T.C. İSTANBUL KÜLTÜR ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

KISITLARA BAĞLI MATEMATİKSEL MODELLEME İLE İNSANSIZ HAVA ARACI İÇİN YUMUŞATILMIŞ ROTA PLANLAMASI

DOKTORA TEZİ

Bayram Ali BURAN

Anabilim Dalı: Matematik-Bilgisayar

Programı: Matematik

MAYIS 2019

T.C. İSTANBUL KÜLTÜR ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

KISITLARA BAĞLI MATEMATİKSEL MODELLEME İLE İNSANSIZ HAVA ARACI İÇİN YUMUŞATILMIŞ ROTA PLANLAMASI

DOKTORA TEZİ

Bayram Ali BURAN

1209241002

Tezin Enstitüye Verildiği Tarih : 25 Haziran 2019
Tezin Savunulduğu Tarih : 28 Mayıs 2019

Tez Danışmanı:	Dr. Öğr. Üyesi Süleyman Hikmet ÇAĞLAR
Eş Danışmanı:	Doç. Dr. Özgür Koray ŞAHİNGÖZ
Diğer Jüri Üyeleri:	Doç. Dr. Coşkun GÜLER (YTÜ)
	Dr. Öğr. Üyesi Yaşar POLATOĞLU
	Prof. Dr. Remzi Tunç MISIRLIOĞLU
	Doç. Dr. Mustafa Emre AYDEMİR (İESÜ)

MAYIS 2019

ÖNSÖZ

Doktora eğitimim süresince gerek verdiği derslerde gerekse tezimin hazırlık aşamasında tüm bilgisini benimle paylaşan, sabırla her konuda desteğini esirgemeyen ve gerekli tüm kaynaklara ulaşmam için her zaman yardımcı olan danışmanım Dr. Öğr. Üyesi Süleyman Hikmet ÇAĞLAR'a, kendisinden aldığım öneriler ve yönlendirmeleri sayesinde tezime çok şey kattığına inandığım eş danışmanım Doç. Dr. Özgür Koray ŞAHİNGÖZ'e, destek ve güvenleriyle her zaman yanımda olan eşim Hülya BURAN'a, kızım Leyla Merve BURAN'a ve oğlum Asım Emre BURAN'a sonsuz teşekkürlerimi sunarım.

Mayıs 2019 Bayram Ali BURAN

Üniversitesi	:	**İstanbul Kültür Üniversitesi**
Enstitüsü	:	**Lisanüstü Eğitim Enstitüsü**
Anabilim Dalı	:	**Matematik-Bilgisayar**
Programı	:	**Matematik**
Tez Danışmanı	:	**Dr. Öğr. Üyesi Süleyman Hikmet ÇAĞLAR**
Eş Danışmanı	:	**Doç. Dr. Özgür Koray ŞAHİNGÖZ**
Tez Türü ve Tarihi	:	**Doktora - Mayıs 2019**

ÖZET

KISITLARA BAĞLI MATEMATİKSEL MODELLEME İLE İNSANSIZ HAVA ARACI İÇİN YUMUŞATILMIŞ ROTA PLANLAMASI

Bayram Ali BURAN

Bu tezde, İnsansız Hava Aracı (İHA) için matematiksel rota planlama yöntemleri incelenmiştir. Bir İHA'nın önceden belirlenen kontrol noktalarını ziyaret etmesi ve yeniden başlangıç noktasına geri dönülmesi problemi İHA'nın hareket kriterlerine bağlı olarak çözülmüştür. Problem iki aşamada ele alınmıştır. Birinci aşama; düzensiz olarak verilen kontrol noktalarının hangi sıra ile ziyaret edileceğinin belirlenmesidir. Gezgin Satıcı Problemi (Traveling Salesman Problem-TSP) olarak da bilinen bu problem NP-Hard olarak tanımlanmıştır. TSP için optimal çözüme yakın sonuç veren bir evrimsel algoritma olan Genetik Algoritma (GA) yöntemi kullanılmıştır. Bu çözümde bir İHA'nın kontrol etmesi gereken çok sayıda kontrol noktası bulunmaktadır ve GA bu kontrol noktalarının hangi sıra ile dolaşılacağını belirlemektedir. Bu dolaşı sıralamasının belirlenmesi ile oluşan yol keskin dönüşler içermektedir. İHA'nın manevra kabiliyeti göz önünde bulundurularak çözümün ikinci aşamasında yumuşatma işlemi uygulanmıştır.

Yumuşatma yöntemleri olarak Bezier Eğrileri, B-Spline Eğrileri ve Dubins Yolu kullanımı iki örnek problem üzerinde incelenmiş ve elde edilen sonuçlar karşılaştırılmıştır. Yumuşatma seviyesinin iyileştirilmesi ve İHA'nın gerçekçi uçuş rotasına benzetilebilmesi için Bezier Eğrileri kullanılırken ortaya çıkan pürüzlü kısımlar sanal kontrol noktaları eklenerek giderilmiş ve rasyonel katsayılı Bezier Eğrileri kullanılarak eğrinin üzerinden geçmediği kontrol noktalarına yaklaşması sağlanmıştır.

