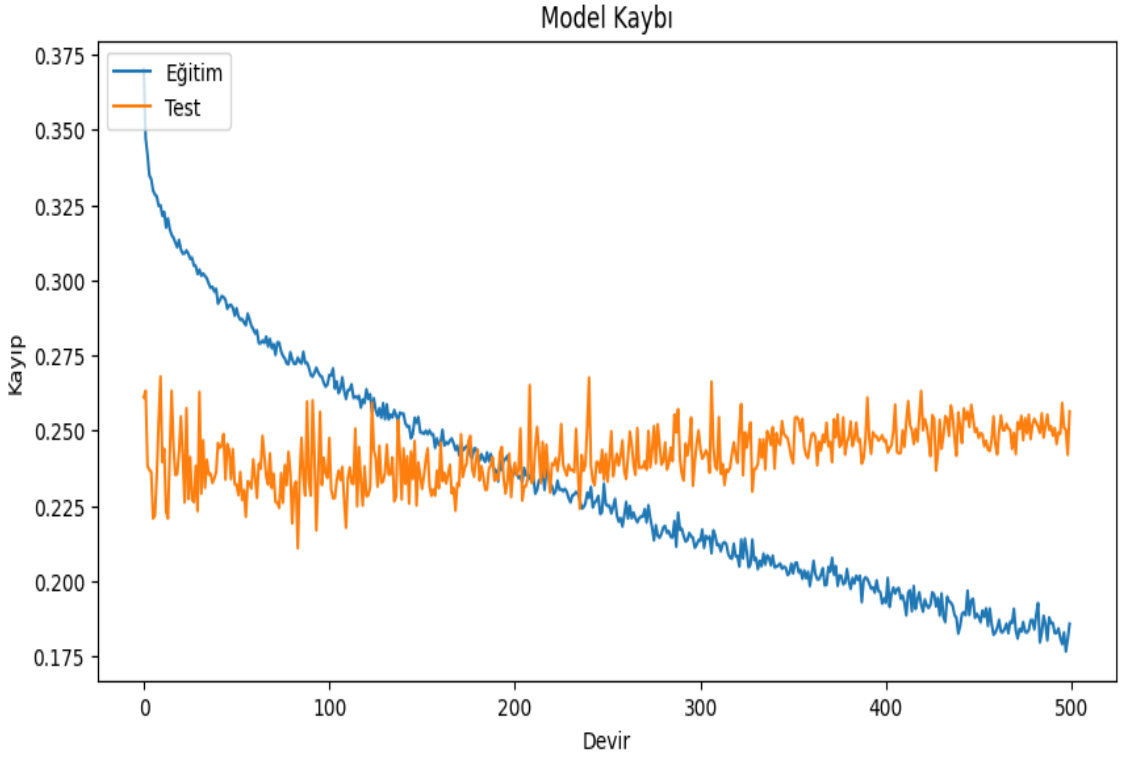


Şekil 3.20. Eğitilen modelin K-fold = 20 için f1 skor grafiği (10 x 10)

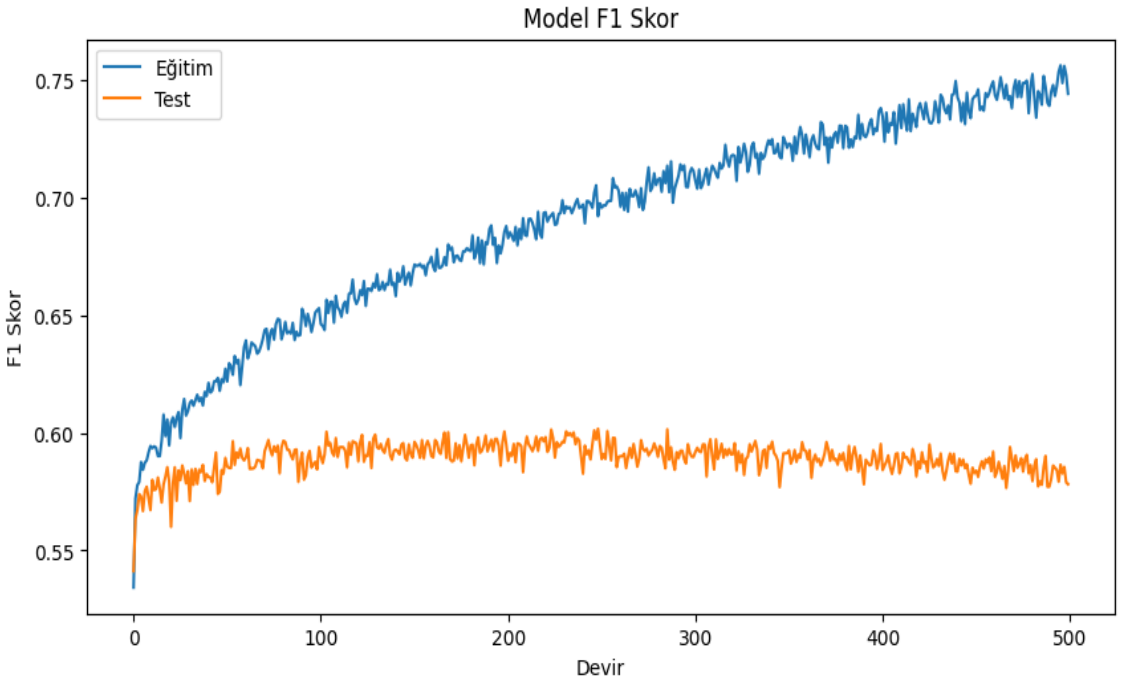
Modelin eğitimi sırasındaki 25 x 25 haritaya ait K-fold = 5 için eğitim ve test doğruluk değerleri Şekil 3.21'deki grafikte, eğitim ve test kayıp değerleri Şekil 3.22'deki grafikte, eğitim ve test f1 skor değerleri Şekil 3.23'deki grafikte gösterilmektedir.



Şekil 3.21. Eğitilen modelin K-fold = 5 doğruluk grafiği (25 x 25)

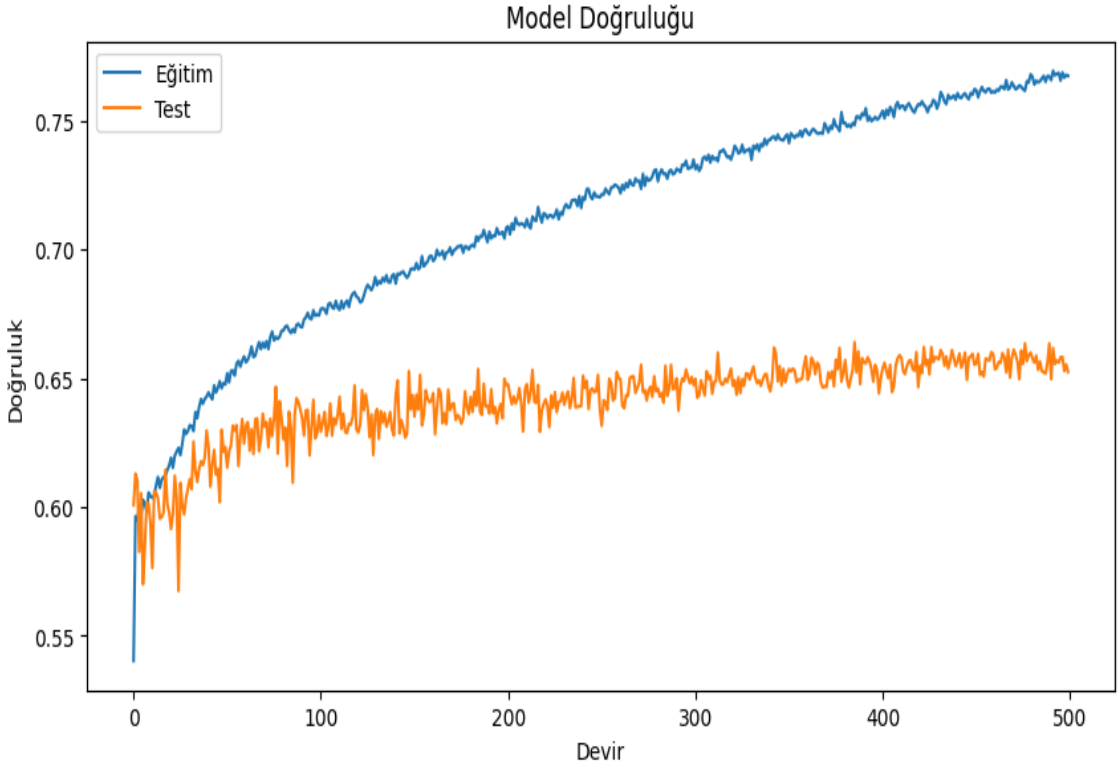


Şekil 3.22. Eğitilen modelin K-fold = 5 kayıp grafiği (25 x 25)

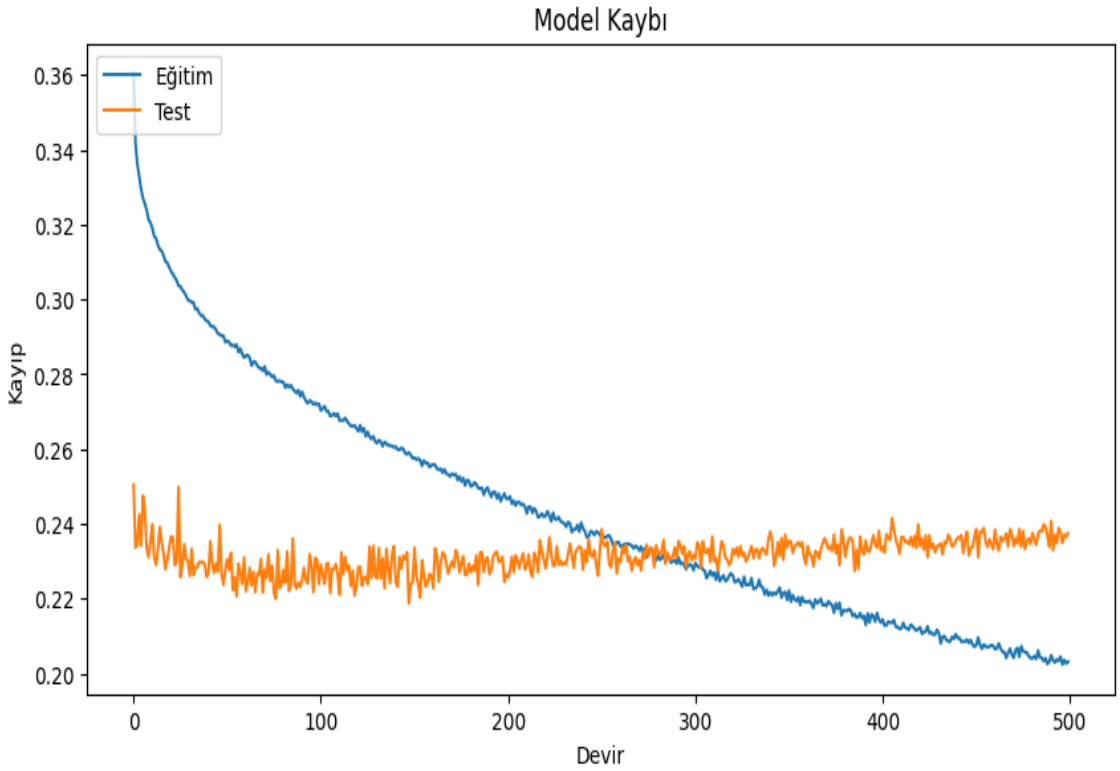


Şekil 3.23. Eğitilen modelin K-fold = 5 için f1 skor grafiği (25 x 25)

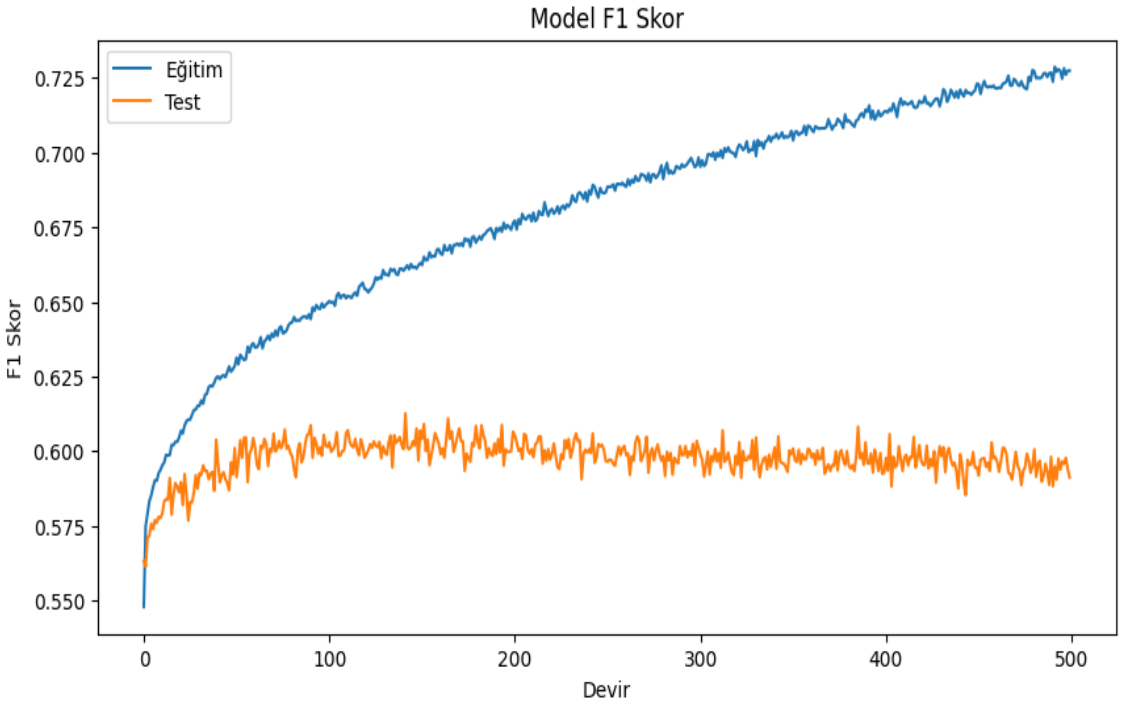
Modelin eğitimi sırasındaki 25 x 25 haritaya ait K-fold = 10 için eğitim ve test doğruluk değerleri Şekil 3.24'deki grafikte, eğitim ve test kayıp değerleri Şekil 3.25'deki grafikte, eğitim ve test f1 skor değerleri Şekil 3.26'daki grafikte gösterilmektedir.



Şekil 3.24. Eğitilen modelin K-fold = 10 doğruluk grafiği (25 x 25)

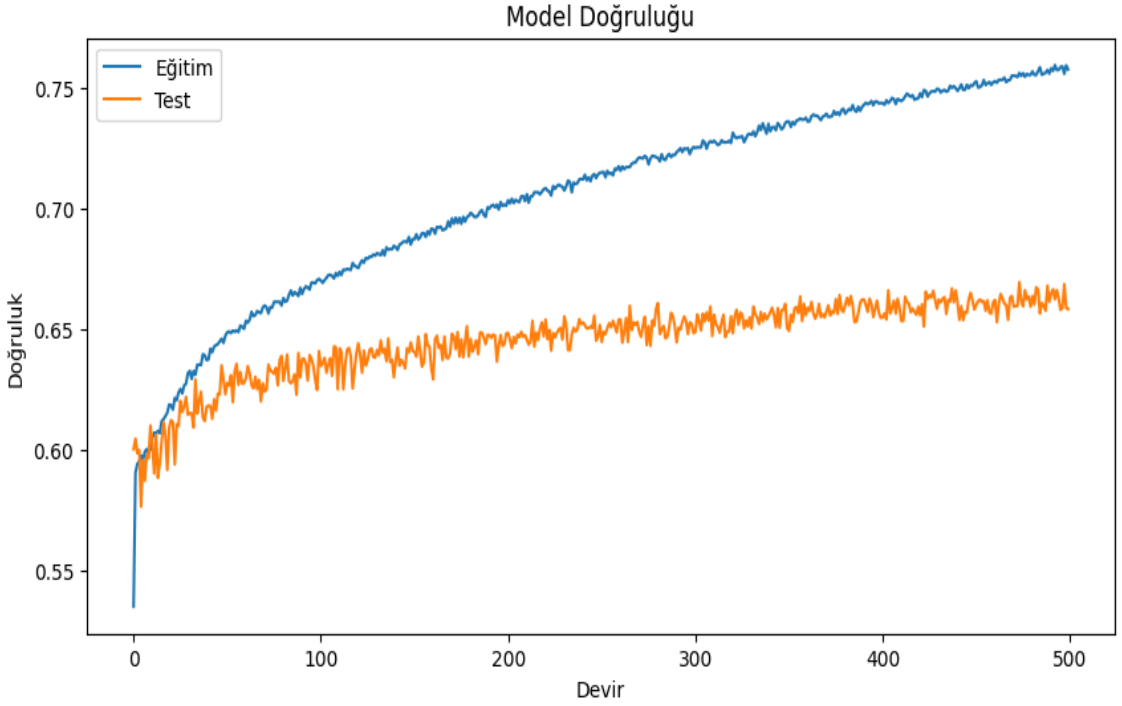


Şekil 3.25. Eğitilen modelin K-fold = 10 kayıp grafiği (25 x 25)

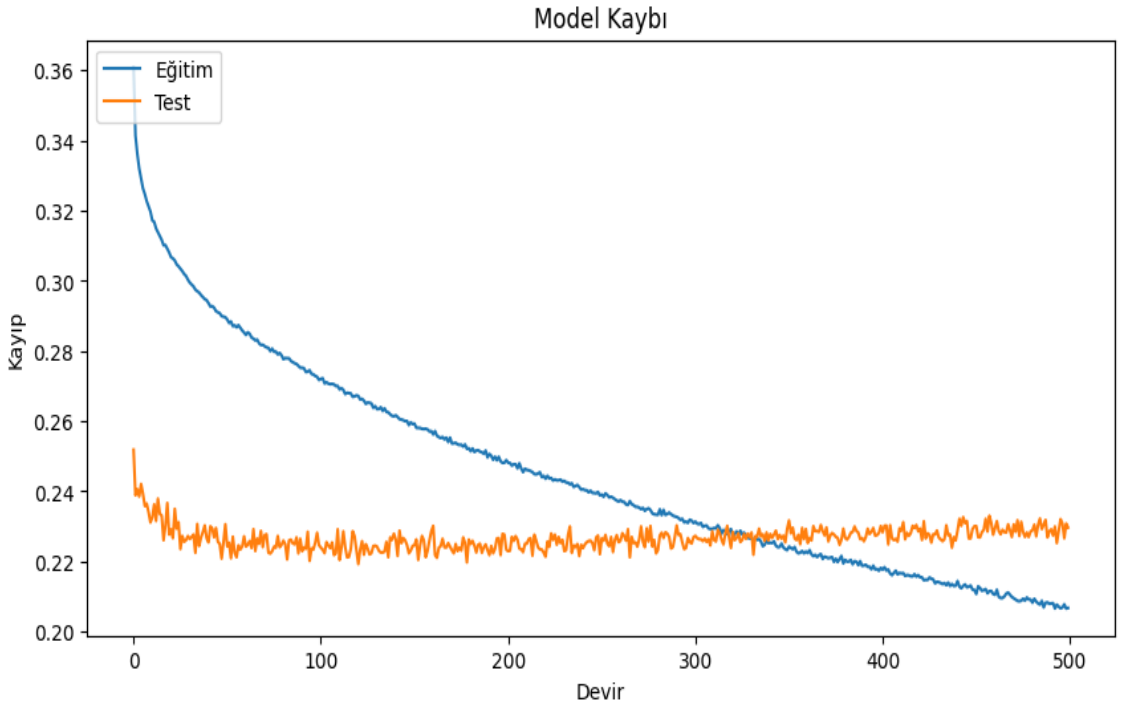


Şekil 3.26. Eğitilen modelin K-fold = 10 için f1 skor grafiği (25 x 25)

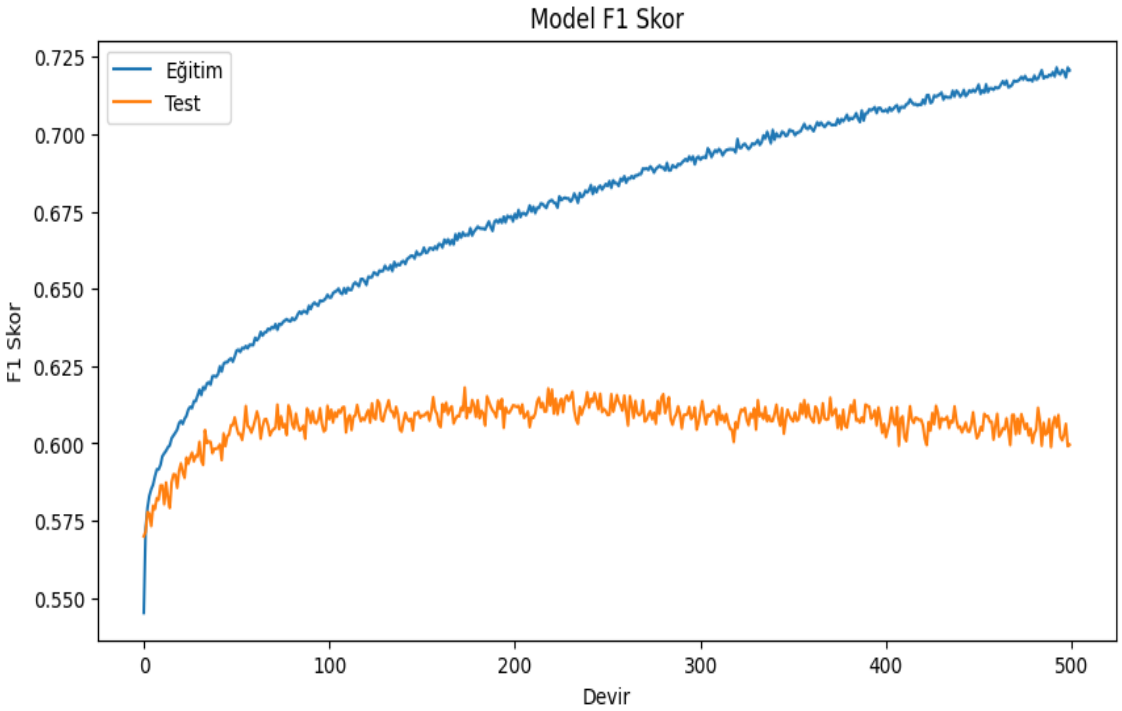
Modelin eğitimi sırasındaki 25 x 25 haritaya ait K-fold = 20 için eğitim ve test doğruluk değerleri Şekil 3.27'deki grafikte, eğitim ve test kayıp değerleri Şekil 3.28'deki grafikte, eğitim ve test f1 skor değerleri Şekil 3.29'daki grafikte gösterilmektedir.



Şekil 3.27. Eğitilen modelin K-fold = 20 doğruluk grafiği (25 x 25)

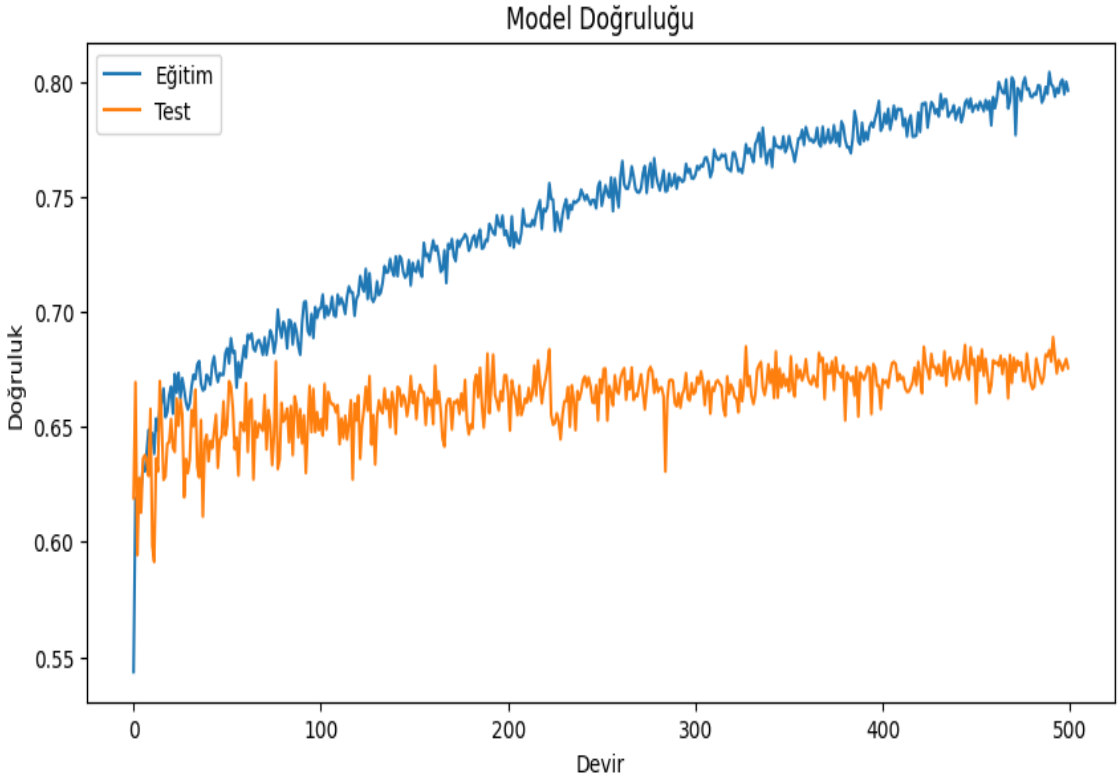


Şekil 3.28. Eğitilen modelin K-fold = 20 kayıp grafiği (25 x 25)

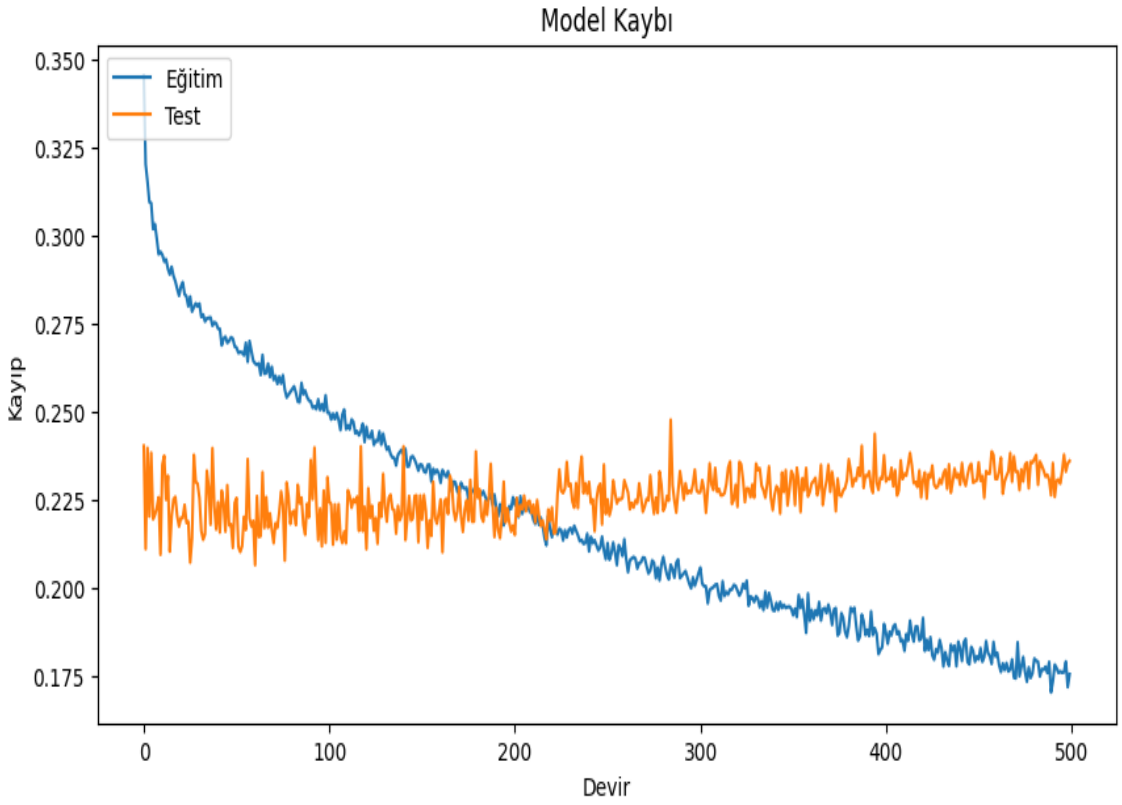


Şekil 3.29. Eğitilen modelin K-fold = 20 için f1 skor grafiği (25 x 25)

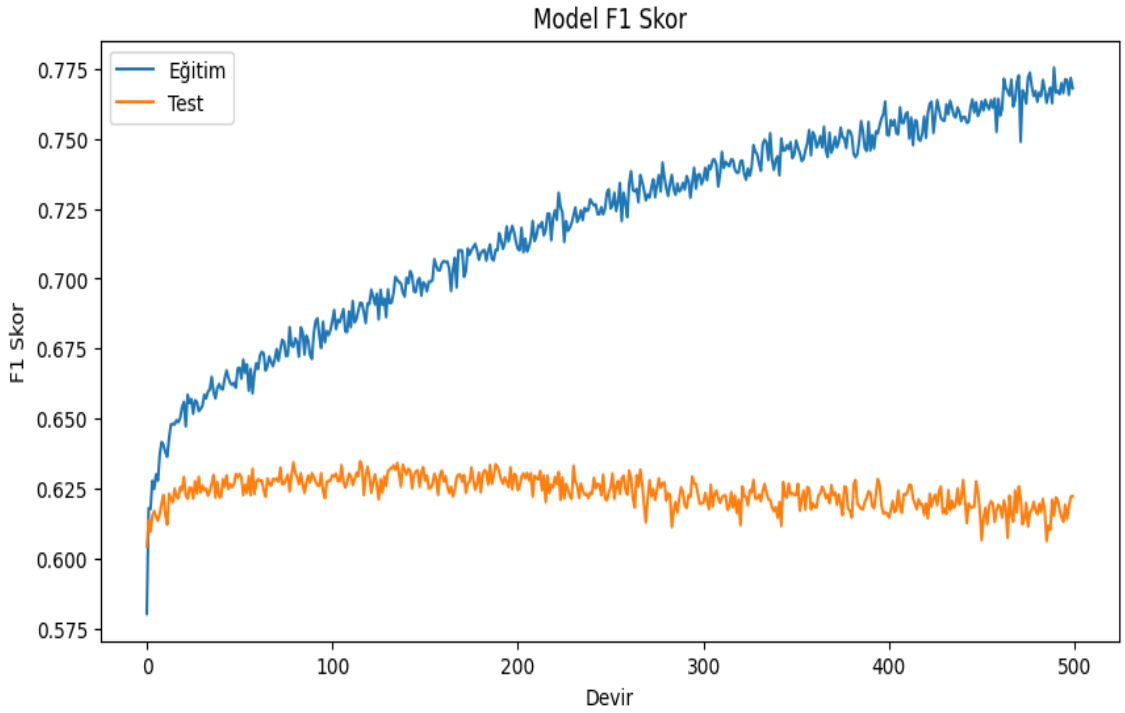
Modelin eğitimi sırasındaki 50 x 50 haritaya ait K-fold = 5 için eğitim ve test doğruluk değerleri Şekil 3.30'daki grafikte, eğitim ve test kayıp değerleri Şekil 3.31'deki grafikte, eğitim ve test f1 skor değerleri Şekil 3.32'deki grafikte gösterilmektedir.



Şekil 3.30. Eğitilen modelin K-fold = 5 doğruluk grafiği (50 x 50)

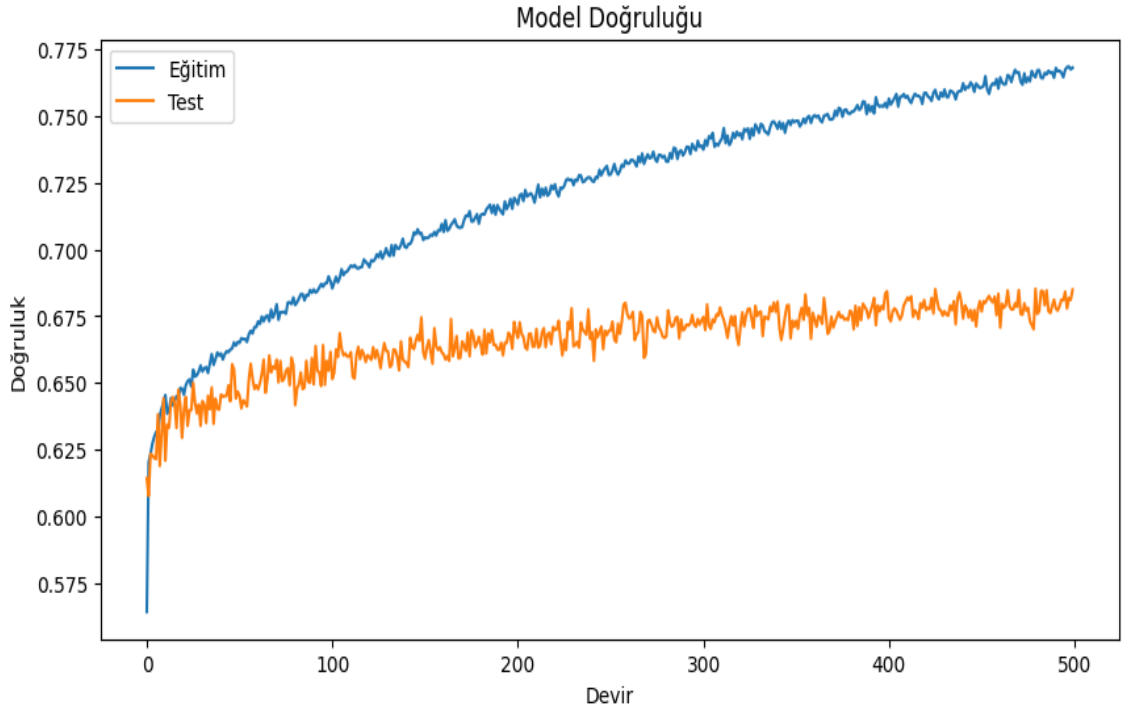


Şekil 3.31. Eğitilen modelin K-fold = 5 için kayıp grafiği (50 x 50)

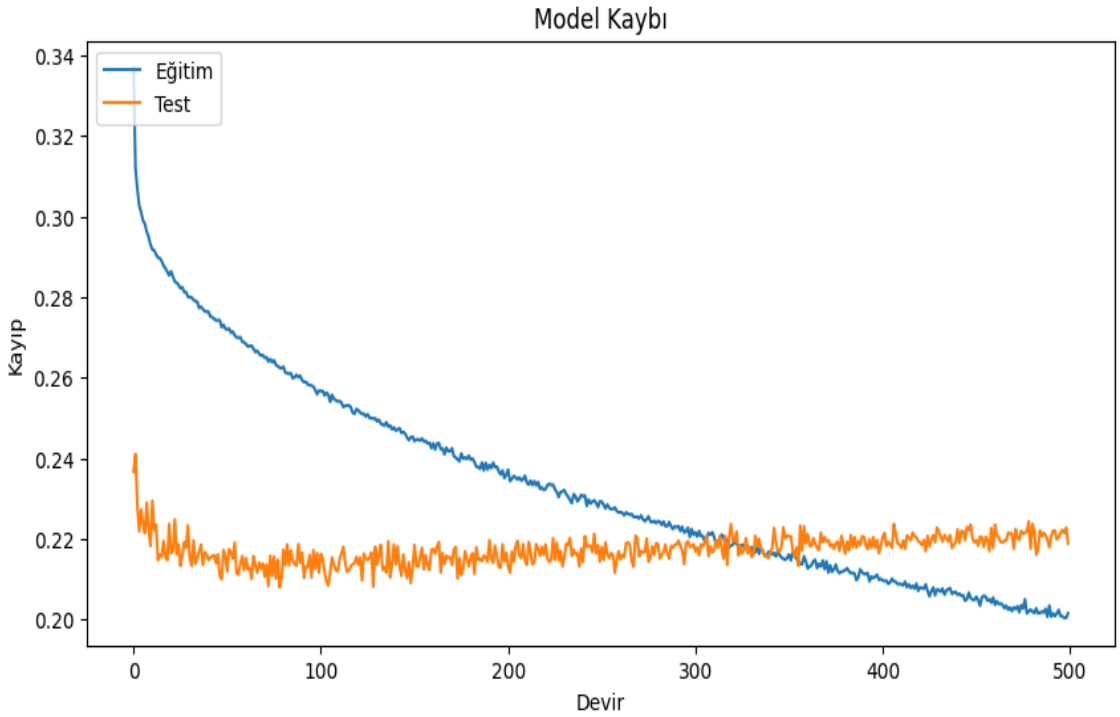


Şekil 3.32. Eğitilen modelin K-fold = 5 için f1 skor grafiği (50 x 50)

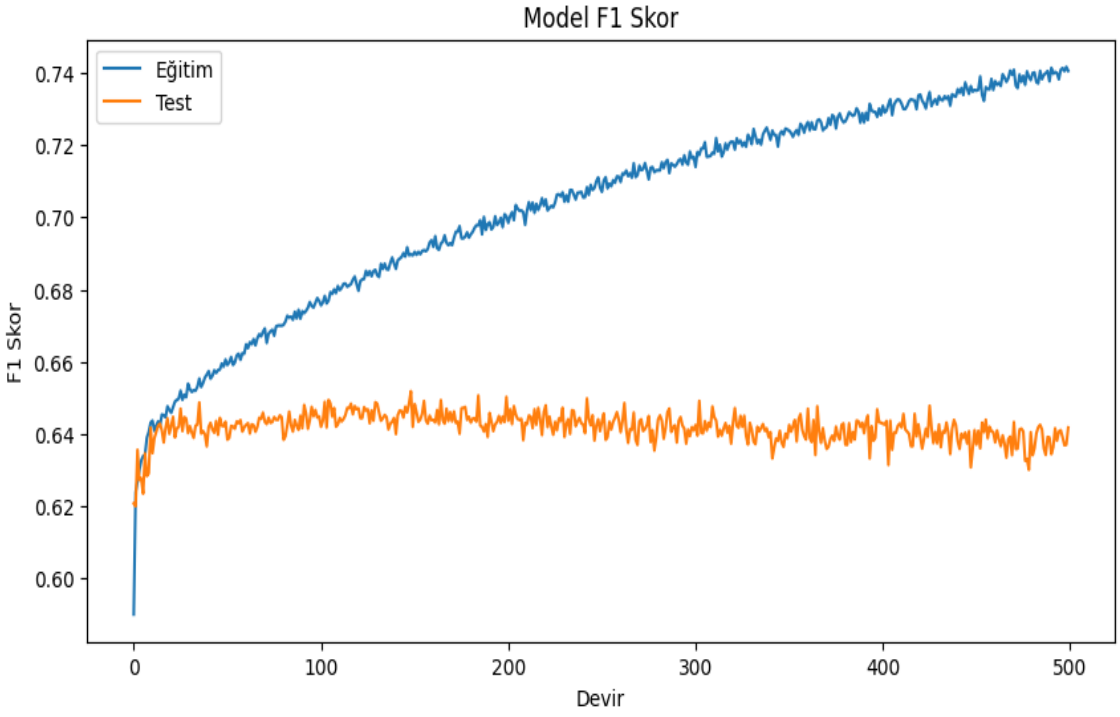
Modelin eğitimi sırasındaki 50 x 50 haritaya ait K-fold = 10 için eğitim ve test doğruluk değerleri Şekil 3.33'deki grafikte, eğitim ve test kayıp değerleri Şekil 3.34'deki grafikte, eğitim ve test f1 skor değerleri Şekil 3.35'deki grafikte gösterilmektedir.