Kuadratik (2'nci derece) Bezier Eğrileri ve Dubins Yolu yöntemleri ile oluşturulan rotalar tüm kontrol noktalarının tam olarak üzerinden geçmektedir. Bu durumda oluşan rota diğer yöntemlere göre daha uzun olmaktadır. Ancak görev planındaki amaca göre bu şekilde İHA uçuşunun da tercih edilebileceği öngörülmektedir. Kübik (3'üncü derece) ve Kuartik (4'üncü derece) Bezier Eğrileri ile oluşturulan rotalar kontrol noktalarının bir kısmının üzerinden geçmekte diğerlerine ise yakınsamaktadır.

Kuadratik, Kübik ve Kuartik B-Spline Eğrileri ile oluşturulan rotalar ise bir kaçı hariç kontrol noktalarının neredeyse hiç birinin üzerinden geçmemektedir. Ancak oluşturulan rotanın uzunluğu daha küçüktür.

Bezier ve B-Spline Eğrilerinde derece arttıkça oluşturulan rota kısalmakta fakat kontrol noktalarının eğriye uzaklıklarının ortalaması artmaktadır.

Oluşturulan yolların avantaj ve dezavantajları belirtilmiş olup, tercih için İHA'nın görev tanımının belirleyici olacağı sonucuna ulaşılmıştır.

Anahtar Kelimeler : Genetik Algoritma, İHA Rota Planlama, Rota Yumuşatma, Bezier Eğrileri, B-Spline Eğrileri, Dubins Yolu.

University	:	**İstanbul Kültür University**
Institute	:	**Institute of Graduate Studies**
Science Programme	:	**Mathematics and Computer Science**
Programme	:	**Mathematics**
Supervisor	:	**Asst. Prof. Dr. Süleyman Hikmet ÇAĞLAR**
Co-Supervisor	:	**Assoc. Prof. Dr. Özgür Koray ŞAHİNGÖZ**
Degree Awarded and Date	:	**PhD - May 2019**

ABSTRACT

SMOOTHED PATH PLANNING FOR UNMANNED AERIAL VEHICLE WITH MATHEMATICAL MODELING BASED ON CONSTRAINTS

Bayram Ali BURAN

In this thesis, mathematical path planning methods for Unmanned Air Vehicle (UAV) are examined. The problem of visiting the determined control points and returning to the starting point has been solved. The problem is discussed in two stages. The first stage; this is to determine the order in which the control points that are given without order are to be visited. This problem, known as Traveling Salesman Problem-TSP, is defined as NP-Hard. As for the Traveling Salesman Problem the Genetic Algorithm (GA) method, an evolutionary algorithm that gives results close to the optimal solution, is used. In this solution, there are a large number of control points that are needed to be checked by a UAV and the GA determines the order in which these checkpoints are to be navigated. The path formed by the determination of this tide sequence contains sharp turns. In the second stage of the solution, the smoothing process was applied considering the maneuverability of the UAV.

As the smoothing methods, Bezier Curves, B-Spline Curves and the

use of Dubins Path were examined on two sample problems and the results were compared. In order to improve the smoothing level and simulate the realistic flight route of the UAV, the rough parts that occur when using Bezier Curves are eliminated by adding imaginary control points and approached to the control points where the curve does not pass by using Bezier Curves with rational coefficients.

The paths that are formed by the Quadratic (2nd degree) Bezier Curves and Dubins Path pass precisely all over the control points. The path formed thus is longer than the other methods. However, according to the purpose in the task plan, it is predicted that UAV flights may be preferred as well. The routes generated by the Cubic (3rd degree) and Quartic (4th degree) Bezier Curves pass over some of the control points while converging the others.

The paths that are formed by the Quadratic, Cubic and Quartic B-Spline Curves pass over almost none of the control points except for the few of them. However, the length of the route formed is smaller. The route formed in Bezier and B-Spline Curves is shortened as the degree increases while the average distance between control points and curve increases.

The advantages and disadvantages of the paths formed were determined and it was concluded that the definition of duty of the UAV would be the determinant.

Keywords : Genetic Algorithm, UAV Path Planning,
Path Smoothing, Bezier Curves,
B-Spline Curves, Dubins Path.

İÇİNDEKİLER

Kısaltmalar

İHA	:İnsansız Hava Aracı
SİHA	:Silahlı İnsansız Hava Aracı
TSP	:Traveling Salesman Problem (Gezgin Satıcı Problemi)
NP	:Non Polnomial (Polinomal Olmayan)
GA	:Genetic Algorithm (Genetik Algoritma)
SK-İHA	:Sabit Kanatlı İHA
DK-İHA	:Döner Kanatlı İHA
VTOL	:Vertical Take off/Landing (Dikey kalkış/iniş)
STOL	:Short Take off/Landing (Kısa kalkış/iniş)
MALE	:Medium Altitude Long Endurance (Orta İrtifa ve Uzun Menzil)
HALE	:High Altitude Long Endurance (Yüksek İrtifa ve Uzun Menzil)
MAM	:Mini Akıllı Mühimmat
UMTAS	:Uzun Menzilli Tank Savar
FOA	:Fruit Fly Optimization Algorithm (Meyve Sineği Optimizasyon Algoritması)
θ-MAFOA	:θ-Mutation Adaptation Fruit Fly Optimization Algorithm (θ-Mutasyon Adaptasyon Meyve Sineği Optimizasyon Algoritması)
BA	:Bat Algorithm (Yarasa Algoritması)
DE	:Differential Evolution (Diferansiyel Evrim)
IBA	:Improved version of Bat Algorithm (Geliştirilmiş Yarasa Algoritması)
MGA	:Modified Genetic Algorithm (Modifiye Genetik Algoritma)
DGA	:Distributed Genetic Algorithm (Dağıtılmış Genetik Algoritma)
DOF	:Degrees of Freedom (Serbestlik Derecesi)
MAT	:Moving Air Target (Hareketli Hava Hedefi)
RHC	:Receding Horizon Control (Azalan Ufuk Kontrolü)
WPS	:Wolf Pack Search (Kurt Paketi Arama)