Şekil 3.33. Eğitilen modelin K-fold = 10 doğruluk grafiği (50 x 50)

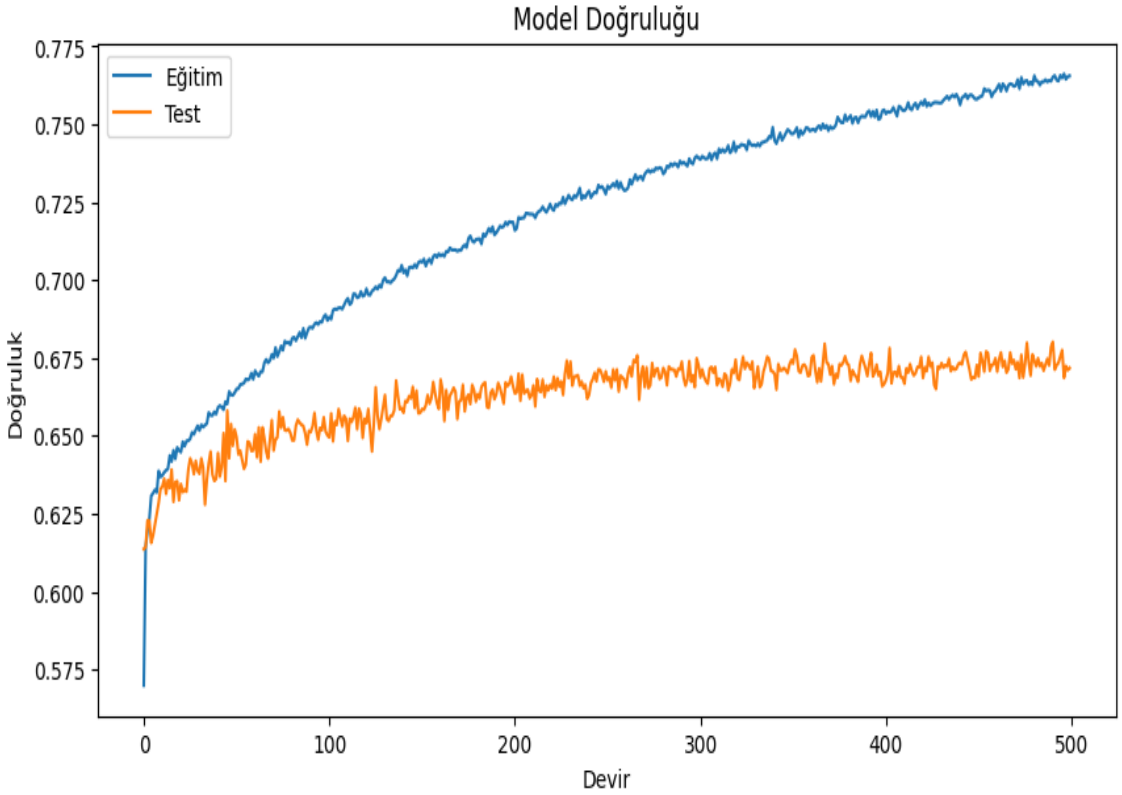


Şekil 3.34. Eğitilen modelin K-fold = 10 kayıp grafiği (50 x 50)

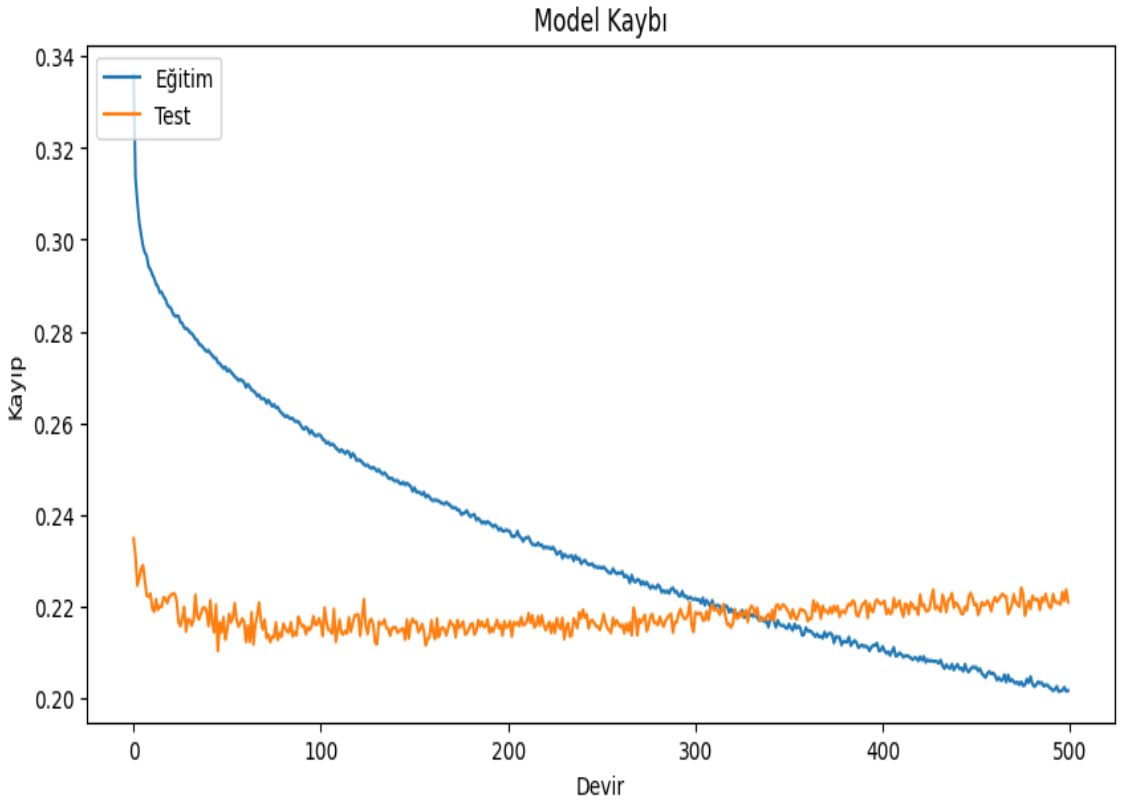


Şekil 3.35. Eğitilen modelin K-fold = 10 için f1 skor grafiği (50 x 50)

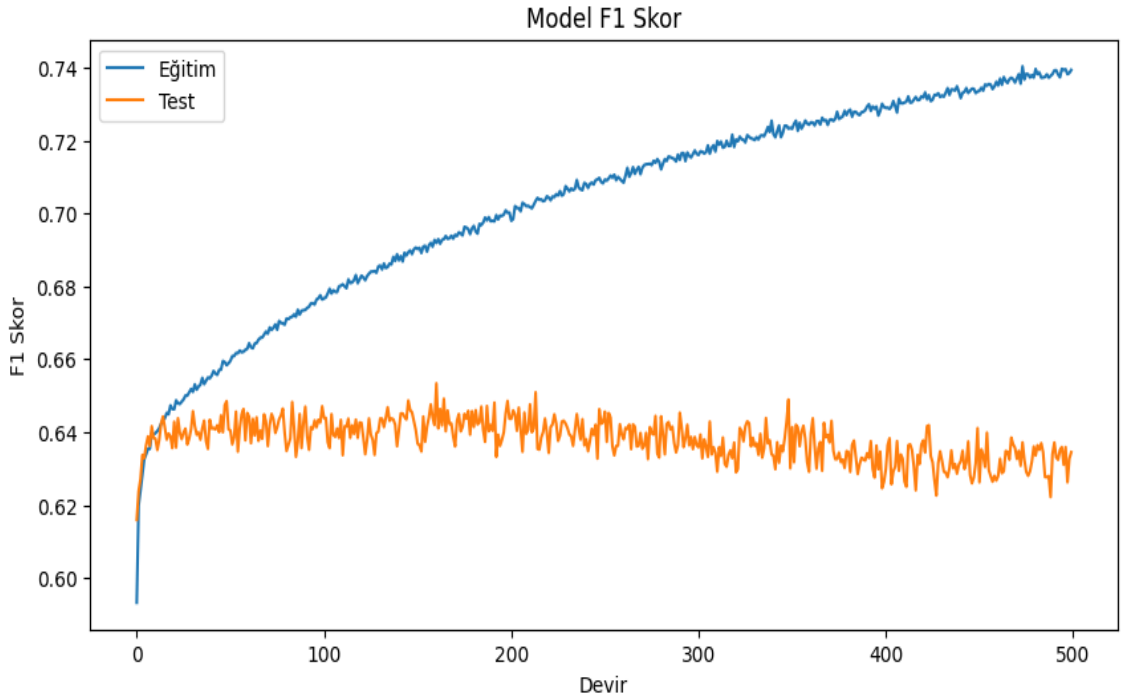
Modelin eğitimi sırasındaki 50 x 50 haritaya ait K-fold = 20 için eğitim ve test doğruluk değerleri Şekil 3.36'daki grafikte, eğitim ve test kayıp değerleri Şekil 3.37'deki grafikte, eğitim ve test f1 skor değerleri Şekil 3.38'deki grafikte gösterilmektedir.



Şekil 3.36. Eğitilen modelin K-fold = 20 doğruluk grafiği (50 x 50)

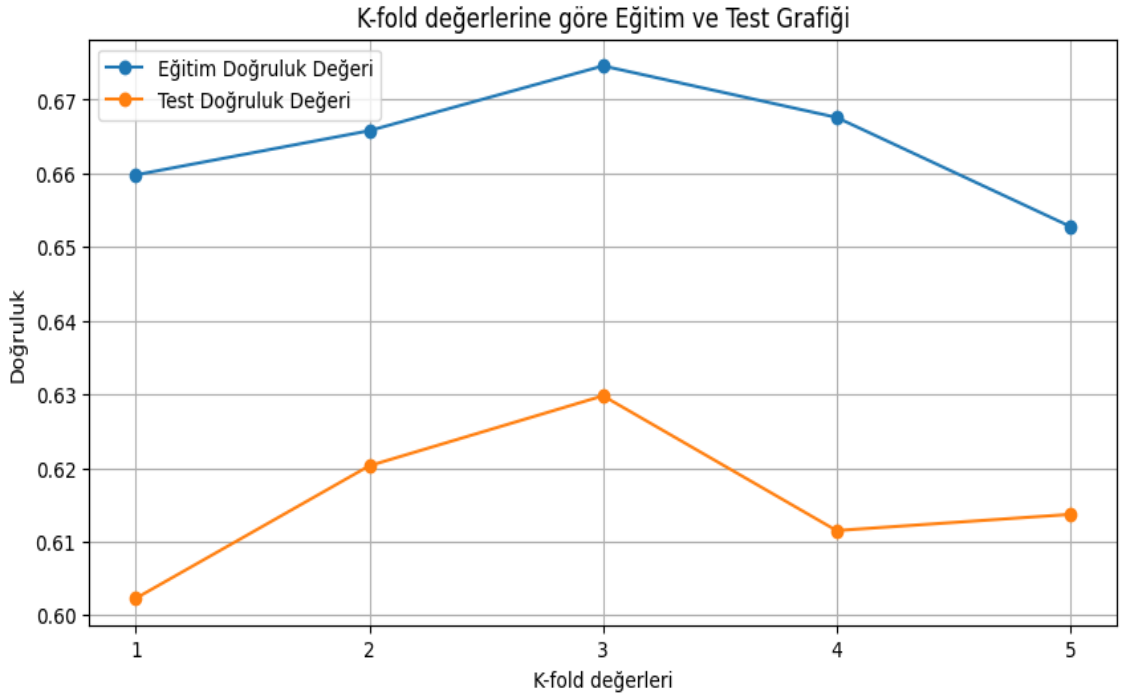


Şekil 3.37. Eğitilen modelin K-fold = 20 kayıp grafiği (50 x 50)

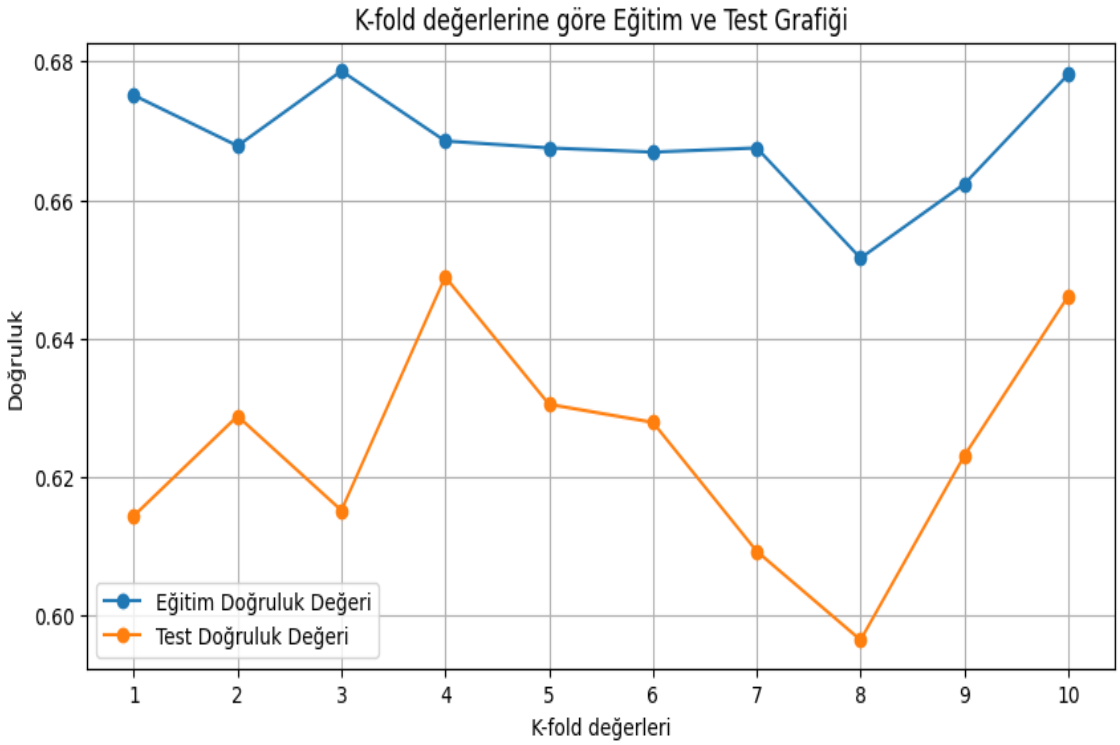


Şekil 3.38. Eğitilen modelin K-fold = 20 için f1 skor grafiği (50 x 50)

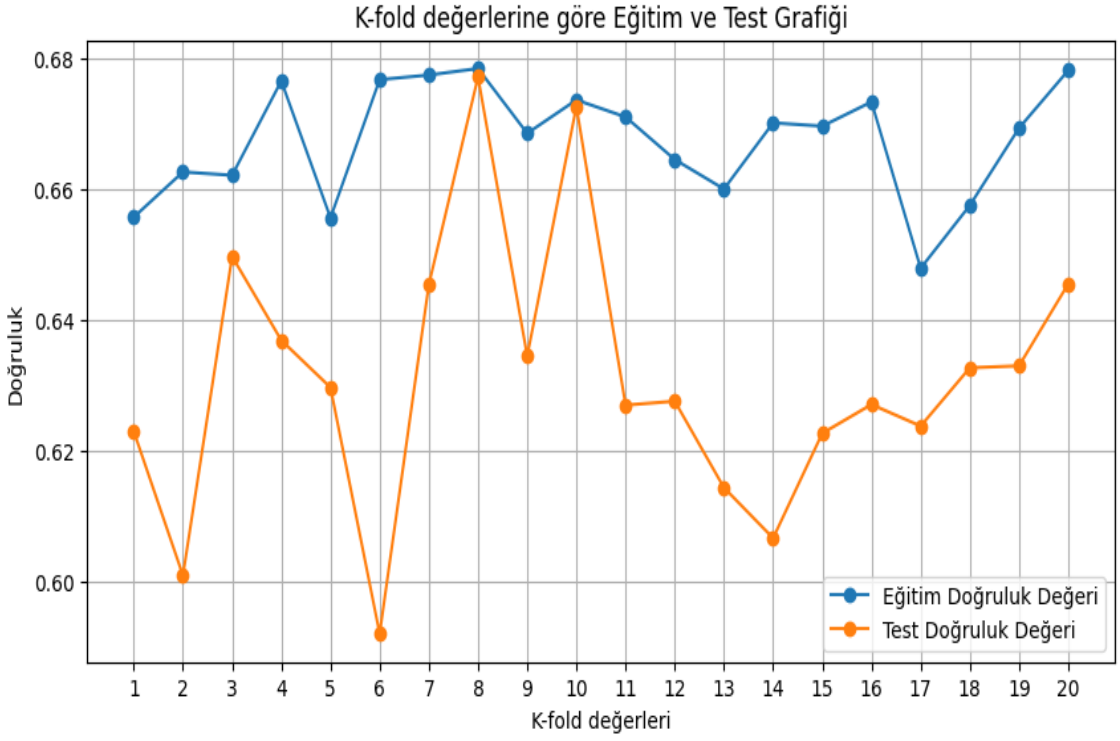
Modelin eğitimi sırasında 10x10 haritaya ait her bir K-fold için eğitim ve test doğruluk grafikleri Şekil 3.39, Şekil 3.40 ve Şekil 3.41'deki grafiklerde gösterilmektedir.



Şekil 3.39. K-fold = 5 için K-Fold eğitim ve test doğruluk dağılım grafiği (10 x 10)

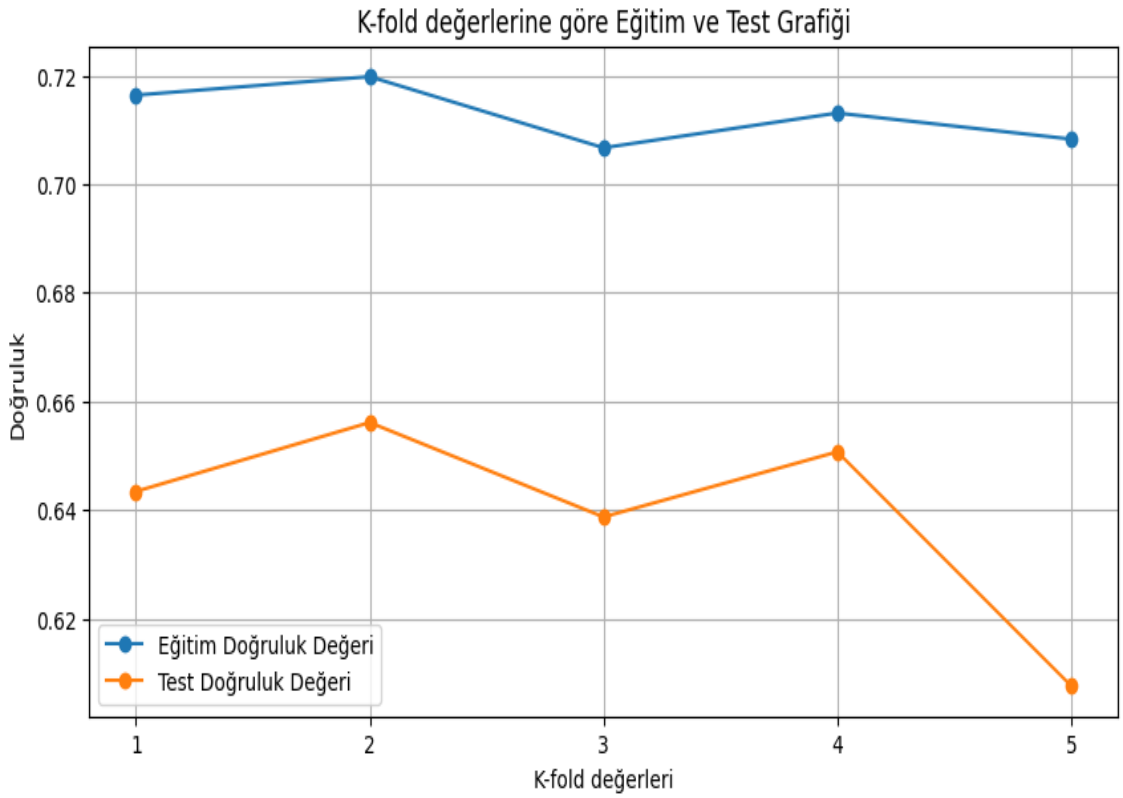


Şekil 3.40. K-fold = 10 için K-Fold eğitim ve test doğruluk dağılım grafiği (10 x 10)

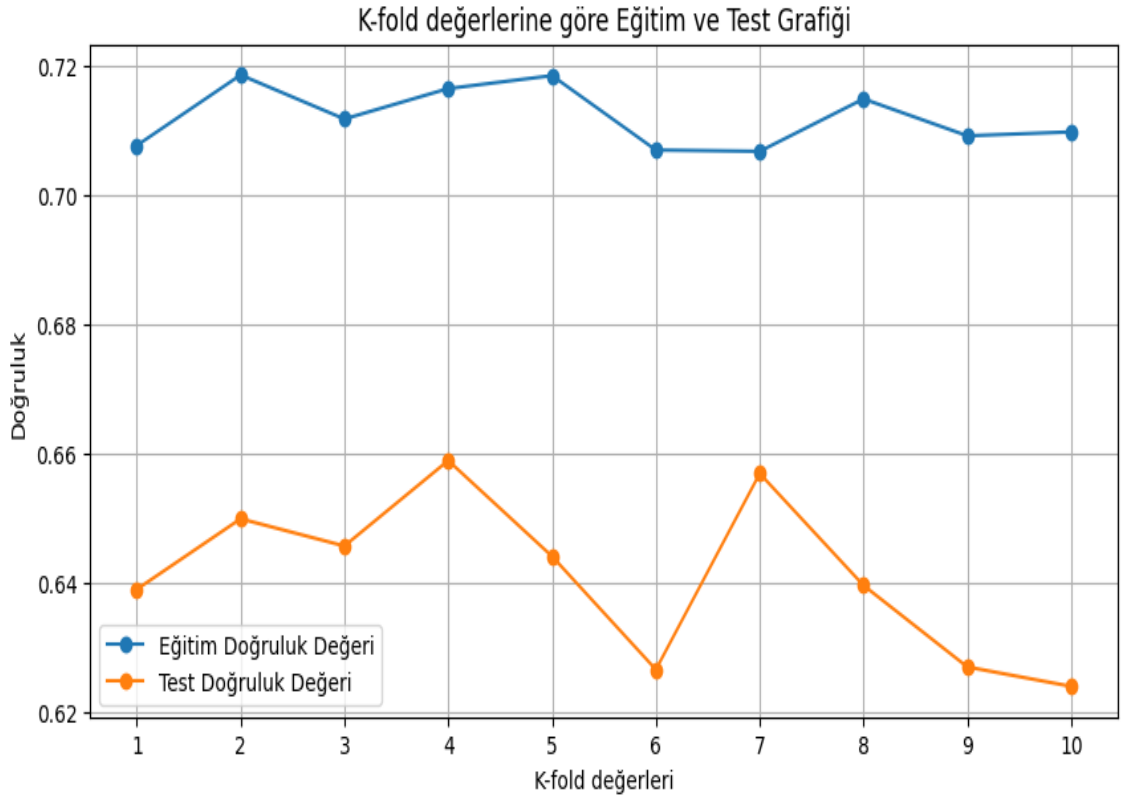


Şekil 3.41. K-fold = 20 için K-Fold eğitim ve test doğruluk dağılım grafiği (10 x 10)

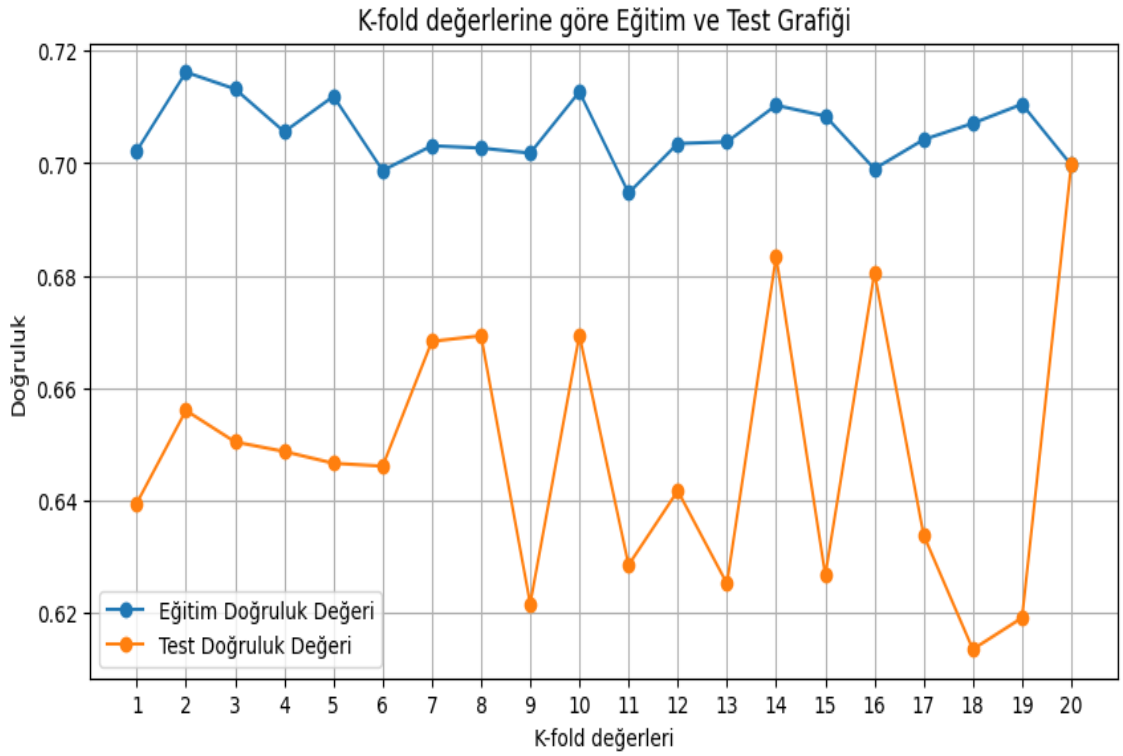
Modelin eğitimi sırasında 25x25 haritaya ait her bir K-fold için eğitim ve test doğruluk grafikleri Şekil 3.42, Şekil 3.43 ve Şekil 3.44'deki grafiklerde gösterilmektedir.



Şekil 3.42. K-fold = 5 için K-Fold eğitim ve test doğruluk dağılım grafiği (25 x 25)

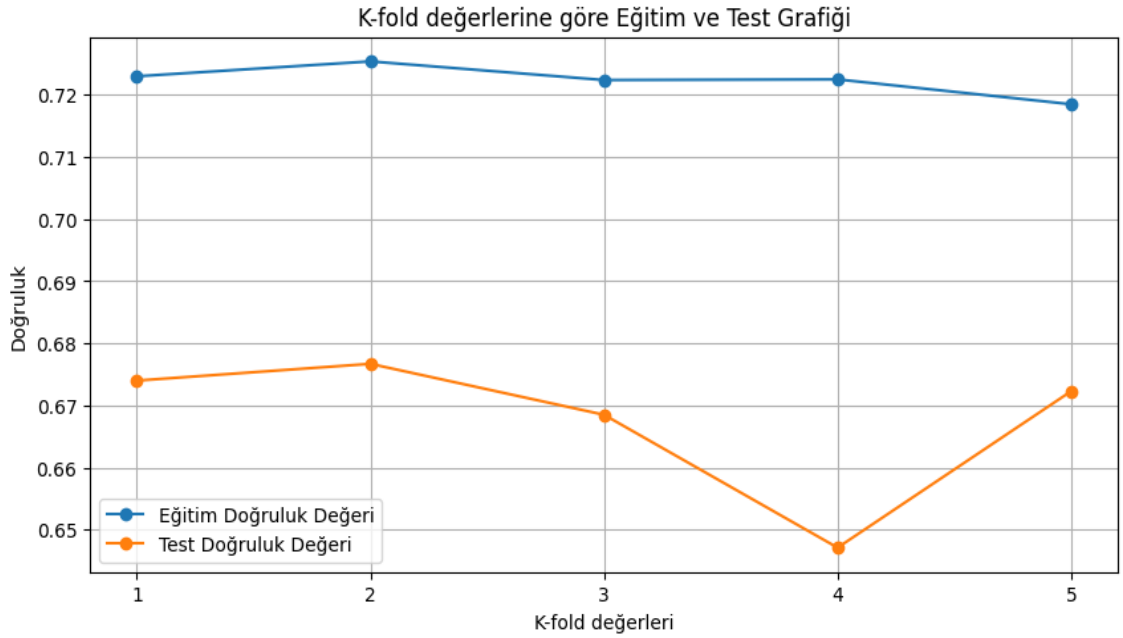


Şekil 3.43. K-fold = 10 için K-Fold eğitim ve test doğruluk dağılım grafiği (25 x 25)

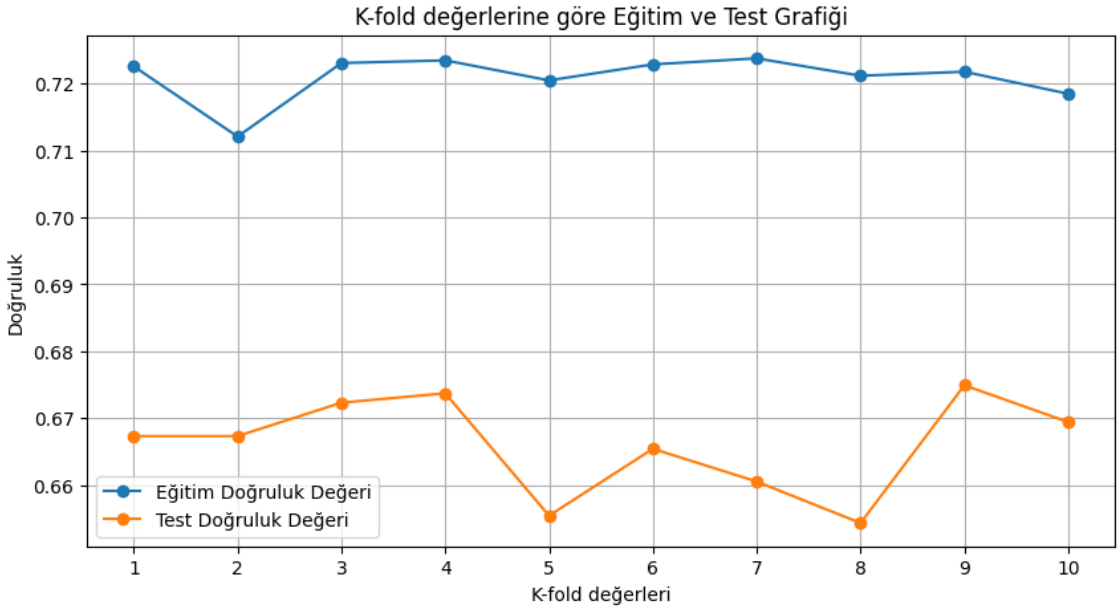


Şekil 3.44. K-fold = 20 için K-Fold eğitim ve test doğruluk dağılım grafiği (25 x 25)

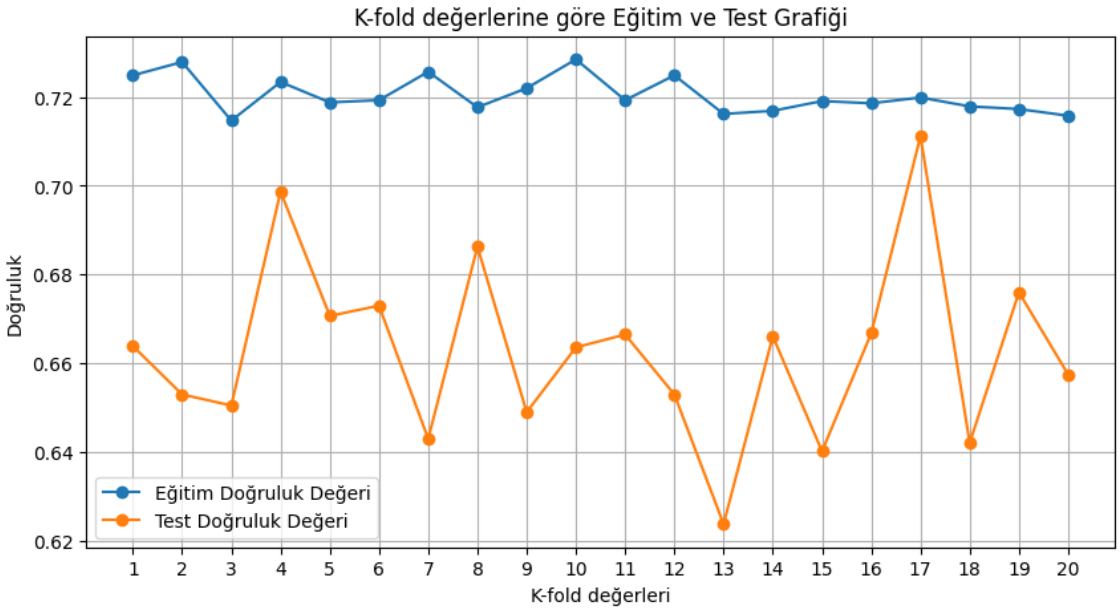
Modelin eğitimi sırasında 50x50 haritaya ait her bir K-fold için eğitim ve test doğruluk grafikleri Şekil 3.45, Şekil 3.46 ve Şekil 3.47'deki grafiklerde gösterilmektedir.



Şekil 3.45. K-fold = 5 için K-Fold eğitim ve test doğruluk dağılım grafiği (50 x 50)



Şekil 3.46. K-fold = 10 için K-Fold eğitim ve test doğruluk dağılım grafiği (50 x 50)



Şekil 3.47. K-fold = 20 için K-Fold eğitim ve test doğruluk dağılım grafiği (50 x 50)

Modelin eğitimi tamamlandıktan sonra simülasyon için geliştirilen uygulamada kullanılabilmesi amacıyla TensorFlow Lite (TFLite) modeline dönüştürülerek “.tflite” uzantılı dosyaya yazılmıştır. Bu dosya geliştirilmiş olan uygulamaya eklenerek model yüklenmiş ve uygulama üzerinde koşulmuştur.

4.ÇALIŞMA KAPSAMINDA KULLANILAN YAZILIMLAR

4.1.İnsansız Hava Aracı Modeli

İnsansız Hava Araçları, (İHA'lar) insan müdahalesi olmadan görev yapabilen, uzaktan kumanda veya önceden belirlenmiş görevlere yönlendirilebilen hava araçlarıdır. Bu araçlar genellikle farklı ölçülerde ve tasarımlarda gelir. Farklı rotor sayısına sahip copter'lar, küçük ve büyük ölçekte insansız uçaklar gibi. İHA'lar, bir dizi uygulamada kullanılır; askeri operasyonlardan, gözetleme ve keşif faaliyetlerine, tarım alanlarında kullanımından, felaket yardımı ve insani yardım görevlerine kadar geniş bir yelpazede hizmet verirler. Bu araçlar, genellikle kameralar, sensörler, GPS ve bazen Lidar gibi çeşitli algılayıcılarla donatılmıştır. Bu algılayıcılar, hava aracının çevresini izlemesine, veri toplamasına ve çevresel değişikliklere tepki göstermesine olanak tanır. İHA'ların çoğu, otonom olarak uçabilen sistemlerle donatılmıştır ve kendi kendine uçma yeteneği, belirlenen görevleri yerine getirmelerini sağlar.

Çalışma kapsamında simülasyon uygulamalarında dört rotorlu Şekil 4.1'de gösterilen 3DR firmasının Iris isimli quadcopter kullanılmıştır.

Şekil 4.1. Iris quadcopter (URL-3)

Quadcopterler 4 rotorlu insansız hava araçları grubuna girer. Quadcopterin hareket ve konumunu belirtmek için 6 Dof serbestlik derecesi tanımı kullanılır. Quadcopter aşağıda belirtilen serbestlik dereceleri için 4 rotorunun ikisi saat yönünde ikiside saat yönünün tersi olacak şekilde hareket eder.

a) İleri-Geri Hareket (X Ekseninde): Nesnenin veya noktanın ileri ve geri yönde hareket etme serbestliği.
b) Yukarı-Aşağı Hareket (Y Ekseninde): Nesnenin veya noktanın yukarı ve aşağı yönde hareket etme serbestliği.
c) Sağa-Sola Hareket (Z Ekseninde): Nesnenin veya noktanın sağa ve sola yönde hareket etme serbestliği.
d) Piksel Dönüş (Yaw): Nesnenin veya noktanın etrafında dikey eksene göre dönme serbestliği.
e) Yalpa Dönüş (Pitch): Nesnenin veya noktanın etrafında yatay eksene göre dönme serbestliği.
f) Dönel Dönüş (Roll): Nesnenin veya noktanın kendi ekseni etrafında dönme serbestliği.

Özellikle robotik, havacılık başta olmak üzere bazı alanlarda nesnenin hareket ve konumunu belirlemek önemlidir. Quadcopter üzerindeki rotorlara bağlı motorlar tarafından itki kuvveti oluşturulur. Oluşan itki kuvveti yerçekimi kuvvetini yendiği noktadan itibaren quadcopter havalanmaya başlar. Aynı zamanda itki kuvveti aracın ileri geri, yukarı, aşağı hareket etmesini sağlar. Quadcopterler büyüklük olarak diğer hava araçlarına oranla daha küçük olduğu için çok hızlı bir biçimde serbestlik dereceleri tanımlarına uygun olarak hareket edebilirler. İtki kuvveti Newton (N) birimi ile ölçülür. Her bir rotorun hızı bağımsız olarak belirlenebilir. Quadcopter tarafından belirlenen görevin yapısına göre ileri, geri ve aşağı yukarı gitmesi kontrol ünitesi tarafından hızın değiştirilmesi ile sağlanır. Ayrıca quadcopter üzerinde farklı görevleri olan ivmeölçer, Jireskop, Light Detection and Ranging (Lidar) , barometre ve Global Positioning System (GPS) gibi ölçüm cihazlarından sensör verileri alınarak istenilen uçuş davranışları için gerekli düzeltmelerin yapılması sağlanır. Quadcopter etki eden kuvvetler ve dönme davranışları (moment) Newton'un yasalarına uygun olarak hareket denklemleri ile ifade edilebilir. Quadcopter tasarımsal durumu ve yerçekimi kuvveti moment kuvvetinin oluşumunda etkilidir. Hareket denklemlerini ifade etmek için öncelikle kinematik ve dinamik kavramlarının tanımlanması gerekir. Kinematik, aracın konumunu, hızını, ivmelerini ve bunların zamanla nasıl değiştiğini açıklar. Örneğin belirli bir zaman aralığında aracın katettiği mesafe kinematik ile ilgilidir. Dinamik ise hareketleri ve bu hareketlere etki eden kuvvetleri ve nedenlerini açıklar. Dinamik kavramı için kuvvet,

kütle, ivme ve moment kavramları önemlidir. Bir cismin hareketini hangi yönde nasıl yaptığını dinamik ile açıklarız.

Aracın konum ve hızı dünya ekseni (earth frame) koordinat sistemine göre ölçülür. Bunun için aracın gövde ekseni (body frame) ve dünya ekseni arasında dönüşüm yapılması gerekir. Bunun için Euler açılarından faydalanılır. Euler açıları nesneni konumunu ve yönelimini tanımlamak için kullanılır.

Euler açıları;

a) Roll(Yana Yatma): İlk açı, nesnenin bir eksen etrafında uzunlamasına dönüşünü ifade eder ve φ (phi) ile gösterilir.
b) Pitch (Burun aşağı/yukarı hareket): İkinci açı, kanatlar arasındaki burun aşağı, burun yukarı dönüşü ifade eder ve θ (theta) ile gösterilir.
c) Yaw(Yön Değiştirme): Son açı , dikey eksene paralel dönüşü ifade eder ve ψ (psi) ile gösterilir.