T-S	:Temporal-Spatial (Zamansal-Mekansal)
AUV	:Autonomous Underwater Vehicle (Otonom Sualtı Aracı)
PSO	:Particle Swarm Optimization (Parçacık Sürüsü Optimizasyonu)
DMS-PSO	:Dynamic Multiswarm Particle Swarm Optimization (Dinamik Çoklu Sürü Parçacık Sürüsü Optimizasyonu)
DDW	:Differential Drive Wheeled (Diferansiyel Sürüş Tekerlekli)
PID	:Proportional Integral Derivative (Oransal İntegral Türev)
İKA	:İnsansız Kara Aracı
DDTSPN	:Dynamic Dubins Traveling Salesman Problem with Neighborhood (Komşularla Dinamik Dubins Gezgin Satıcı Problemi)
OP	:Orienteering Problem (Oryantiring Problemi)
DA-OP	:Dubins Airplane Orienteering Problem (Dubins Uçak Oryantiring Problemi)
RVNS	:Randomized Variable Neighborhood Search (Rastgele Değişken Komşuluk Arama)
CS	:Cuckoo Search (Guguk Kuşu Arama)
RRT	:Rapidly Random Tree (Hızlı Rastgele Ağaç)
UAS	:Unmanned Aircraft Systems (İnsansız Hava Aracı Sistemleri)
DTSP	:Dubins TSP
SAMSOA	:Distributed Search–Attack Mission Self-Organization Algorithm (Dağıtılmış Arama-Saldırı Görevi Kendinden Organizasyon Algoritması)
ACO	:Ant Colony Optimization (Karınca Kolonisi Optimizasyonu)
DEA	:Differential Evolution Algorithm (Diferansiyel Evrim Algoritması)
PSO	:Partical Swarm Optimizition (Parçacık Sürü Optimizasyonu)
ABC	:Artificial Bee Colony (Yapay Arı Kolonisi)
mGA	:Modified Genetic Algorithm (Geliştirilmiş Genetik Algoritma)
GPU	:Graphical Processing Units
EYK	:En Yakın Komşu
CAGD	:Computer Aided Geometric Design (Bilgisayar Destekli Geometrik Tasarım)
C	:Circle
S	:Straight
L	:Left
R	:Right

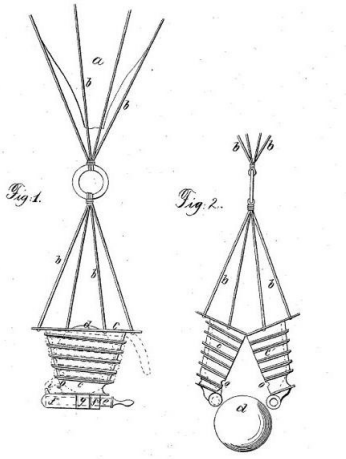
Şekil Listesi

Tablo Listesi

BÖLÜM 1

GİRİŞ

İnsansız Hava Araçları (İHA)'nın kullanımı; genellikle insanlar için risk teşkil eden ortamlarda ve insanların fiziki koşullarına dayanamayacağı ve veriminin düşeceği uzun süreli görevlerde, insanı tehlikeden uzak tutmak adına 18.Yüzyılda başlamış ve gelişerek günümüze kadar devam etmiştir. Bir İHA'nın kaydedilmiş ilk kullanımı, Avusturyalıların, patlayıcı madde yüklü insansız balonları kullanarak İtalya'nın Venedik kentine saldırdıkları 1849 yılı olarak gösterilmektedir [1]. Amerika İç Savaşı sırasında Charles Perley tarafından 24 Şubat 1863 tarihli patentinde balonla silah taşımak için; Şekil 1.1'de gösterilen zamanlanmış bir sigortanın süresi dolduğunda kapak gibi açıldığı ve böylelikle bir bomba çıkacak olan "bölünmüş sepet" başvurusu yapılmıştır [2].



Şekil 1.1: Charles Perley'in Balonla Zamanlanmış Bomba Patenti 1863

1883 yılına gelindiğinde ilk havadan fotoğraf, bir uçurtma ve ona bağlı bir fotoğraf makinesi kullanılarak gerçekleştirildi. Daha sonra bu yöntem 1898 yılında İspanyol-Amerikan savaşında ilk askeri havadan keşif fotoğrafı olarak kullanıldı [3]. Birleşik Devletler tarafından 1918 yılında cayroskoplar, barometre ve bir mekanik bilgisayar tarafından yönlendirilen, ilk otonom İHA olarak da adlandırabileceğimiz "Kettering Bug" geliştirildi [2]. 1937 yılına gelindiğinde radyo kontrollü uçak olan Curtiss N2C-2 uçağı ABD Donanması tarafından geliştirildi [1]. 1935 yılında I. Dünya savaşında pilot olarak görev almış aktör Reginald Denny ortağı ile birlikte kurduğu şirketinde Radioplane OQ-2 adlı uzaktan kumandalı uçağı üretti [2].