Matematiksel olarak dönüşümler, dönüş matrislerinin hesaplanması ile elde edilir. Dönüş matrisleri sırasıyla Denklem (4.1), (4.2) ve (4.3)'de gösterilmiştir.

$$Roll\ (\varphi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos\varphi & -sin\varphi \\ 0 & sin\varphi & cos\varphi \end{pmatrix} \tag{4.1}$$

$$Pitch\ (\theta) = \begin{pmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{pmatrix} \tag{4.2}$$

$$Yaw\ (\psi) = \begin{pmatrix} cos\psi & -sin\psi & 0 \\ sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.3}$$

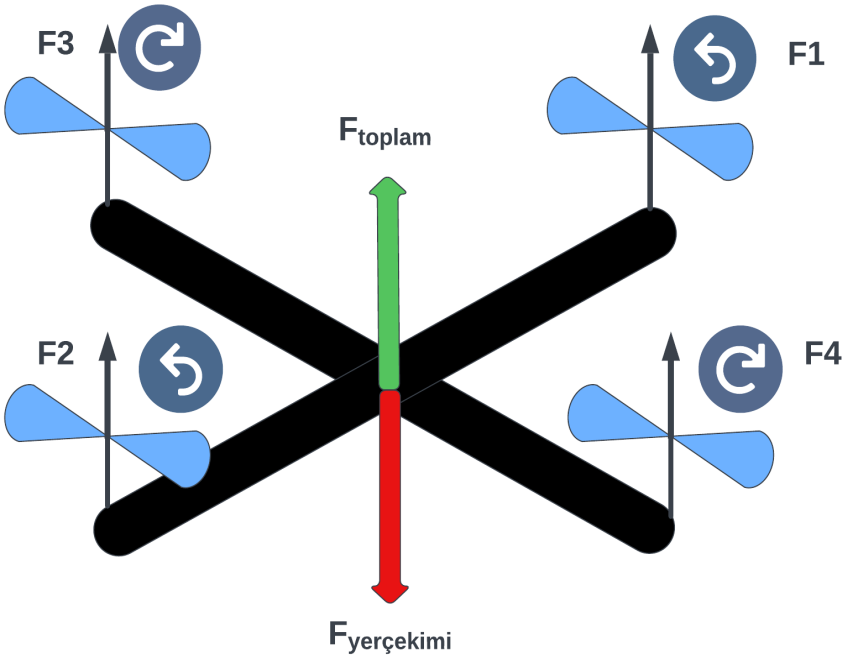
Daha sonra her açı için oluşturulan matrisler toplam dönüş matrisi için birbirleri ile çarpılır. Toplam dönüş matrisi Denklem (4.4)'te gösterilen eşitlik ile hesaplanmaktadır.

$$\text{Toplam\_Dönüş\_Matrisi} = Roll\ (\varphi)\ x\ Pitch\ (\theta)\ x\ Yaw\ (\psi) \tag{4.4}$$

Toplam dönüşüm matrisi nesnenin başlangıç noktasına göre ve hesaplanan Euler açılarına göre dönüş konumunu ifade eder. İHA hareket boyunca kuvvetler ve momentler etkisinde

kalır. Bu kuvvetler genel olarak rotorların hareketi sonucu oluşan itki (thrust) kuvveti ve aracı aşağı doğru çeken yerçekimi kuvvetidir.

Quadrotorda toplam 4 adet rotor bulunur. Bu rotorların farklı yönlere, farklı hızlarda yapmış oldukları hareketle İHA hareket eder. Şekil 4.2'de İHA'ya etkin kuvvet ve momentler gösterilmiştir.



Şekil 4.2. İHA'ya etki eden kuvvet ve momentler

Tüm rotorlar tarafından üretilen itki kuvvetinin yerçekimi kuvvetine eşit olduğu durumda İHA askıda kalır, Askıda kalma durumunda kuvvet ve moment toplamı sıfır olur. Bu durum Denklem (4.5)'te gösterilen eşitlik ile hesaplanmaktadır. Toplam itki kuvveti yerçekimi kuvvetinden büyük olduğunda İHA yukarı doğru hareket eder.

$$F_1 + F_2 + F_3 + F_4 = F_{yerçekimi} \quad (4.5)$$

4.2.Matlab

MATLAB (Matrix Laboratory), matris işlemleri ve sayısal hesaplamalar için özel olarak tasarlanmış bir yüksek seviyeli programlama dilidir. MATLAB, mühendislik, matematik,

fizik, bilgisayar bilimi ve diğer birçok disiplinde yaygın olarak kullanılmaktadır. İşte MATLAB hakkında bazı temel akademik bilgiler:

Matematik ve Mühendislik Uygulamaları: MATLAB, matris ve dizi operasyonları için optimize edilmiş bir dil olduğundan, matematik ve mühendislik uygulamalarında yaygın olarak kullanılır. Lineer cebir, diferansiyasyon, integral hesaplamaları, optimizasyon ve benzeri birçok matematiksel operasyon MATLAB ile kolayca gerçekleştirilebilir.

Grafik ve Görselleştirme: MATLAB, verileri analiz etmek ve sonuçları görselleştirmek için güçlü grafik ve çizim araçları içerir. 2D ve 3D grafikler oluşturabilir, çoklu veri kümelerini karşılaştırabilir ve çeşitli görselleştirmeleri yapabilirsiniz.

Simülasyon ve Modelleme: MATLAB, dinamik sistemlerin simülasyonu ve modellenmesi için yaygın olarak kullanılır. Simulink adlı bir eklenti, kullanıcıların fiziksel sistemleri modellerken blok diyagramları kullanmalarını sağlar.

Sinyal İşleme ve Görüntü İşleme: MATLAB, sinyal işleme ve görüntü işleme uygulamalarında etkili bir şekilde kullanılabilir. Ses işleme, görüntü iyileştirme ve analiz, filtreleme gibi birçok işlem MATLAB içinde yer alır.

Veri Analizi ve İstatistik: MATLAB, veri analizi ve istatistik uygulamaları için geniş bir araç setine sahiptir. Veri kümelerini işleyebilir, istatistiksel analizler yapabilir ve sonuçları görselleştirebilirsiniz.

Araştırma ve Eğitim: MATLAB, birçok üniversite ve araştırma kuruluşunda öğretim ve araştırma amaçlarıyla kullanılmaktadır. Öğrenciler ve araştırmacılar, MATLAB'ı matematiksel modeller oluşturmak, simülasyonlar yapmak ve çeşitli disiplinlerdeki problemleri çözmek için kullanabilirler.

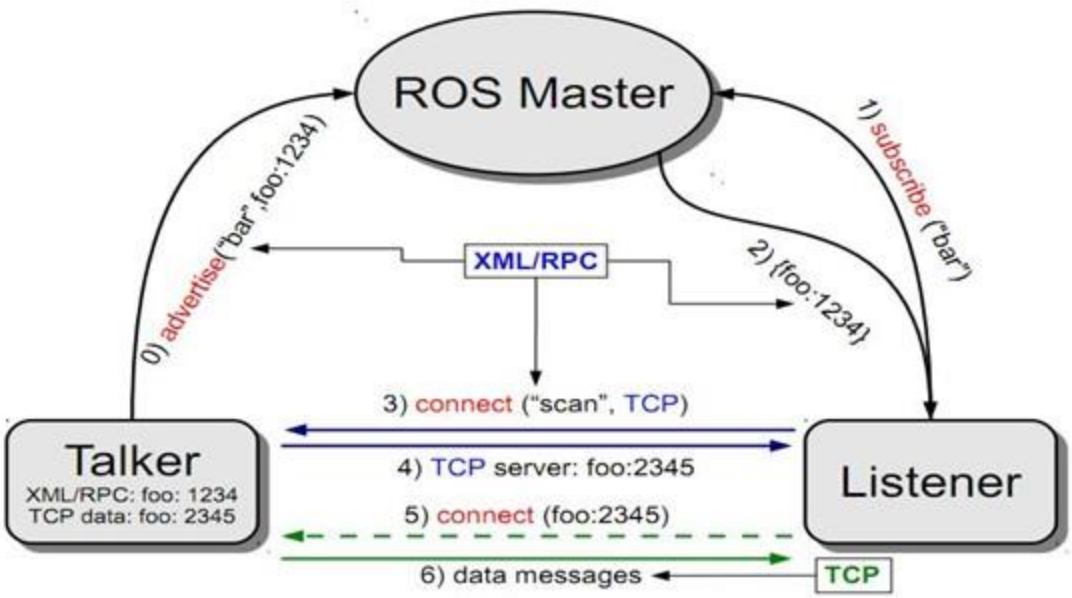
Scripting ve Programlama: MATLAB, işlevsel programlama ve komut satırı betiği yazımına olanak tanır. Ayrıca, MATLAB dilinde yazılmış işlevleri içeren kendi özel programlarınızı oluşturabilirsiniz.

MATLAB, endüstride ve akademik çevrelerde geniş bir kullanıcı tabanına sahiptir ve sürekli olarak geliştirilmektedir. MATLAB'ın kendi dokümantasyonu ve çevrimiçi

toplulukları, kullanıcıların sorunları çözmelerine ve yeni özellikleri öğrenmelerine yardımcı olur.

4.3.ROS

Gazebo içerisinde modellenen İHA simülasyonuna komut göndermek, uçuş verilerini almak ve yol planlama koordinatlarını yeniden güncellemek için ROS kullanılmıştır. ROS isminden farklı olarak işletim sistemi değildir, çalışmak için işletim sistemine ihtiyaç duyar. ROS robotları programlamak için kütüphane ve framework alt yapısı sağlar. ROS yazılım altyapısı grafiksel gösterimi Şekil 4.3'de verilmiştir.



Şekil 4.3. ROS yazılım altyapısının grafiksel gösterimi (URL-4)

ROS dış dünyadan aldığı verileri işleyip yeniden robot platformuna aktaran ve oldukça yaygın olarak açık kaynak kodlu bir yazılımdır. ROS ile robot (İHA platformu) arasındaki iletişim ROS alt yapısının sunduğu topic ve mesajlarla sağlanır. ROS'un birçok avantajı vardır, C++, Python, Java gibi programlama dillerini ayrı ayrı destekleyebilmektedir.

Çalışma kapsamında öncelikle Roscore ile master node çalıştırılmıştır. ROS üzerinden çalıştırılacak diğer node'lar (robot üzerinden işlem yapabilen birim/yazılım), master node üzerinden haberleşecektir. ROS üzerinden Package haline getirilen veri çok kolay bir biçimde Gazebo üzerinden simüle edilebilmektedir.

Aşağıda örnek olarak Ubuntu İşletim Sistemi ortamında Roscore komutuyla master node'un çalıştırılması gösterilmiştir; master node, ihtiyaç halinde çalıştırılacak diğer node'lar çalıştırılmadan öncelikli olarak çalıştırılmalıdır.

ayhangultekin@ayhangultekin-Lenovo-G50-70:~$ sudo su

root@ayhangultekin-Lenovo-G50-70:/home/ayhangultekin# roscore

logging to /root/.ros/log/5806fd84-dc97-11ea-bafc-adbee19e1858/roslaunch-ayhangultekin-Lenovo-G50-70-15480.log

Checking log directory for disk usage. This may take a while.

Press Ctrl-C to interrupt

Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://ayhangultekin-Lenovo-G50-70:37805/

ros_comm version 1.15.8

SUMMARY

========

PARAMETERS

**** /rosdistro: noetic***

**** /rosversion: 1.15.8***

NODES

auto-starting new master

process[master]: started with pid [15507]

ROS_MASTER_URI=http://ayhangultekin-Lenovo-G50-70:11311/

setting /run_id to 5806fd84-dc97-11ea-bafc-adbee19e1858

WARNING: Metapackage "roscopter_pkgs" should not have other dependencies besides a buildtool_depend on catkin and exec_depends.

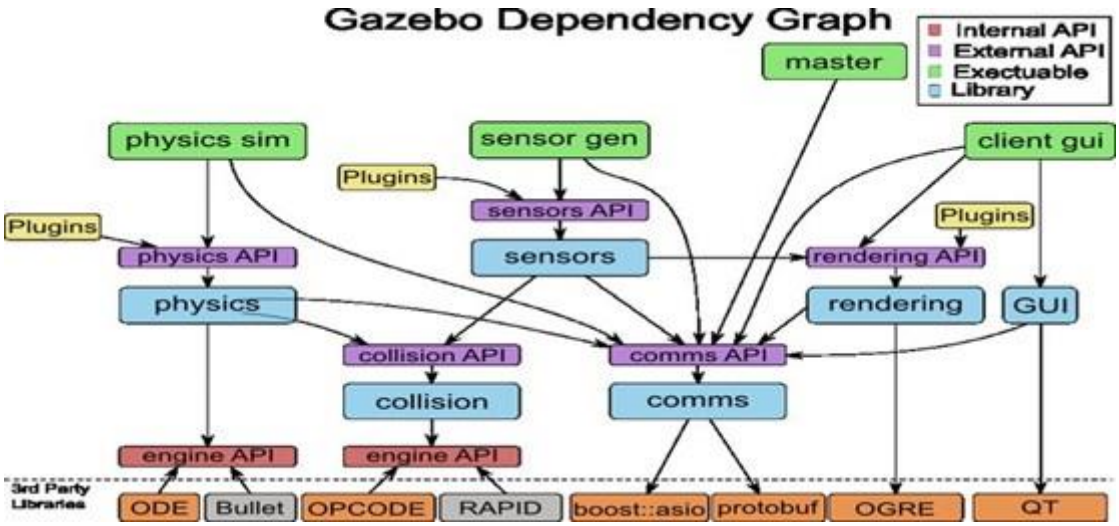
process[rosout-1]: started with pid [15517]

started core service [/rosout]

Şekil 4.4. ROS master node çalıştırılması

4.4.Gazebo

Çalışma kapsamında İHA'nın gerçek dünya şartlarında davranışlarını gözlemleyebilmek amacıyla Gazebo simülasyon programı kullanılacaktır. Gazebo içerisinde oldukça zengin simülasyon ortamı barındıran, tam ve verimli bir modelleme yapılmasını sağlayan açık kaynak kodlu bir yazılımdır. Gazebo oldukça güçlü bir grafik ara yüzüne sahiptir. Oldukça hızlı ve dinamik bir şekilde testlerin yapılmasına imkân sağlar. Çalışma kapsamında Gazebo 11.0 versiyonun Ubuntu 20.04.1 LTS işletim sistemi üzerine kurulumu yapılmıştır. Gazebo yazılım altyapısı grafiksel gösterimi Şekil 4.5'te verilmiştir.



Şekil 4.5. Gazebo yazılım altyapısının grafiksel gösterimi (URL-5)

Physics sim, temel simülasyon bileşenlerinin kullanılması için gerekli olan bir kütüphanedir, bu bileşenler arasında robot eklem noktaları, sert gövde kısımları ve çarpışma şekilleri bulunmaktadır, Physic sim üzerinden farklı açık kaynak motorları ile internal api'ler üzerinden iletişim sağlanabilir.

Sensor gen, rendering ve physics kütüphanelerine bağlı olarak simülasyon modellemesinde kullanılacak sensörlere erişim ve sensörler üzerinden alınacak verilerin dışa aktarımını sağlar.

İstemci arayüzü, rendering ve sensör kütüphanelerini kullanarak 3 boyutlu sahnelerin oluşturulmasını sağlar.

Physics, sensor ve rendering kütüphaneleri plugin kullanımı destekler. Böylece kullanıcılar gazebo iletişim sistemini kullanmadan gazebo kütüphanelerine erişebilir.

4.5.Px4

Çalışma kapsamında simülasyon ortamında yapılacak İHA uçuşları için uçuş kontrol yazılımı olarak Px4 açık kaynaklı uçuş kontrol yazılımı kullanılmıştır. Aşağıda belirtilen özellikler açısında Px4 kullanımı avantajlıdır (URL-6).

Açık Kaynaklı: PX4, açık kaynaklı bir proje olup, GNU Genel Kamu Lisansı (GPL) altında dağıtılmaktadır. Bu, kullanıcıların kodu inceleyebilmesi, değiştirebilmesi ve projeye katkıda bulunabilmesi anlamına gelir.

Modüler Mimarisi: PX4, modüler bir yazılım mimarisine sahiptir. Bu modüler yapı, farklı sensörleri, uçuş kontrol algoritmalarını, iletişim protokollerini ve görevlendirme stratejilerini kolayca entegre etmeyi mümkün kılar.

Çeşitli Donanım Desteği: PX4, farklı donanım platformlarını destekleyen geniş bir cihaz yelpazesine uygulanabilir. Bu, çeşitli İHA ve İHS donanımları için kullanım esnekliği sağlar.

Gelişmiş Uçuş Kontrol Algoritmaları: PX4, farklı uçuş kontrol algoritmalarını destekler. Bu algoritmalar, otomatik uçuş, otonom gezinti, görev planlama ve diğer uçuş görevlerini gerçekleştirmek için kullanılabilir.

QGroundControl Arayüzü: PX4 ile genellikle QGroundControl adlı bir yer istasyonu yazılımı kullanılır. QGroundControl, İHA'nın uçuş parametrelerini yapılandırmak, uçuş görevlerini planlamak ve gerçek zamanlı uçuş verilerini izlemek için kullanılır.

Açık Standartlar ve Protokoller: PX4, açık standartlar ve iletişim protokollerini destekler. Bu, PX4'ün çeşitli donanım ve yazılım bileşenleri ile entegrasyonu kolaylaştırır.

Araştırma ve Geliştirme İçin Kullanım: PX4, araştırmacılar ve geliştiriciler için bir platform sağlar. Akademik araştırmalarda, prototipler oluşturmakta ve öğrencilere uçuş kontrolü konularında pratiğe imkân tanımak amacıyla kullanılabilir.

Şekil 4.6’da Px4 oto pilot yazılım grafiksel gösterimi verilmiştir.

Şekil 4.6. Px4 oto pilot yazılımı (URL-6)

5.YÖNTEM

Tez çalışması kapsamında, İHA yol planlama probleminin çözümü için hibrid olarak çalıştırılan RRT ve YPA algoritmaları ile üretilen yol veri kümesinin, YSA türlerinden biri olan FFNN ile oluşturulan YSA modelinin eğitimi için kullanılacağı bir yöntem önerilmiştir. Literatür araştırması sonucunda incelenen diğer yöntemlere göre aşağıda belirtilen yönler açısından geliştirilen yöntemin daha avantajlıdır.

1. Esneklik ve öğrenme yeteneği,
2. Karmaşık ve belirsiz ortamlarda yönetme yeteneği,
3. Büyük veri setleri ile geniş öğrenme yeteneğini ,
4. Öznitelik öğrenme ve veriler arasındaki karmaşık ilişkiyi anlama,

Hibrid algoritmada kullanılan RRT, başlangıç noktasından belirlenen bir hedef noktaya ulaşmak için geniş bir ortamda yol planlama probleminin çözümü için kullanılan genel yol planlama yöntemidir. Genel yol planlama yöntemlerinde harita veya ortam bilgisi ile engellerin konumuda önceden bilinmelidir. Hibrid algoritmada kullanılan diğer yöntem YPA ise anlık engellerden sakınma ve çevresel değişikliklere anlık olarak tepki gösterebilen yerel yol planlama yöntemlerinden birisidir. Bu iki yöntemin birlikte kullanımı ile yol planlama problemine bütüncül bir stratejik yaklaşım geliştirilmiştir. Bu iki yöntemin birlikte kullanımı ile her ikisininde dezavantajlı olduğu noktaların minimize edilmesi sağlanmıştır. Öncelikle RRT ile başlangıç noktasında hedef noktaya harekette genel bir plan oluşturulmuş sonrasında YPA ile rastgele üretilen nokta için bileşke vektörel alanlar hesaplanarak hareket edilecek bir sonraki nokta belirlenmiştir. Böylece hedefe doğru sürekli bir yönelme olması sağlanmıştır. Bu durum özellikle RRT algoritmasının önemli bir dezavantajı olan bulunan yolun süre ve uzunluk açısından en uygun yol olmama sorununu azaltmaktadır. YPA algoritmasının temelinde hedefe doğru sürekli bir yönelme durumu vardır. YPA algoritması gradyan temelli bir algoritmadır. YPA algoritmasında iten ve çeken kuvvetlerin bileşkesi alınarak bir sonraki hareket noktası hesaplanır.

Geliştirilen yöntem içerisinde oluşturulan yolun grafiksel gösterimi öncesinde yolun pürüzsüz ve daha yumuşak olması için spline fonksiyonları kullanılmıştır. Spline matematikte belli aralıklarda düzgün eğri ve yüzey oluşturmak için kullanılan bir

tekniktir. Spline'lar genellikle kontrol noktalarına polinom setleri kullanılarak tanımlanır ve bu polinom setleri belirli kurallara göre birbirine bağlanır. Genellikle, bir dizi kontrol noktası veya veri noktası arasında düzgün bir eğri veya yüzey oluşturmak için kullanılır.

Matematiksel olarak, *n* adet kontrol noktasını içeren bir Spline, Denklem (5.1)'deki gibi ifade edilir.

$$S(x) = \sum_{i=1}^{n} P_i(x).N_i(x) \tag{5.1}$$

Burada ;

$P_i(x)$, i-inci kontrol noktasının polinomunu temsil eder.