II. Dünya savaşının hemen sonrasında jet motorları ilk kez Teledyne Ryan Firebee-I modeline uygulandı. 1955'te Amerikan Deniz Kuvvetleri için Beechcraft tarafından 1001 modeli geliştirildi. Amerika tarafından başlatılan gizli İHA projesi "Red Vagon" kapsamında geliştirilen modern İHA'lar Ağustos 1964'te Kuzey Vietnam'a karşı ilk kez kullanıldı. 1973'te Yom Kippur Savaşı esnasında Mısır ve Suriye tarafından kullanılan yerden havaya füzeleri İsrail savaş uçaklarına ağır hasar verdikten sonra İsrail gerçek zamanlı gözlem yapan ilk İHA'yı geliştirdi. Bu İHA'nın sağladığı görüntüler ile radarların tespit edilmesi sağlandı ve 1982 yılında Lübnan Savaşının başlangıcında Suriye hava savunmalarını hiç pilot kaybetmeden etkisiz hale getirmesine yardımcı oldu. 1987 yılına gelindiğinde İsrail sınırsız yetenekleri olan, gizli tabanlı, üç boyutlu vektörsel uçuş kontrolü olan jet yönlendirmeli İHA'ları ilk kez geliştirdi [4]. 1986'da ABD ve İsrail ortaklığı ile RQ2 Pioneer İHA'sı geliştirildi. İHA'ya olan ilgi 1980-1990 yıllarında arttı. Çoğunlukla İHA'lar gözlem amaçlı kullanılırken bazıları ise mühimmat taşımaya başladı. Muhtemelen en tanınmış İHA olan RQ-1 Predator 1994 yılında geliştirildi. Predator Hava-Yer mühimmatı kullanabilen bir İHA sistemidir [2]. 1990 yılından itibaren daha küçük boyutlu, omuzdan atılabilen, İHA'lar geliştirildi.

İHA'ya olan ilgi, düşük maliyeti ve mürettebat kaybı riskinin bulunmaması nedeniyle gün geçtikçe artmaktadır. Keşif, arama, yangın kontrolü, haberleşme gibi sivil alanlarda kullanımın yanısıra günümüzde silahlı İHA teknoljisi askeri

operasyonlarda yaygın olarak kullanılmakta ve uzun vadede bu teknolojinin insanlı uçakların yerini alması planlanmaktadır. Ülkemiz de az sayıdaki İHA teknolojisi üreten ülkeler sınıfında yer almaktadır. Silahlı İnsansız Hava Aracı (SİHA) ülkemizde askeri ve yarı askeri kuruluşlar tarafından üretilmekte ve kullanılmaktadır. İHA'lar bir merkezden uzaktan kontrol edilmekte veya otonom olarak görev icra edebilmektedir. İHA'ların hareket kabiliyeti havada kalış süreleri, mürettebat riske etmemesi, düşük maliyeti, radar görünürlüğünün düşük olması vb. nedenlerle daha çok tercih edilmektedir.

1.1 İHA'nın Hareketinin İncelenmesi

Bilim ve teknolojideki ilerlemeler ile birlikte karada, denizde ve havada icra edilecek çeşitli görevlerde kullanılmak üzere birçok farklı araç geliştirilmiştir. Geliştirilen araçlar hareket kabiliyetine göre; Holonomik hareket edenler ve Non-Holonomik hareket edenler olarak sınıflandırılmaktadır. Basitçe açıklayacak olursak;

Holonomik Hareket: Bir önceki konumundan bağımsız bir şekilde bir sonraki konumuna hareket edebilen, yani her yöne hareket edebilen, araçlardır. Ör: Drone

Non-Holonomik Hareket: Bir sonraki konumuna hareket etmek için bir önceki konumunun yeri ve yönünün önemli olduğu, sadece ileri yönlü hareketin veya ileri-geri yönlü hareketin mümkün olduğu araçlardır. Ör: Otomobil, Denizaltı, İHA vb.

Bu çalışmada özel olarak Non-Holonomik (Holonomik olmayan) hareket edebilen bir araç olan İHA'lar ele alınmıştır. İHA'nın Non-Holonomik hareket yapması bizim için önemli bir husustur. İlerleyen kısımlarda bu konu daha ayrıntılı olarak anlatılacaktır. İHA'lar uzaktan kumanda ile yönlendirilen veya otonom olarak hareket edebilen, aracın yapısına göre dizel, benzinli veya elektrikli motorları sayesinde ileri yönde itki gücü üreten, gövdesinde taşıyabildiği maksimum faydalı yük miktarı, havada kalış süresi ve menziline göre sınıflandırılan hava araçlarıdır. İHA denildiğinde akla ilk olarak askeri

kullanım alanları gelse de sivil alanlarda da giderek daha yaygın olarak kullanılmaktadır. Askeri alanda keşif, gözetleme, istihbarat, lazer ile diğer savaş uçakları için hedef işaretleme, havadan havaya ve havadan yere mühimmat ile hedefe yönelik atış yapmakta kullanılmaktadır. Sivil alanda ise gözetleme, taşımacılık, hava tahmini, harita oluşturma, belgesel çekimi, doğal afetlerin izlenmesi, deniz kirliliğinin gözlemlenmesi vb. her geçen gün sayısı artan pek çok kullanım alanına sahiptir.