$N_i(x)$, bir baz (basis) fonksiyonu olarak adlandırılır ve genellikle her iki kontrol noktası arasındaki etkiyi belirler. Baz fonksiyonları, özellikle B-spline'lar gibi spline türlerinde, genellikle lokal destekli ve birbirine bağlıdır, bu da spline'ın belirli bir bölgesindeki değişikliklerin diğer bölgeleri etkilememesini sağlar.

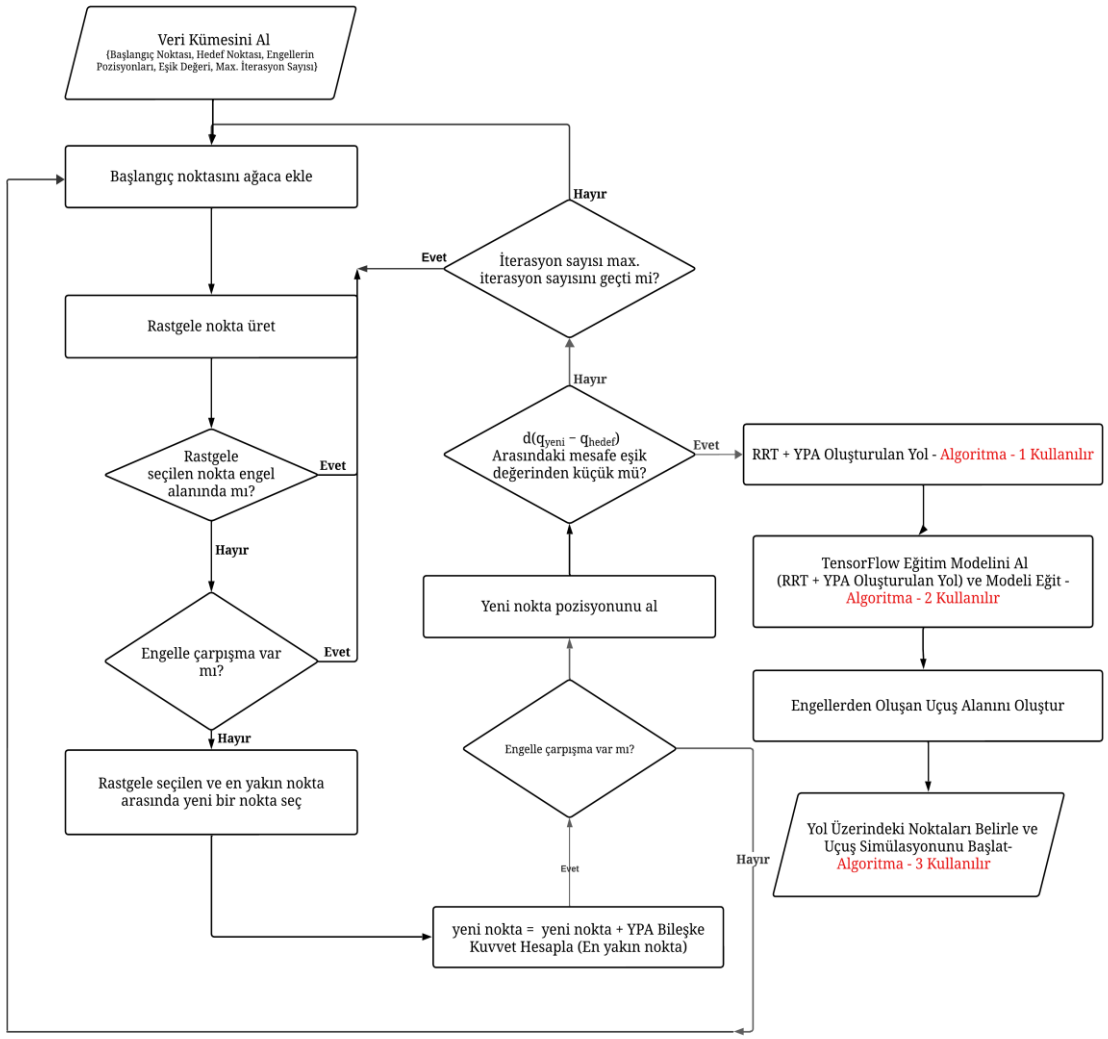
Sunulan çalışma içerisinde geliştirilen yöntemde kubik spline kullanılmıştır. Baz fonksiyonları, özellikle B-spline'lar gibi spline türlerinde, genellikle lokal destekli ve birbirine bağlıdır, bu da spline'ın belirli bir bölgesindeki değişikliklerin diğer bölgeleri etkilememesini sağlar.

Matematiksel olarak kubik spline Denklem (5.2)'de ki gibi ifade edilir.

$$S(x) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i \tag{5.2}$$

Sunulan yöntem içerisinde elde edilen noktalar içinde yol grafiği çizdirilmeden önce kontrol noktaları oluşturularak spline fonksiyonları hesaplanmış ve yolun daha yumuşak ve pürüzsüz olması sağlanmıştır. Çalışma kapsamında kullanılan spline fonksiyonları ile özellikle kinematik kısıtların aşılması sağlanmıştır. Böylece iki nokta arasında keskin dönüşlerin yapılması durumunda bu iki nokta arasında aracın hareketinin daha yumuşak yapılabilmesi sağlanmıştır.

Yöntemin genel olarak akış şeması Şekil 5.1'de verilmiştir.



Şekil 5.1.Yöntem akış şeması

Yöntemin akış şemasına uygun olarak öncelikle MATLAB ortamında Hibrid algoritma kodlanmıştır. 10x10, 25x25 ve 50x50 ortamları için ayrı ayrı algoritma çalıştırılmıştır. Kodlama içerisinde her bir ortam için kullanılacak engel sayıları ortam büyüklüğüne göre belirlenmiştir. Ortam içerisinde kullanılacak engellerin bulunacakları koordinatlar tamamen rastgele olarak belirlenmiştir. Yöntemin temelinde bu rastgele seçimler vardır. Bu şekilde yol veri kümesinin oluşturulmasından sonra YSA ile eğitim aşamasında daha iyi bir eğitim doğruluk oranı ile eğitim kayıp oranlarınında düşürülmesi hedeflenmiştir.

Veri kümelerinin oluşturulması sonrasında TensorFlow YSA modelinin eğitimi aşamasında modelin farklı uygulamalar tarafından çağrılarak kullanılabilmesi için .tflite dosyası oluşturulmuştur.

TFLite (TensorFlow Lite), Google'ın TensorFlow makine öğrenimi çerçevesinin hafif bir sürümüdür. TensorFlow Lite, özellikle mobil cihazlar, gömülü sistemler ve diğer kaynak sınırlı ortamlar için optimize edilmiş modellerin dağıtımını ve çalıştırılmasını sağlamak için tasarlanmıştır.

TensorFlow Lite, orijinal TensorFlow'un çeşitli avantajlarından vazgeçmeden, hafif ve hızlı olması için bir dizi optimizasyon içerir. Bu, mobil cihazlar gibi kaynak sınırlı cihazlarda daha etkili bir şekilde çalışabilen makine öğrenimi modellerini uygulamak için ideal bir çözümdür. Başlangıç ve hedef noktalar verilerek hibrid yol planlama algoritmasının çalıştırılmasının sözde kodu Tablo 5.1'de gösterilmiştir.

Tablo 5.1. Hibrid algoritma yol veri kümesi üretme sözde kodu

Algoritma 2. Hibrid algoritma yol veri kümesi üretme sözde kodu

function *RRT_APF_Yol_Üretim*($x_{baslangıç}$, x_{hedef})
    $V \leftarrow \{x_{başlangıç}\}$;
    $E \leftarrow \emptyset$;
    $G = (V, E)$;
    for $i = 1, \ldots, n$ **do**
        $x_{rastgele} \leftarrow Random\_Nokta$;
        $x_{enyakın} \leftarrow En\_Yakın\_Nokta\ (G \leftarrow (V, E), x_{rastgele})$;
        $x_{yenidurum} \leftarrow En\_Yakın\_Nokta\ (YPA\_Bileşke\_Kuvvet)$;
        $x_{yenidurum} \leftarrow X_{yenidurum} + Uzunluk(x_{enyakın} + x_{rastgele})$
        if *Çarpışma_Kontrol*($x_{yenidurum}$) ***then***
            $V \leftarrow V \cup \{ x_{yenidurum} \}$;
            $E \leftarrow E \cup \{(x_{enyakın}, x_{yenidurum})\}$;
        end if
    end for
    $Yol\_Veri\_Kümesi = (V, E)$;
    return *Yol_Veri_Kümesi*;
end function

Hibrid algoritmanın çalıştırılması sonrası oluşturulan yol veri kümesinin çalışma kapsamında oluşturulan yapay zekâ modelinin eğitimine bir girdi olarak verilmesi, eğitim süreci ve son olarak eğitim süreci tamamladıktan sonra uçuş ortamından alınan bir

noktanın yol seçiminde kullanılıp kullanılmayacağını gösteren sözde kodu Tablo 5.2' de gösterilmiştir.

Tablo 5.2. Yapay sinir ağı modeli eğitim sözde kodu

Algoritma 2. Yapay sinir ağı modeli eğitim sözde kodu

```
function Eğitim(Yol_Veri_Kümesi)
    val_dataframe = all_data.sample(Yol_Veri_Kümesi)
    tra_dataframe = all_data.drop(val_dataframe.idx)
    encode_numerical_feature(feature, name, dataset)
    model.compile()
        if model_tahmin > 0.2 then
                nokta_seçilmedi
        else
                nokta_seçildi
        end if
     return nokta_seçilen
end function
```

Yapay zekâ modeli tarafından seçim kriterine uygun olarak belirlenen noktalardan oluşan yolun uçuş ortamı içerisinde simülasyon ortamı üzerinden gerçeklenmesinin sözde kodu Tablo 5.3'de gösterilmiştir.

Tablo 5.3. Uçuş simülasyonu sözde kodu

Algoritma 3. Uçuş simülasyonu sözde kodu

```
function Uçuş(Nokta)
      val_tf_model = invoke(Eğitim_Model)
        for i = 1, . . . , n do
                tf_model ← Nokta_i;
        if tf_model_prediction > 0.2 then
            Nokta_seçilmedi
        else
           Nokta_seçildi
        Uçuş_Simülasyon_Noktası.add = nokta_seçildi
        end if
      return Uçuş_Simülasyon_Noktası
      end for
end function
```

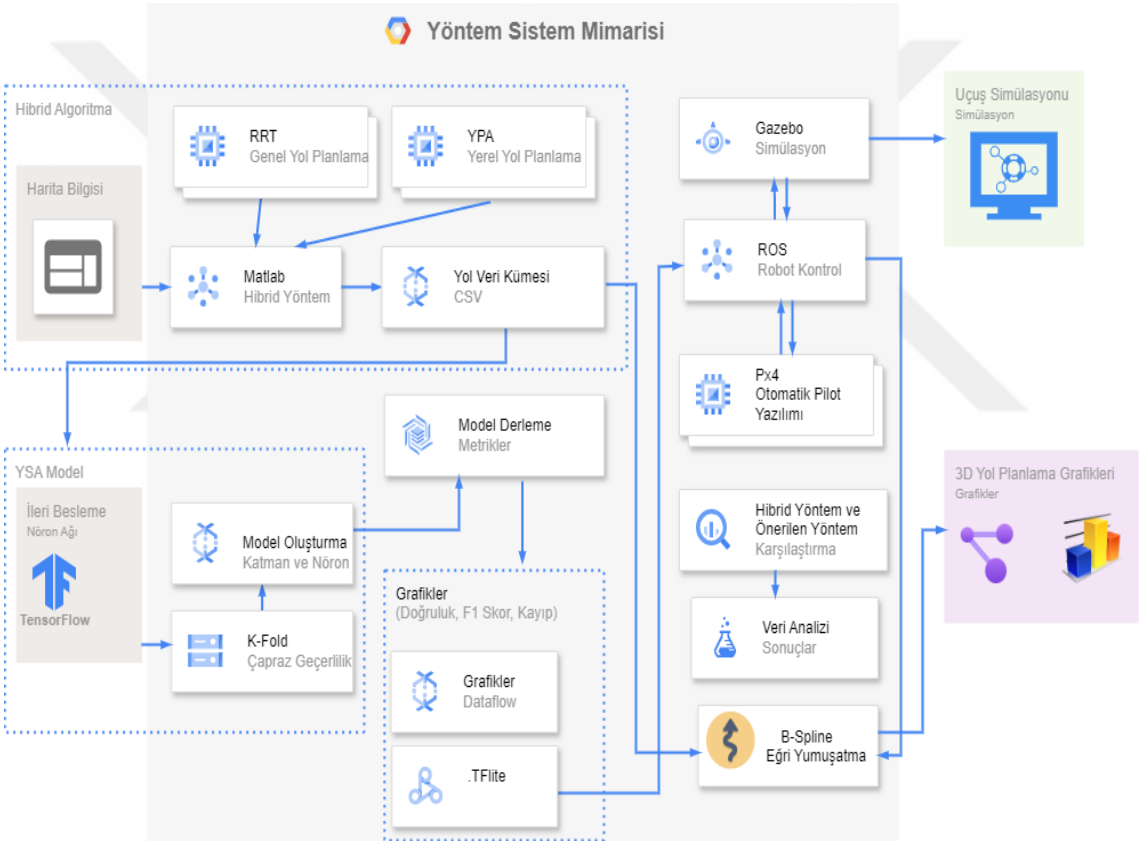
Yukarıda belirtilen fonksiyonel blokların çağrılması için kullanılan ana bloğun sözde kodu Tablo 5.4' de gösterilmiştir.

Tablo 5.4. Önerilen yöntemin ana blok sözde kodu

Algoritma 4. Önerilen yöntemin ana blok sözde kodu
function *Ana()*
$G_{rrt\text{-}apf}$ = *RRT_APF_Path_Generate* ($x_{başlangıç}$, x_{hedef});
$G_{eğitim}$ = *Eğitim(Yol_Veri_Kümesi);*
$G_{uçuş\_noktaları}$ = *Flight(Yol_Veri_Kümesi);*
end function

5.1.Sistem Mimarisi

Çalışma kapsamında oluşturulan yöntemin sistem mimarisi Şekil 5.2'de gösterilmiştir.



Şekil 5.2. Sistem Mimarisi

6. SONUÇLAR VE ÖNERİLER

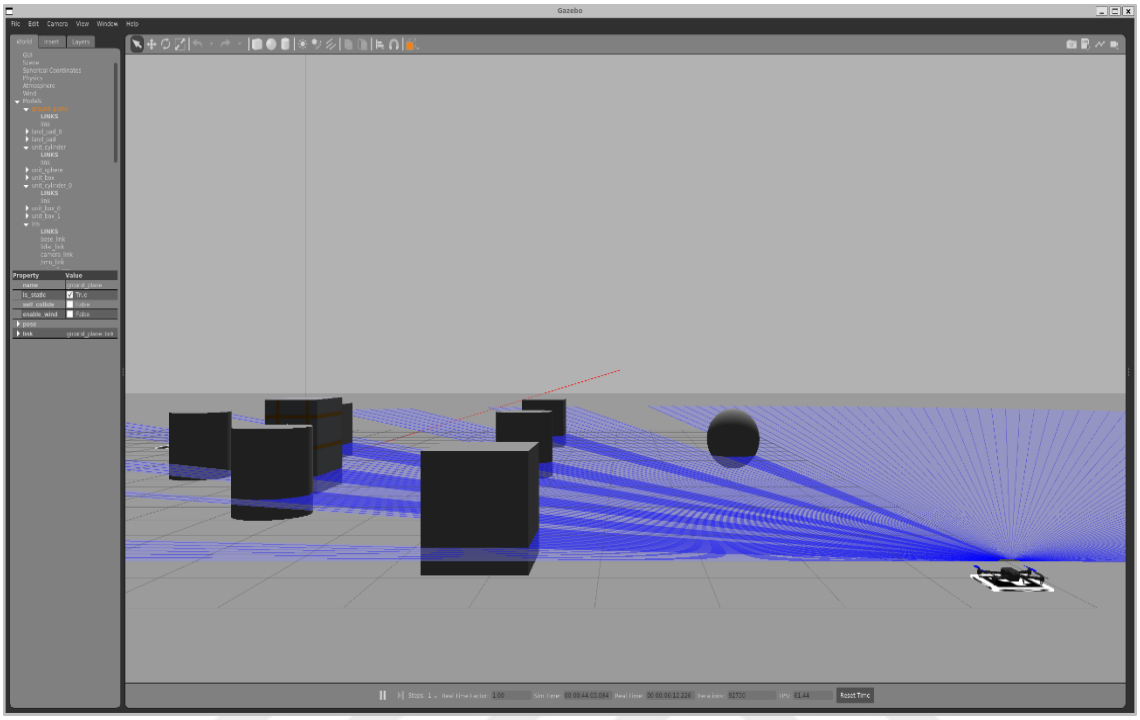
Bu tez çalışması kapsamında eğitilmiş olan YSA modelinin performansını değerlendirebilmek amacıyla testler yapılmıştır. Yapılan testlerde hibrid yol planlama algoritmasından elde edilen sonuçlarla YSA modelinden alınan sonuçlar karşılaştırılmıştır. Alınan sonuçlar Tablo 6.1'de verilmiştir.

Tablo 6.1. Farklı haritalar için alınan deneysel sonuçlar

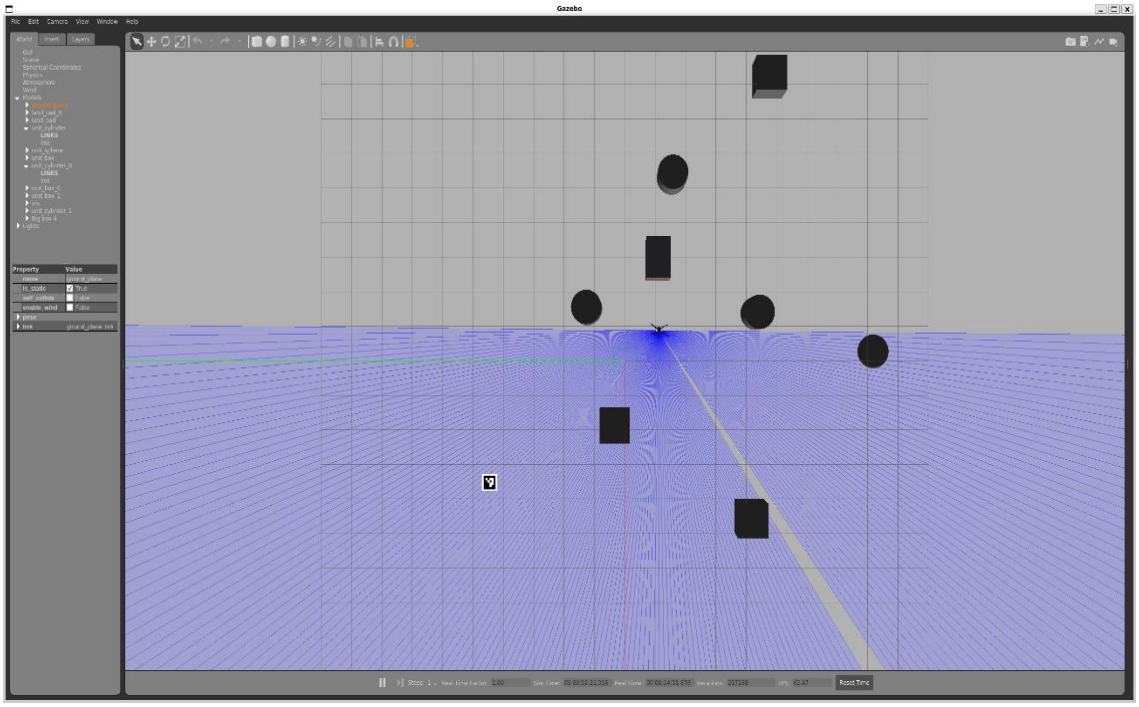
Harita Büyüklüğü [m]	Algoritma	Başlangıç Noktası	Hedef Noktası	Yol Uzunluğu [m]	Engel Sayısı	Çalışma Zamanı [sec]	Çalışma Zamanı Farkı [sec]	Uzunluk Farkı [m]
Map1 10 x 10	Hibrid	(8.76, 8.97, 2.05)	(0.36, 8.22, 3.15)	15.89	5	80.04	5.01	1.04
	Önerilen Yöntem	(8.76, 8.97, 2.05)	(0.36, 8.22, 3.15)	14.85	5	75.03		
Map2 25 x 25	Hibrid	(15.22, 16.59, 11.29)	(8.24, 20.05, 13.79)	13.14	10	65.05	5.02	0.03
	Önerilen Yöntem	(15.22, 16.59, 11.29)	(8.24, 20.05, 13.79)	13.11	10	60.03		
Map3 50 x 50	Hibrid	(29.80,33.99,27.74)	(24.97, 35.04, 29.98)	10.59	15	70.04	10.03	0.62
	Önerilen Yöntem	(29.80, 33.99, 27.74)	(24.97, 35.04, 29.98)	9.97	15	60.01		

Çalışma kapsamında sunulan yöntem, 64 bit Ubuntu 20.04 işletim sistemi ve 16GB ram donanımı üzerinden geliştirilen kodlarla test edilmiştir. İHA'ya gönderilecek uçuş verileri için ROS Noetic versiyonu Python üzerinden Rospy paketleri entegre olarak çalıştırılmış ve Gazebo 11 simülasyon yazılımı kullanılarak İHA uçuş simülasyonu yapılmıştır. Gazebo içerisinde simülasyon ortamını bulunan ve İHA modellemesi yapılması için kullanılan açık kaynak kodlu bir platformdur. ROS robotların kontrol edilmesi için kullanılan açık kaynak kodlu bir yazılı platformudur. Özellikle C++ ve Python programlama dillerine entegre bir şekilde bu diller için geliştirilen kütüphanelerin kullanımı ile robotlarla iletişim için uygulama geliştirilebilir. ROS içerisinden kullanılan ve farklı amaçlara hizmet eden uygulamalar ROS Core yapısı ile birbirleri ile iletişime kurulabilir. Ayrıca YSA modelinin eğitimi için kullanılacak veri kümelerinin oluşturulması için Matlab R2023a platformu kullanılmıştır. Şekil 6.1, Şekil 6.2, Şekil 6.3,

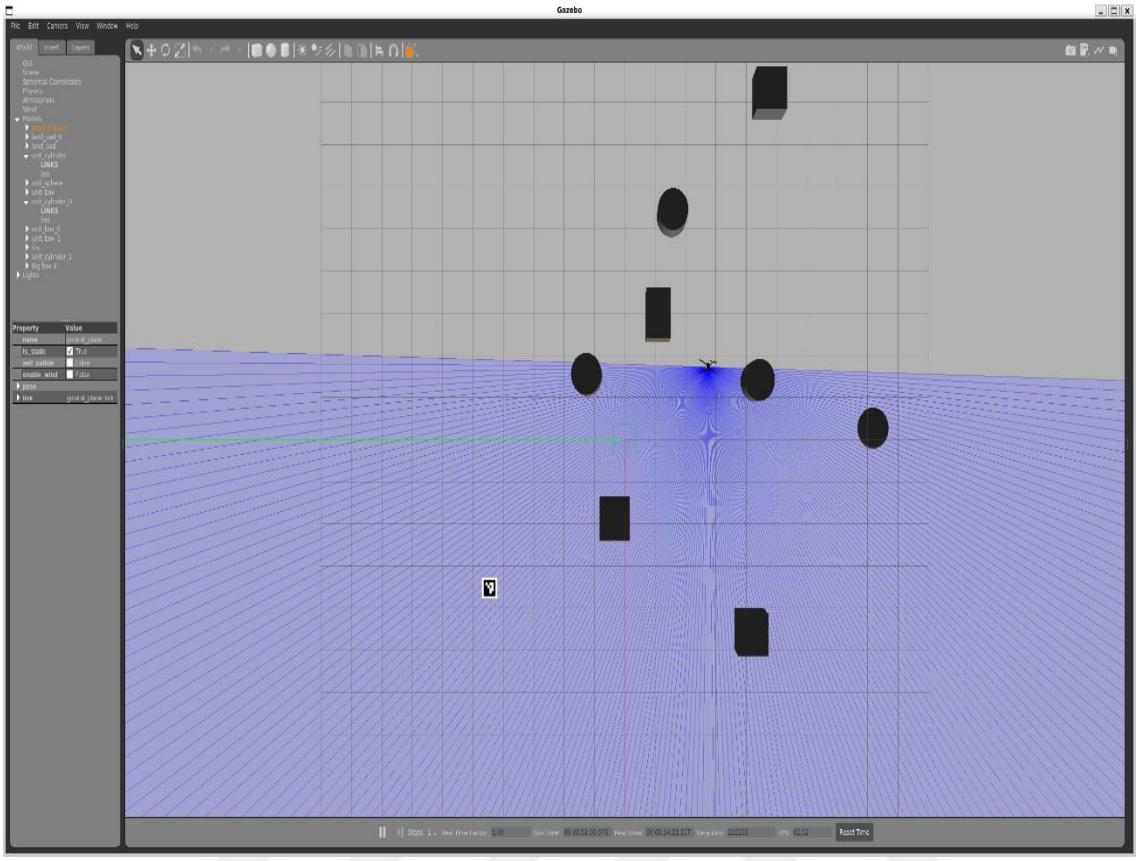
Şekil 6.4, Şekil 6.5, ve Şekil 6.6'da sırayla gazebo simülasyon ortamı içerisinde İHA'nın başlangıç noktasından hedef noktaya ilerlemesi gösterilmiştir.



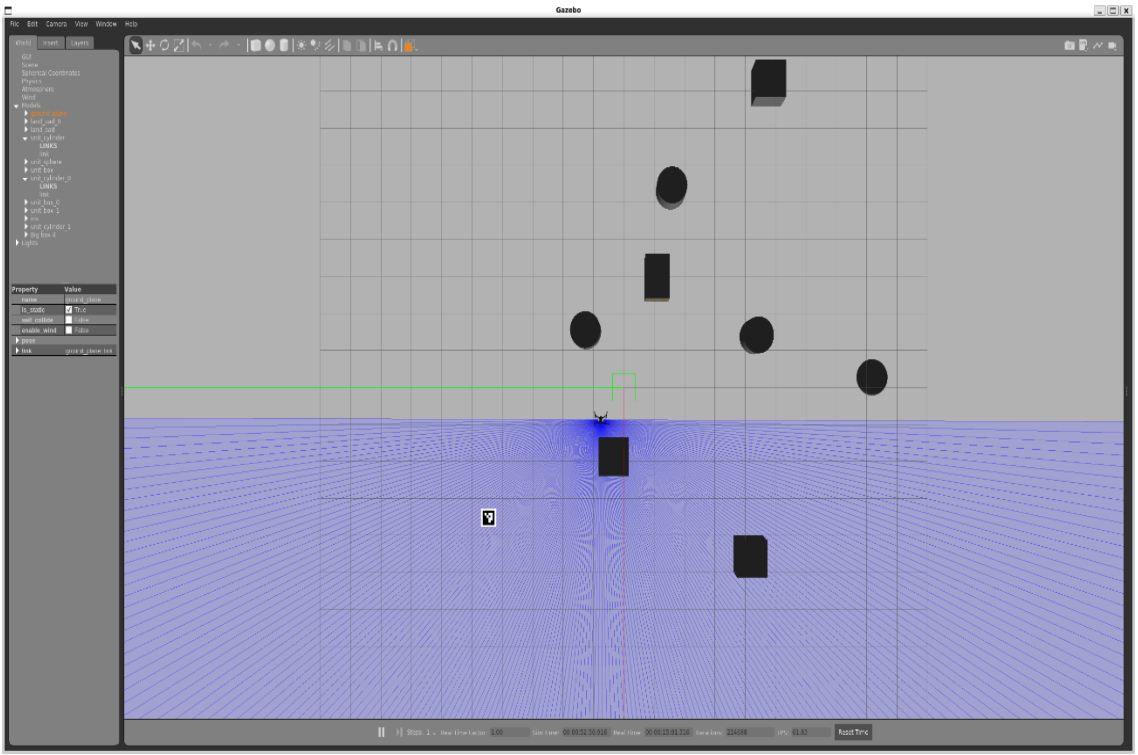
Şekil 6.1. Gazebo uçuş simülasyon - 1



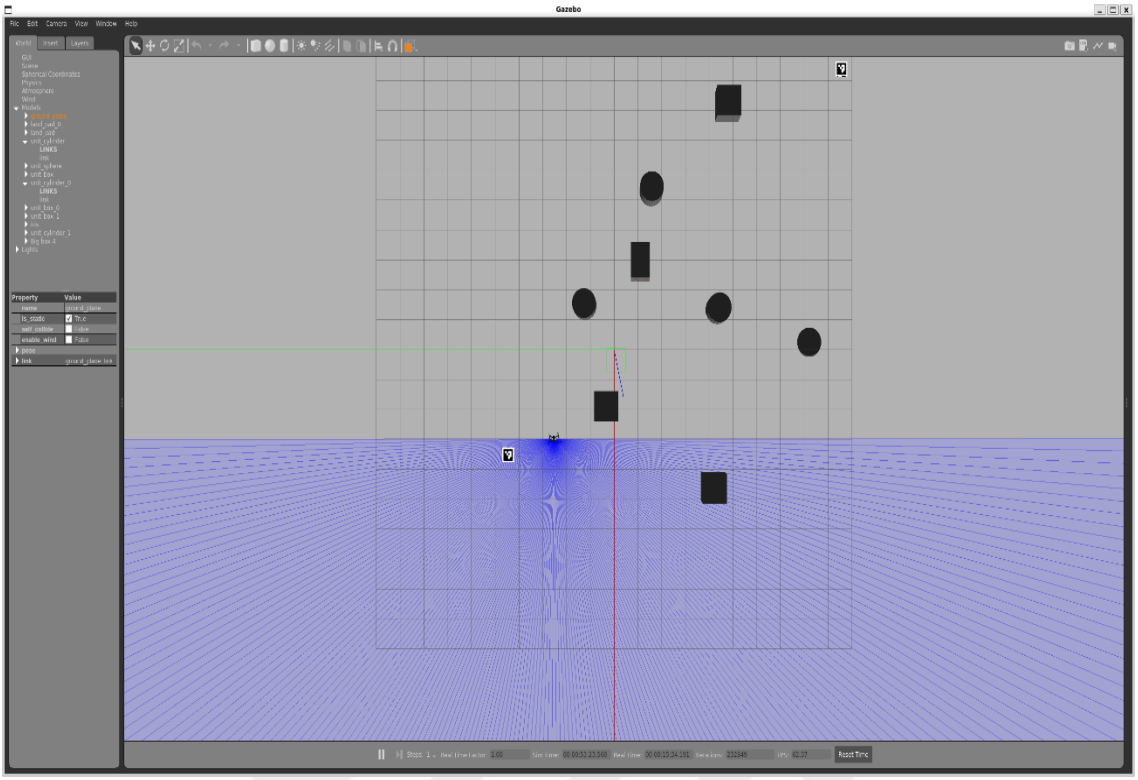
Şekil 6.2. Gazebo uçuş simülasyon - 2



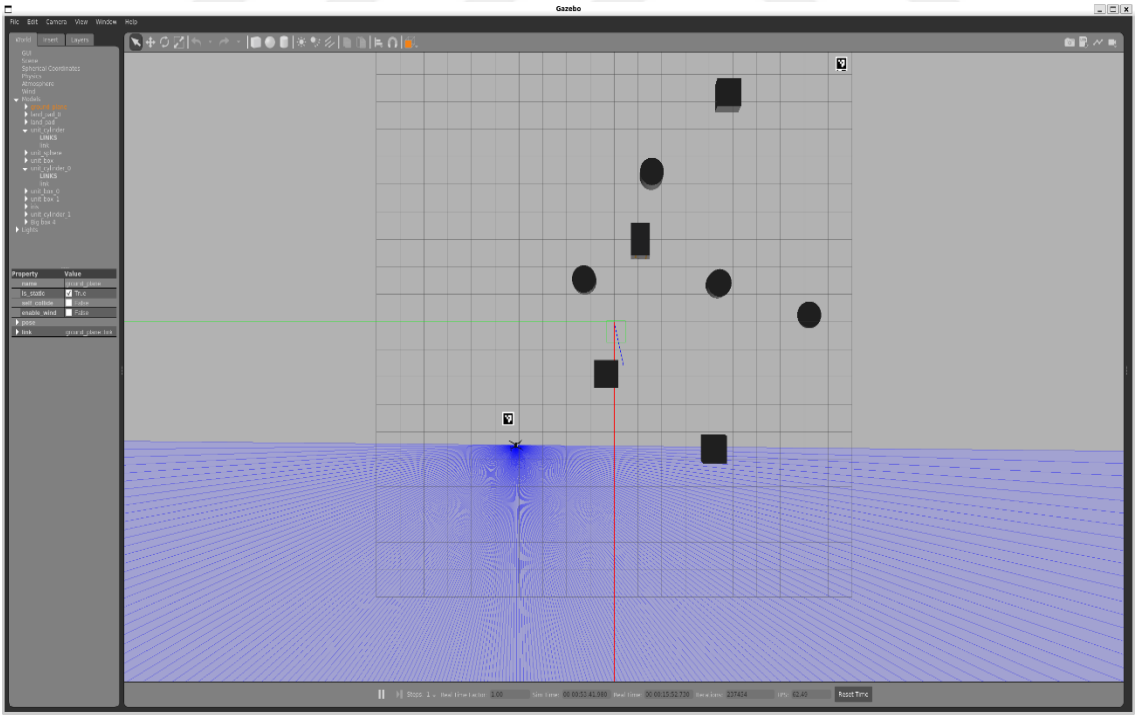
Şekil 6.3. Gazebo uçuş simülasyon - 3



Şekil 6.4. Gazebo uçuş simülasyon - 4

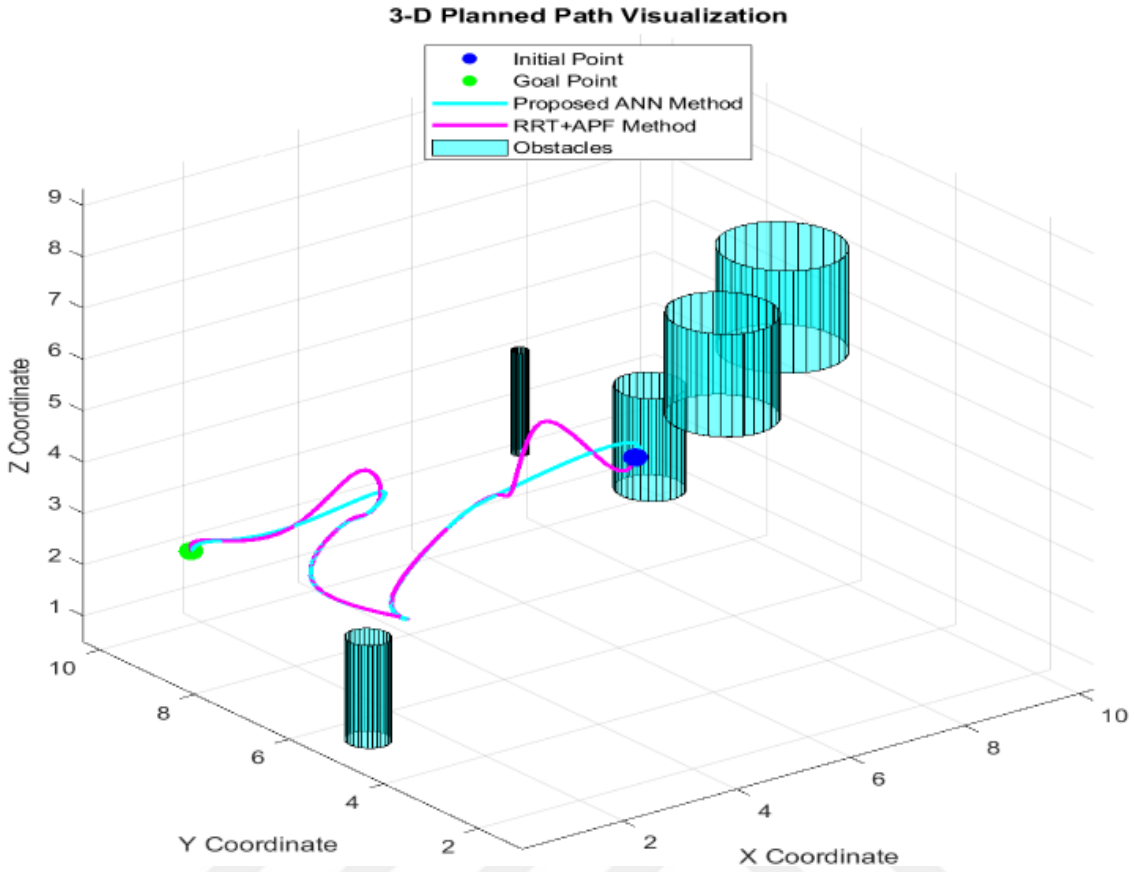


Şekil 6.5. Gazebo uçuş simülasyon - 5

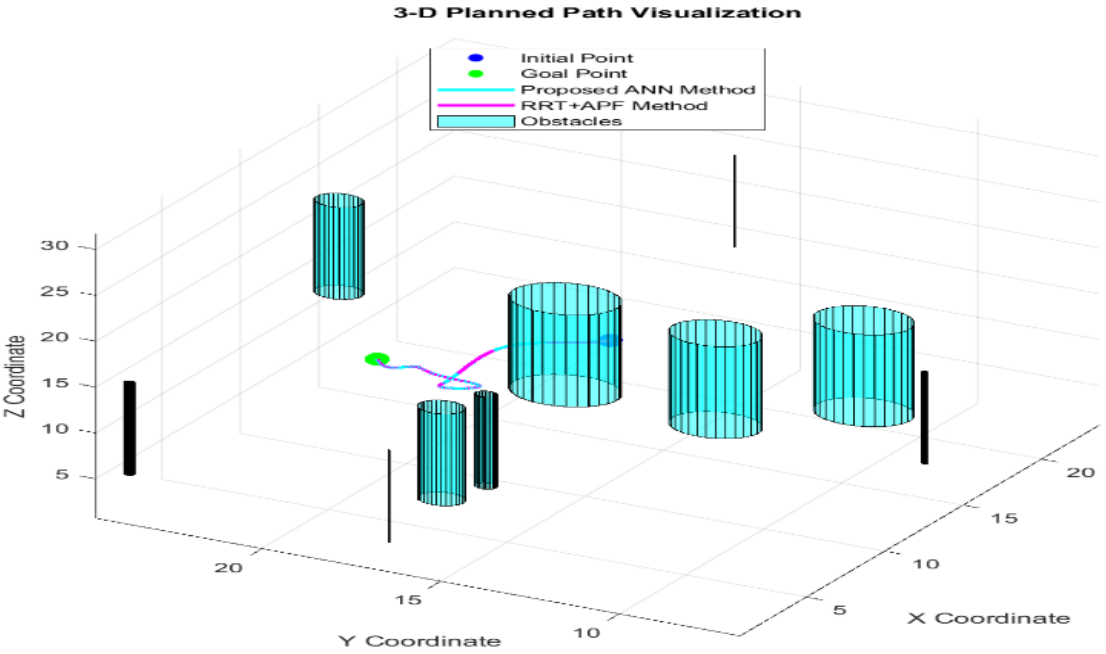


Şekil 6.6. Gazebo uçuş simülasyon - 6

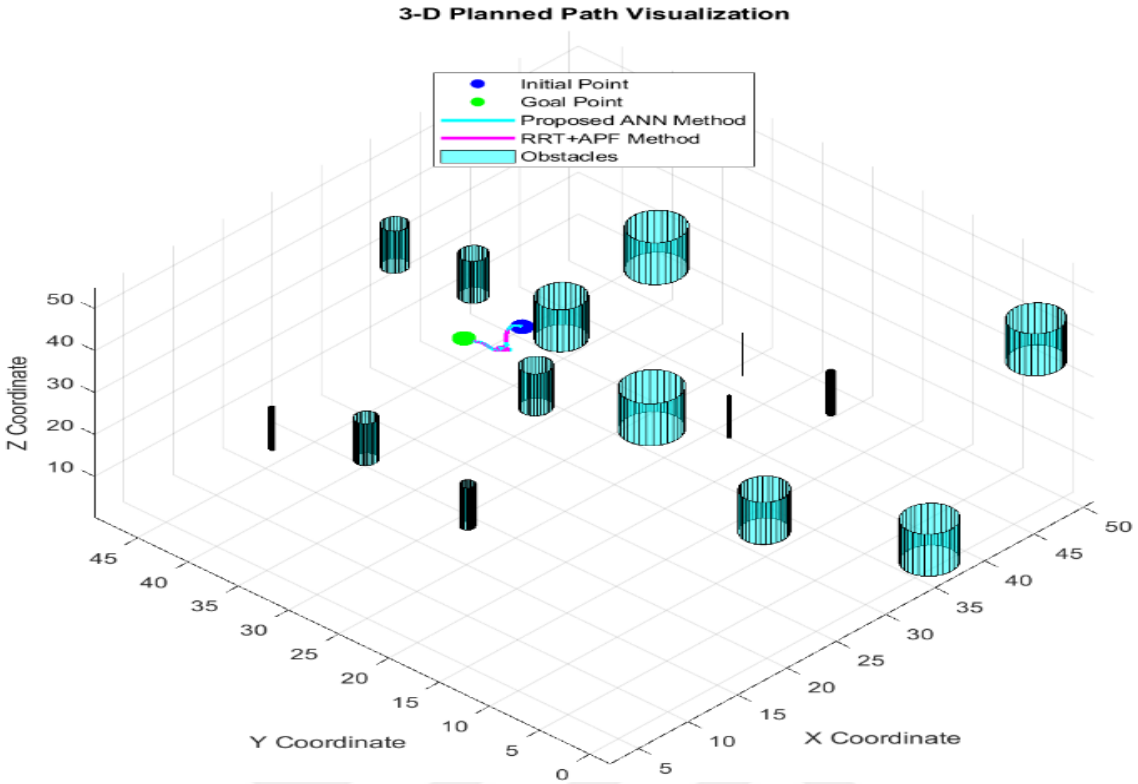
Şekil 6.7, Şekil 6.8 ve Şekil 6.9'da sırayla farklı boyutta haritalar için önerilen ve hibrid algoritma tarafından oluşturulan yollar gösterilmiştir.