1.2 Rota Planlama

Rota planlama problemi olarak adlandırdığımız problem, seyahat etmekte olan bir İHA'nın bir başlangıç noktasından başlayıp, her noktayı sadece bir kez ziyaret etme koşulu ile arada kalan noktaların en kısa yol ile ziyaret edilip tekrar başlangıç noktasına dönülmesi şeklinde tanımlanmıştır. Literatürde bu tip problem Gezgin Satıcı Problemi (TSP-Traveling Salesman Problem) olarakta geçmektedir. TSP ile ilgili matematiksel problemler 1800'lerde İrlandalı matematikçi Sir William Rowan Hamilton ve İngiliz matematikçi Thomas Penyngton Kirkman tarafından işlenmiştir. Şekil 1.2'de, sadece belirtilen bağlantıları kullanarak oyuncuların 20 puan boyunca tur atmalarını gerektiren Hamilton Icosian Game'in fotoğrafı görülmektedir [5].

TSP'nin genel formu ilk kez Karl Menger tarafından 1930'da önerilmiştir [6]. Büyük Ölçekli bir TSP'nin çözümü ilk kez 49 şehirde Dantzig, Fulkerson ve Johnson tarafından 1954 yılında uygulanmıştır [7]. Ziyaret edilmesi planlanan şehir sayısı N olmak üzere, N olasılıktan biri olarak seçilen başlangıç noktasından gidilebilecek (N-1) farklı şehir bulunmaktadır. Her adımda bir azalacak olan olasılıkların sonucunda N! farklı olasılık olacağı açıktır. Şehir sayısı artırıldığında bu sayı çok büyük olacaktır. Örneğin 10 şehir olması durumunda 10!=3,628,800 farklı yol kombinasyonu vardır. Tüm yol uzunluklarının hesaplanması ve en kısa yol uzunluğunun belirlenmesi oldukça zahmetli ve zaman alacak bir iştir. Bu nedenle, TSP'nin Non Polnomial-Hard (NP-Hard) bir problem olduğu Karp tarafından 1972 yılında tanımlanmıştır [8].

Şekil 1.2: Hamilton Icosian Game

TSP başlangıçta kara araçları ele alınarak çalışılmıştır [9]. Ancak daha sonrasında deniz ve hava araçlarının rota planlamasında da uygulanmaya başlanmıştır. TSP tek bir araç için planlanabileceği gibi birden fazla aracın rota planlamasının yapılacağı çoklu görevler için de kullanılabilmektedir [10]. Planlanacak rotanın hangi araç için olduğu, aracın hareket kabiliyeti, menzili, maksimum yük kapasitesi gibi hususlar açısından kullanılacak yöntemler önem arz etmektedir.

Bu çalışmada özel olarak İHA'ların görevleri kapsamında izlemeleri istenen rota problemi göz önünde bulundurulmaktadır. İHA'nın görevi dahilinde kontrol noktasına belli bir uzaklıktan geçerek (veya tam olarak kontrol noktasının üzerinden geçerek) kontrol noktalarının topladığı bilgileri elde etmesi ve kontrol merkezine iletmesi istenmektedir. Burada İHA bir veri toplama arabirimi olarak çalışabileceği gibi, kontrol edilmesi gereken bir sınır bölgesinin veya bir askeri birliğin yada nükleer santralin çevresinin görüntü temelli fiziki kontrolünün sağlanması amaçlarıyla da kullanılabilir. Görev dahilinde oluşabilecek kısıtlar minimize edilerek problem ele alınmıştır. Mesela 3 boyutlu ortamdaki dağ, tepe,

binalar, ağaçlar gibi engeller ile İHA'nın kalkış ve iniş işlemleri gözönüne alınmamıştır. İHA'nın sabit bir yükseklikten uçması yeterli bulunmuş iki boyutlu bir rota planı çözüm olarak kabul edilmiştir.

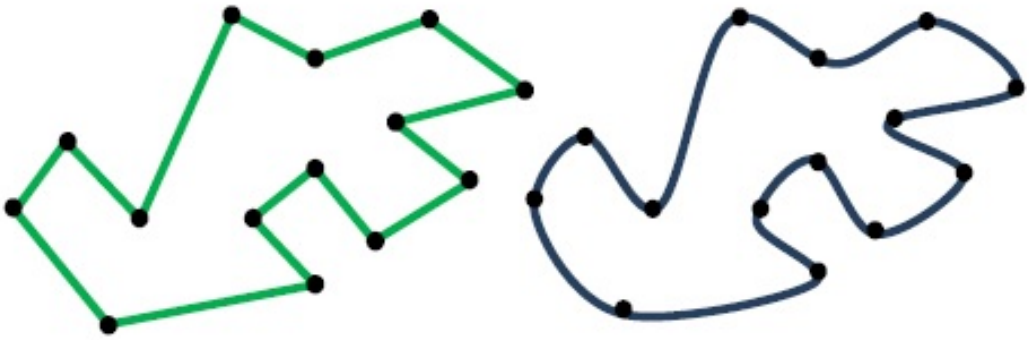
1.3 Evrimsel Algoritmalar

İHA'ların görev icrası kapsamında; belli amaçlar doğrultusunda belirlenmiş olan hedef/kontrol noktalarının, rota planlamasında göz önünde bulundurulması gerekmektedir. Kontrol noktalarının sayısı planlanan göreve göre değişebilmekle birlikte bu sayı bazen binleri bulabilmektedir. Kontrol noktalarının sayısının artması ile birlikte noktaların hangi sıra ile ziyaret edileceği zaman ve toplam yol açısından bir optimizasyon problemine dönüşmektedir. Bu problemin çözümü bizim İHA için yapacağımız rota planlamasının birinci adımı olan noktaların hangi sıra ile ziyaret edileceğini bulmamızı sağlayacaktır. TSP probleminin çözümü optimizasyon alanında birçok çalışmada ele alınmış olup, her geçen gün yeni yöntemler geliştirilen bir alandır [11–14].

Bu konuda yaptığımız literatür araştırmamızda daha yaygın olarak kullanılan ve birçok araştırmada kabul gören yöntemin Evrimsel Algoritmalar olduğu görülmektedir [15, 16]. TSP'nin Evrimsel Algoritmalar tarafından çözümü bazı dezavantajları da barındırmaktadır. İncelenen yöntemler arasında daha etkili ve uygulanabilir olan çözümün Genetik Algoritma tarafından sağlandığı görülmüştür [17, 18]. Genetik Algoritma yöntemi ile TSP'nin çözümü birçok çalışmada ele alınmış ve geliştirilmiştir [19–23]. Bu çalışmada Genetik Algoritma, TSP çözümü için bir araç olarak kullanılmaktadır. Problemin TSP olarak çözümü ile rota planlamasında kullanacağımız kontrol noktaların hangi sıra ile ziyaret edileceği belirlenmiş olacaktır.

1.4 Yumuşatma Teknikleri

Kontrol noktalarının ziyaret sırasının belirlenmesinin ardından noktaların birleştirilmesi ile keskin dönüşlerden dolayı İHA'nın hareket kabiliyetinde olmayan yani uçulabilir olmayan bir rota karşımıza çıkmaktadır. Bu rotanın yumuşatılması (smoothing) ile rota gerçek dünyadaki harekete daha benzer durum almaktadır. İHA rotalarındaki yumuşatma etkisi Şekil 1.3 de görülmektedir [24]. Yumuşatma işlemi rota planlaması probleminin ikinci kısmı olarak adlandırılmaktadır. Bu konu üzerinde yaptığımız araştırmalarda en yaygın olarak kullanılan yumuşatma işlemlerinin Bezier Eğrileri, B-Spline Eğrileri ve Dubins Yolu yöntemleri kullanılarak elde edildiği görülmektedir [25–31].



Şekil 1.3: İHA Rotaları

1.5 Tezin Amacı

Bu çalışmada; İHA'nın kısıtlara bağlı olarak rota planlama probleminin matematiksel yöntemler kullanılarak çözülmesi amaçlanmıştır.
Problem iki aşamadan oluşmaktadır:

Birinci aşama kontrol noktalarının hangi sıra ile ziyaret edileceğidir.

İkinci aşama ise ziyaret sırası belirlenen kontrol noktalarının yumuşatılmasıdır.

Problemin birinci aşaması için amaç verilen kontrol notalarının en kısa dolaşı sırasının Genetik Algoritma kullanılarak belirlenmesidir.

Problemin ikinci aşamasında ise için ziyaret sırası belirlenen, ancak İHA tarafından uçulabilir olmayan rotaya matematiksel yumuşatma yöntemleri uygulamaktır. Rotanın yumuşatılması sırasında; farklı matematiksel yöntemlerin kullanılması ve karşılaştırmalı olarak en uygun sonucun belirlenmesi amaçlanmaktadır. İHA rota planlamasında karşılaşılan en popüler yumuşatma yöntemleri olan Bezier Eğrileri, B-Spline Eğrileri ve Dubins Path uygulnacak yöntemlerdir. Çalışma, yapılan literatür araştırmasında rota planlaması için en çok kullanılan üç yöntemin aynı örnek problem üzerinde uygulanması ve çıkan sonuçların karşılaştırılması açısından önem arz etmektedir. Bezier Eğrilerinin birleştirilmesinde karşılaşılan rotalar sanal noktalar eklenerek C^1 sürekli hale getirilecektir. 2'nci derece-Kuadratik, 3'üncü derece-Kübik ve 4'üncü derece-Kuartik Bezier Eğrileri kullanılacaktır. Eklenen sanal noktanın eğriye olan etkisi üzerine çalışma yapılacaktır. Rasyonel Bezier Eğrileri de incelenerek ve ziyaret edilmeyen kontrol noktalarına eğrinin yakınsaması sağlanacaktır. Benzer şekilde Kuadratik, Kübik ve Kuartik B-Spline Eğrileri aynı problemlere uygulanacaktır. Gerek Bezier Eğrileri gerekse B-Spline Eğrileri tam anlamıyla interpolasyon eğrileri olmadığı için oluşturulacak eğri kontrol noktalarının hepsinden geçmeyecektir. Bu kapsamda oluşturulan eğrilerin üzerinden geçmediği kontrol noktalarına olan uzaklığının ortalaması maliyet fonksiyonu olarak değerlendirilerek, elde edilen sonuçlar karşılaştırılacaktır. Dubins Yolu belirtilen tüm kontrol noktalarından geçmektedir. Bu durumda oluşacak eğri uzunluğu diğer yöntemlere göre daha büyük olacaktır. Oluşturulan eğrilerin toplam uzunlukları, olumlu ve olumsuz yönleri karşılaştırılacaktır. Elde edilen sonuçların sonraki çalışmalara ışık tutacağı değerlendirilmektedir.