Şekil 6.7. 10x10 Harita için önerilen ve hibrid algoritma tarafından oluşturulan yol



Şekil 6.8. 25x25 Harita için önerilen ve hibrid algoritma tarafından oluşturulan yol



Şekil 6.9. 50x50 Harita için önerilen ve hibrid algoritma tarafından oluşturulan yol

Uzun yıllar boyunca optimal yol planlaması ile ilgili farklı algoritmalar kullanılmıştır. Kullanılan her algoritmanın avantajları olduğu gibi dezavantajlarıda vardır. Bu çalışmada öncelikli olarak hibrid olarak RRT ve YPA algoritmaları birlikte kullanılarak çalışma kapsamında kullanılacak ve YSA modelinin eğitimi için kullanılanak yol veri kümesi oluşturulmuştur. Hibrid algortima kullanımı ve oluşturulan veri kümesi için farklı türde özellikler olan alanlar oluşturularak YSA modeli öğrenme sürecinin daha başarılı olması hedeflenmiştir. Çalışma kapsamında sunulan YSA modeline girdi olarak verilen bu veri kümesi ile eğitim süreci tamamlanmış ve YSA modelinin başarımı hibrid algoritma ile karşılaştırılarak ölçülmüştür. Ayrıca YSA modeli tarafından seçilen noktalardan oluşan yol B-spline ile yumuşatılmıştır. Gelecekte sunulacak çalışmalar kapsamında daha kompleks ortamlarda ve dinamik olarak çalışacak şekilde yöntem dahada geliştirilebilir.

Matematiksel, olasıksal, sezgizel metotların kullanımı ile İHA yol planlama probleminin çözümü için farklı yaklaşımlar geliştirilmektedir. İnsansız hava araçlarının yol planlama probleminde özellikle ortamın dinamik olarak değiştiği ve kompleks yapıda olduğu durumlarda hesaplama maliyeti oldukça önemli bir problem olarak ortaya çıkmaktadır. Bu çalışmada farklı engel pozisyonlarına sahip ortamlar için engelleride içerecek şekilde

haritalar oluşturulmuştur. RRT ve YPA algoritmaları hibrid olarak çalıştırılarak yol planlaması için elde edilen yollardan bir veri kümesi oluşturulmuştur. Oluşturulan bu veri kümesi kullanılarak bir YSA modeli eğitimi, validate ve test edilmesi işlemleri gerçekleştirilmiştir. Eğitimi tamamlanan YSA modeli, ROS kütüphanesi kullanılarak Python üzerinde geliştirilen kod tarafından çağrılmış ve YSA modeline girdi olarak verilerek koordinatların yol oluşturulmasında kullanılıp kullanılmayacağına YSA modeli tarafından karar verilmesi sağlanmıştır. Kullanılmasına karar verilen koordinatlar İHA uçuş simülasyonunda kullanılmıştır. Önerilen yeni yol planlama yaklaşımı ile geleneksel yol planlama yöntem ve algoritmalarınında olan RRT VE YPA hibrid algoritmasına göre hesaplama çalışma süresi ve yol uzunluğu gibi maliyetleri azaltılması sağlanmıştır. Sonuç olarak eğitimi gerçekleştirilen YSA modeli kullanılarak elde edilen yolun klasik RRT ve YPA yöntemi kullanılarak elde edilen yoldan %4.27 daha kısa olduğu, çalışma süresinin ise 20.06 ms daha kısa olduğu gözlemlenmiştir.

Tez çalışması kapsamında geliştirilen yöntem, alınan sonuçlarla ilişkili olarak aşağıda detayları verilen avantajlar yönünden katkı sağlamaktadır.

Öğrenme ve Optimizasyon: Yöntem içinde kullanılan YSA modeli sürekli öğrenme ve hatayı azaltma yeteneğine sahiptir. Böylece yol planlama stratejileri iyileştirebilmekte ve daha sonraki aşamalarda daha verimli kararlar verebilmektedir.

Doğrusal Olmayan İlişkiler: Tez çalışması kapsamında geliştirilen yöntem doğrusal olmayan ilişkileri yakalayıp modellemekte, böylece daha verimli yol planlaması yapılabilmesi sağlanmaktadır.

Genelleme: Tez çalışması kapsamında geliştirilen yöntem, yeni ortamlarda veya görev senaryolarında genelleştirebilmekte ve önceden elde edilen bilgi farklı şartlarda kullanabilmelerini sağlar.

KAYNAKLAR

Anderson, E., Beard, R., & McLain, T. (2005). Real-time dynamic trajectory smoothing for unmanned air vehicles. *IEEE Transactions on Control Systems Technology, 13*(3), 471–477. https://doi.org/10.1109/tcst.2004.839555

Bai, W., Wu, X., Xie, Y., Wang, Y., H, Zhao., K, Chen., Li, Y., & Hao, Y. (2018). A cooperative route planning method for multi-uavs based-on the fusion of artificial potential field and b-spline interpolation, 2018 37th Chinese Control Conference (CCC), IEEE, 2018

Bashir, N., Boudjit, S., Dauphin, G., & Zeadally, S. (2023). An obstacle avoidance approach for UAV path planning. *Simulation Modelling Practice and Theory, 129*, 102815. https://doi.org/10.1016/j.simpat.2023.102815

Cao, L., Wang, L., Liu, Y., & Yan, S. (2022). 3D trajectory planning based on the Rapidly-exploring Random Tree–Connect and artificial potential fields method for unmanned aerial vehicles. *International Journal of Advanced Robotic Systems, 19*(5), 172988062211188. https://doi.org/10.1177/17298806221118867

Chen, T., Zhang, G., Hu, X., & Xiao, J. (2018). Unmanned aerial vehicle route planning method based on a star algorithm, *2018 13th IEEE conference on industrial electronics and applications (ICIEA)*

Cui, Z., & Wang, Y. (2021). UAV Path Planning Based on Multi-Layer Reinforcement Learning Technique. *IEEE Access, 9*, 59486–59497. https://doi.org/10.1109/access.2021.3073704

Çetin, Ö. (2015). Çoklu Otonom İnsansız Hava Araçları İçin Paralel Programlama Tabanlı Yol Planlaması, Doktora Tezi, Hava Harp Okulu Havacılık Ve Uzay Teknolojiler, Enstitüsü.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik, 1*(1), 269–271. https://doi.org/10.1007/bf01386390

Eriksson, U. (2018). Dynamic Path Planning for Autonomous Unmanned Aerial Vehicles, *Degree Project In Computer Science And Engineering*, Second Cycle, 30 Credits Stockholm, Sweden.

Feng, J., Zhang, J., Zhang, G., Xie, S., Ding, Y., & Liu, Z. (2021). UAV Dynamic Path Planning Based on Obstacle Position Prediction in an Unknown Environment. *IEEE Access, 9*, 154679–154691. https://doi.org/10.1109/access.2021.3128295

Goerzen, C., Kong, Z., & Mettler, B. (2009). A Survey of Motion Planning Algorithms from the Perspective of Autonomous UAV Guidance. *Journal of Intelligent and Robotic Systems, 57*(1–4), 65–100. https://doi.org/10.1007/s10846-009-9383-1

Gültekin, A., Diri, S. & Becerikli, Y. (2023). Simplified and Smoothed Rapidly-Exploring Random Tree Algorithm for Robot Path Planning. *Tehnički vjesnik, 30*(3), 891-898. https://doi.org/10.17559/TV-20221015080721.

Hart, P., Nilsson, N., & 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. https://doi.org/10.1109/tssc.1968.300136

Holland, J.H. (1975). Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence, *University Michigan Press*, Ann Arbor, MI, USA, 1975.

Jamshidi, V., Nekoukar, V., & Hossein Refan, M. (2022). Implementation of UAV Smooth Path Planning by Improved Parallel Genetic Algorithm on Controller Area Network. *Journal of Aerospace Engineering, 35*(2). https://doi.org/10.1061/(asce)as.1943-5525.0001395

Karaman, S., & Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research, 30*(7), 846–894. https://doi.org/10.1177/0278364911406761

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. Proceedings of ICNN'95 - *International Conference on Neural Networks.* https://doi.org/10.1109/icnn.1995.488968

Khatib, O. (1986). Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. *The International Journal of Robotics Research*, *5*(1), 90–98. https://doi.org/10.1177/027836498600500106

Koşan, M. A., Coşkun, A., & Karacan, H. (2019). Yapay Zekâ Yöntemlerinde Entropi. *Journal of Information Systems and Management Research* (C. 1, Sayı 1, ss. 15-22). M. Hanefi CALP.

LaValle, S. M., (1998). Rapidly exploring random trees: A new tool for path planning, (PDF). *Technical Report.* Computer Science Department, Iowa State University,1998, (TR 98–11).

Li, C., Si, Q., Zhao, J., & Qin, P. (2023). A robot path planning method using improved Harris Hawks optimization algorithm. *Measurement and Control.* https://doi.org/10.1177/00202940231204424

Li, H., Lv, T., Shui, Y., Zhang, J., Zhang, H., Zhao, H., & Ma, S. (2023). An Improved grey wolf optimizer with weighting functions and its application to Unmanned Aerial Vehicles path planning. *Computers and Electrical Engineering, 111*, 108893. https://doi.org/10.1016/j.compeleceng.2023.108893

Liu, Y., & Xu, W. (2020). Research and Application of Path Planning Algorithm in Complex Terrain. *Journal of Physics: Conference Series, 1570*(1), 012008. https://doi.org/10.1088/1742-6596/1570/1/012008

Matlekovic, L., & Schneider-Kamp, P. (2023). Constraint Programming Approach to Coverage-Path Planning for Autonomous Multi-UAV Infrastructure Inspection. *Drones*, *7*(9), 563. https://doi.org/10.3390/drones7090563

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey Wolf Optimizer. *Advances in Engineering Software*, *69*, 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007

Özalp, N., Şahingöz, Ö. K., & Ayan, U. (2014). Autonomous Multi Unmanned Aerial Vehicles Path Planning On 3 Dimensional Terrain, *2014 IEEE 22nd Signal Processing and Communications Applications Conference (SIU)*

Qureshi, A. H., & Ayaz, Y. (2015). Potential functions based sampling heuristic for optimal path planning. *Autonomous Robots*, *40*(6), 1079–1093. https://doi.org/10.1007/s10514-015-9518-0

Rosenblatt, F. (1957). The perceptron A perceiving and recognizing automaton. *Technical Report 85-460-1, Cornell Aeronautical Laboratory*, Ithaca, New York, January .

Sharma, S., Sharma, S., & Athaiya, A. (2020). Activation Functions In Neural Networks. *International Journal of Engineering Applied Sciences and Technology*, *04*(12), 310-316. https://doi.org/10.33564/IJEAST.2020.V04I12.054

Shin, Y., & Kim, E. (2021). Hybrid path planning using positioning risk and artificial potential fields. *Aerospace Science and Technology*, *112*, 106640. https://doi.org/10.1016/j.ast.2021.106640

Sujit, P.B., & Beard, R. (2009). Multiple UAV path planning using anytime algorithms, *2009 American control conference*

URL-1. https://www.tensorflow.org/ (Ziyaret tarihi: 18 Ocak 2023)

URL-2. https://keras.io/ (Ziyaret tarihi: 21 Temmuz 2023)

URL-3. http://www.arducopter.co.uk/iris-quadcopter-uav.html (Ziyaret tarihi: 6 Temmuz 2023)

URL-4. http://ros.org (Ziyaret tarihi: 10 Mart 2022)

URL-5. http://gazebosim.org (Ziyaret tarihi: 15 Mart 2022)

URL-6. https://px4.io/ (Ziyaret tarihi: 25 Şubat 2023)

Wang, X., Luo, X., Han, B., Chen, Y., Liang, G., & Zheng, K. (2020). Collision-Free Path Planning Method for Robots Based on an Improved Rapidly-Exploring Random Tree Algorithm. *Applied Sciences*, *10*(4), 1381. https://doi.org/10.3390/app10041381

Wang, F., Chen, Z., Wu, C., & Yang, Y. (2019). Prediction on sound insulation properties of ultrafine glass wool mats with artificial neural networks. *Applied Acoustics*, *146*, 164–171. https://doi.org/10.1016/j.apacoust.2018.11.018

Wu, Y., Nie, M., Ma, X., Guo, Y., & Liu, X. (2023). Co-Evolutionary Algorithm-Based Multi-Unmanned Aerial Vehicle Cooperative Path Planning. *Drones*, *7*(10), 606. https://doi.org/10.3390/drones7100606

Wu, K., Wang, H., Esfahani, M. A., & Yuan, S. (2022). Achieving Real-Time Path Planning in Unknown Environments Through Deep Neural Networks. *IEEE Transactions on Intelligent Transportation Systems*, *23*(3), 2093–2102. https://doi.org/10.1109/tits.2020.3031962

Yang, L., Qi, J., Xiao, J., & Yong, X. (2014). A Literature Review of UAV 3D Path Planning*. *Proceeding of the 11th World Congress on Intelligent Control and Automation* Shenyang, China, June 29 - July 4 2014

Yang, J., Zhang, H., & Ning, P. (2023). Path Planning and Trajectory Optimization Based on Improved APF and Multi-Target. *IEEE Access*, *11*, 139121–139132. https://doi.org/10.1109/access.2023.3338683

Yoshida, E. (2005). Humanoid motion planning using multi-level DOF exploitation based on randomized method. *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. https://doi.org/10.1109/iros.2005.1544954

Zhang, R., Li, X., Ren, H., Ding, Y., Meng, Y., & Xia, Q. (2023). UAV Flight Path Planning Based on Multi-Strategy Improved White Sharks Optimization. *IEEE Access*, *11*, 88462–88475. https://doi.org/10.1109/access.2023.3304708

Zhang, Z., & Sabuncu, M. R. (2018). Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels. *Advances in Neural Information Processing Systems*, 2018-December, 8778-8788. https://arxiv.org/abs/1805.07836v4

Zhang, H., Xin, B., Dou, L. H., Chen, J., & Hirota, K. (2020). A review of cooperative path planning of an unmanned aerial vehicle group. *Frontiers of Information Technology & Electronic Engineering*, *21*(12), 1671–1694. https://doi.org/10.1631/fitee.2000228

Zhang, Z., & Sabuncu, M. R. (2018). Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels. *Advances in Neural Information Processing Systems*, 2018-December, 8778-8788. https://arxiv.org/abs/1805.07836v4

KİŞİSEL YAYIN VE ESERLER

Diri, S., **Gültekin,** A., Şahin, S. A., & Kayapınar, Ö. B. (2022). Classification of Server Network Traffic for DDoS Attacks Using MapReduce. *7th International Scientific Conference "Telecommunications, Informatics, Energy and Management"*, 139-142.

Gültekin, A., Becerikli, Y. (2024). A New UAV Path Planning Approach Based on Deep Neural Networks. *IEEE Access* (Hakem Sürecinde)

Gültekin, A., Diri, S. & Becerikli, Y. (2023). Simplified and Smoothed Rapidly-Exploring Random Tree Algorithm for Robot Path Planning. *Tehnički vjesnik, 30*(3), 891-898. https://doi.org/10.17559/TV-20221015080721.

Gültekin, A., Becerikli, Y. (2023). An Improved Hybrid Robot Path Planning Algorithm Based On Artificial Potential Field And RRT. *18th International European Conference On Mathematics, Engineering, Natural & Medical Sciences*

Gültekin, A., Diri, S., & Becerikli, Y. (2022). Çoklu İHA'nın Haberleşmesi ile Düşman Ortamının Matematiksel Modellemesi. *3rd International Conference on Applied Engineering and Natural Sciences ICAENS 2022*, 877-882.

Gültekin, A., Becerikli, Y. (2022). YPA ve RRT Temelli Hibrid İHA Yol Planlama Algoritması. *Ispec 14. Uluslararası Mühendislik ve Fen Bilimleri Kongresi 18-19 Temmuz 2022*

Gültekin, A., Diri, S., Altuntaş, F., & Yerlikaya, Z. (2018). Rol Tabanlı Erişim Kontrolü Kullanılarak Öğrenci Bilgi Sistemi Modellenmesi. *Uluslararası Marmara Fen ve Sosyal Bilimler Kongresi*, 23-25 Kasım, Kocaeli

Omurca, S, İ., Sayar, A., Şahin, S., Ardıç, S., Erdem, T., **Gültekin, A.,** Ekinci, E.,& Eken, S. (2020). Dijital Dokümanların Anahtar Kelime Tabanlı Doğrulanması. *6. Ulusal Yüksek Başarımlı Hesaplama Konferansı* / 8-9 Ekim 2020, Ankara Yıldırım Beyazıt Üniversitesi, Ankara

ÖZGEÇMİŞ

İlk ve orta öğrenimini Kayseri'de tamamladı. 2004 yılında Anadolu Üniversitesi İşletme Fakültesi İşletme Bölümünden mezun oldu. 2006 yılında İstanbul Teknik Üniversitesi Bilişim Enstitüsü Enformasyon Sistemlerinin Tasarımı ve Yönetimi Anabilim dalı yüksek lisans programından mezun oldu. 2013 yılında Kocaeli Üniversitesi Sosyal Bilimler Enstitüsü İşletme Anabilim dalı Yönetim ve Organizasyon yüksek lisans programından mezun oldu. 2014 yılında Sakarya Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümünden mezun oldu. Ardından Kocaeli Üniversitesi Bilgisayar Mühendisliği Anabilim Dalı'nda doktora eğitimine başladı. Doktora eğitimi sırasında başta robot yol planlama olmak üzere çeşitli alanlarda çalışmaları bulunmaktadır.

2000 - 2001 yılları arasında İstanbul Üniversitesi'nde programcı olarak, 2001 - 2005 yılları arasında Garanti Teknoloji'de sistem yönetimi biriminde, 2005 - 2008 yılları arasında Ulaştırma Bakanlığı'nda programcı olarak ve 2008 - 2015 yılları arasında Kocaeli Üniversite Bilgi İşlem Daire Başkanlığı'nda yazılımcı ve birim sorumlusu olarak ve 2016 yılından bu yana da Kocaeli Üniversitesi Öğrenci İşleri Daire Başkanlığı Otomasyon ve İstatistik Şube Müdür V. olarak çalışmaktadır. Evli ve iki kız çocuğu babasıdır.