BÖLÜM 2

İLGİLİ KONULAR

2.1 İnsansız Hava Araçları

2.1.1 İHA Türleri

Son yıllarda gelişen teknoloji ile birlikte İHA kavramı üzerinde en çok konuşulan konulardan biri haline gelmiştir. Genellikle robot olarak adlandırılan insansız makine teknolojisi her geçen gün farklı bir alanda kullanıma sunulmaktadır. Kumandalı kontrol sistemleri ile kullanıma başlayan teknoloji adım adım otomatikleşme alanında bir tür evrim süreci geçirmektedir. Bu süreçte gelinen seviye; kendi kendisini kontrol edebilen otonom robotların yaygın olarak kullanıma başlamasını sağlamıştır.

Uygulama alanlarının gereksinimlerine göre kullanılacak teknolojinin çeşidinin belirlenmesi gerekmektedir. Havadan yürütülmesi gereken görevler için kullanılan İHA'ların birçok çeşidi ve kullanım amacı bulunmaktadır. Örneğin, uzun süre havada kalmasını gerektiren bir görev için planlama yapılırken bu durum göz önünde bulundurularak, görevi icra edebilecek yeterli havada kalış süresine sahip bir İHA tercih edilmelidir.

İHA'lar kanat yapısına göre Sabit Kanatlı İHA (SK-İHA) ve Döner Kanatlı İHA (DK-İHA) olarak ikiye ayrılmaktadır.

Sabit Kanatlı İHA

SK-İHA'lar önceden belirlenmiş, uçağın kalkış gücüne destek olmak için tasarlanmış eğimli yüzeylere sahip olan ve İHA'nın ileri yönde itkisinin, hava hızının neden olduğu kaldırma kuvvetinin üretilmesiyle uçuş yapabilen katı bir kanattan oluşurlar. Bu hava hızı, genellikle bir içten yanmalı motor veya elektrik motoru tarafından döndürülen bir pervane tarafından elde edilen ileri itme ile üretilir [32].

DK-İHA ile karşılaştırıldığında SK-İHA'nın temel avantajı, çok daha basit bir yapıdan oluşmasıdır. Daha basit yapıdan oluşması sonucunda, daha az karmaşık bir bakım ve onarım işlemi gerektirir. Bunun sonucunda kullanıcıya daha düşük bir maliyetle daha fazla havada kalış süresi sağlar. Ayrıca, daha yüksek hızlarda daha uzun uçuş süresi avantajını sağlayan daha verimli bir aerodinamik yapı sağlar, böylece görevlendirilen her uçuş başına daha geniş araştırma alanlarına sahip olur. SK-İHA'ların bir başka önemli avantajı da enerjinin olmadığı durumlarda hava da süzülme yetenekleridir.

SK-İHA'lar daha az güçle daha büyük faydalı yükleri daha uzun mesafeler için taşıyabilmektedir. Ancak SK-İHA'ların önemli bir dezavantajı iniş ve kalkış için bir pist veya fırlatıcıya ihtiyaç duyulmasıdır. Bu dezavantajı ortadan kaldırmaya yönelik VTOL-Vertical Take off/Landing (Dikey kalkış/iniş) ve STOL-Short Take off/Landing (Kısa kalkış/iniş) çözümleri kullanılmaktadır. Ayrıca SK-İHA'lar kaldırma gücünü üretmek için kanatlarının üzerinde hareket eden havaya ihtiyaç duyarlar, bu nedenle sürekli olarak ileri yönde hareket halinde olmaları gerekir. Bir DK-İHA gibi havada sabit kalamayacağı anlamına gelir. Bu da, planlanan görevin gereksinimlerine göre bazı görevler için uygun olmayacaklarını göstermektedir.

Döner Kanatlı İHA

DK-İHA'lar sabit bir mil etrafında dönen iki veya üç pervane bıçağından oluşur, bu yapı pervane olarak bilinir. DK-İHA'lar en az bir pervaneli (helikopter), üç pervaneli (trikopter), dört pervaneli (kuadkopter), altı pervaneli (heksakopter),

sekiz pervaneli (oktokopter) olmak üzere ve pervanelerin daha sıra dışı kullanımları ve kurulumlarından oluşan geniş bir kurulum yelpazesine sahiplerdir. Bu kurulumlar planlayıcının hedeflediği amaç doğrultusunda farklı şekillerde oluşturulabilir. Örneğin, bir Y6 kurulumunda her bir kolunda biri aşağı bakan diğeri yukarı bakan ikişer pervaneli bir trikopterdir. Her kurulumun kendine göre avantajları ve dezavantajları vardır [32].

Pervane bıçakları bir sabit kanatla aynı şekilde çalışır, ancak kanatların üzerinden hava akımı üretmek için sürekli ileri yönde bir uçak hareketine ihtiyaç duyulmaz, bunun yerine pervanelerin kendileri, kaldırma kuvveti oluşturmak için pervanelerin üzerinde gerekli hava akışını üreten sabit bir hareket halindedir. DK-İHA'ların kontrolü, itme ve torkdaki pervanelerin değişikliklerinden gelir. Örneğin; bir kuadkopterin aşağı inişi, arkadaki pervanelerin öndeki pervanelerden daha fazla itiş üretmesi ile sağlanır. Sapma hareketi; çapraz pervanelerin diğer çapraz pervanelere göre daha fazla tork, dönme gücü, üretmesi ile kuadkopterin dikey eksen üzerinde dönmesine neden olan bir dengesizlik oluşması ile sağlanır.

DK-İHA'ların en önemli avantajı dikey olarak iniş kalkış yapabilmeleridir. Bu avantaj sayesinde küçük bir alan kullanılarak iniş/kalkış yapılabilir. DK-İHA'ların holonomik hareket etmesini sağlayan özellikleri sayesinde çevik manevra kabiliyetine sahiptirler. Bir noktada sabit kalabilme özellikleri sayesinde denetleme ve kontrol görevleri için uygundurlar. Ancak diğer yandan DK-İHA'lar daha karmaşık ve maliyetli bakım gerektiren elektronik ve mekanik parçaya sahiptirler. Düşük hızları ve kısa havada kalış süreleri nedeniyle çok sayıda uçuş gerektireceğinden maliyetlerde artışa neden olacaktır.

SK-İHA'lar ve DK-İHA'lardan; planlanan görevin gereksinimleri ve çevre koşulları, örneğin, iniş/kalkış pistinin olması veya olmaması durumu, göz önünde bulundurularak uygun olan tercih edilecektir.

Büyüklüklerine Göre İHA'ların Sınıflandırılması

İHA'lar ayrıca büyüklüklerine ve diğer temel özelliklerine göre üç ana sınıf ve bu sınıfların altında yedi gurupta ifade edilmektelerdir. İHA'ların sınıflanmasında

Türkiye'de temel kriter İHA'nın irtifası iken, NATO ve AB ülkelerinde İHA'nın ağırlığı göz önünde bulundurulmaktadır. Büyüklüklerine göre İHA'ların sınıflandırılması Tablo 2.1'de gösterilmektedir [33].

İHA'lar ağırlıklarına göre kategorize edildiğinde; ağırlığı 150 kg kadar olan İHA'lar Sınıf 1 olarak tanımlanmış olup, 2 kg dan hafif olanlar Mikro, ağırlığı 2 kg ile 20 kg arasında olanlar Mini, ağırlığı 20 kg dan fazla olanlar Küçük kategorisi olarak belirlenmiştir. Ağırlığı 150 kg ile 600 kg arasında olan İHA'lar Sınıf 2 olarak tanımlanmış olup; Taktik İHA kategorisi olarak belirlenmiştir. 600 kg dan daha ağır olan İHA'lar Sınıf 3 olarak tanımlanmış olup , Operatif (MALE-Medium Altitude Long Endurance-Orta İrtifa ve Uzun Menzil), Starejik (HALE-High Altitude Long Endurance-Yüksek İrtifa ve Uzun Menzil), Taarruz-Atak kategorisi olarak belirlenmiştir.

Tablo 2.1: İHA Sınıflandırma Tablosu.

Sınıf	**Kategori**	**Operasyon İrtifası (feet)**	**Menzil Yarıçapı (km)**	**Havada Kalma Süresi (saat)**	**Örnek Sistemler**
Sınıf 1 (< 150 kg)	Mikro < 2 kg	AGL* + 200	5	1	Black Widow, MicroStar, Microbat, FanCopter, QuattroCopter, M05quito, Homet, Mite, Arı
	Mini 2-20 kg	AGL + 3000	25	<2	ScanEagle, Skylark, DH3, Mikado, Aladin, Tracker, DragonEye, Raven, Pointer II, Carolo C40/P50, 5korpio, R-Max and R-50, RoboCopter, YH- 3005L, Bayraktar, Efe, Gözcü
	Küçük > 20 kg	AGL + 5000	50	3-6	Hermes 90, Scorpi 6/30, Luna, SilverFox, EyeView, Firebird, R-Max Agri/ Photo, Homet, Raven, Phantom, GoldenEye 100, Flyrt, Neptune
Sınıf 2 (150-600 kg)	Taktik	AGL + 10000	200	6-10	Sperwer, İview 250, Watchkeeper, Hunter B, Mücke, Aerostar, Sniper, Falco, Armor X7, Smart UAV, UCAR, Eagle Eye+, Alice, Extender,Shadow 200/400, Taktik (ODTÜ), Çaldıran, Karayel
Sınıf 3 (> 600 kg)	Operatif (MALE)	AGL + 45000	Sınırsız	24-48	Reaper, Hermes 900, Skyforce, Hermes 1500, Heron TP, MQ-l Predator, Predator-IT, Eagle1/2, Darkstar, E-Hunter, Dominator, Anka
	Stratejik (HALE)	AGL + 65000	Sınırsız	24-48	Global Hawk, Raptor, Condor, Theseus, Hellos, Predator B/C, Libellule, EuroHawk, Mercator SensorCraft, Global Observer, Pathfinder Plus
	Taarruz - Atak	AGL + 65000	Sınırsız	>48	Pegasus
*AGL: Zemin seviyesi (Above Ground Level)					

İHA teknolojisine sahip ülkeler tarafından dünyada üretilen başlıca İHA modelleri Tablo 2.2'de verilmiştir. Ülkeler tarafından üretilen envanterde kayıtlı İHAların çoğunluğu gözetleme ve keşif amaçlı kullanılan silahsız modellerdir. Örneğin; ABD ordusunun sahip olduğu 7448 İHA içindeki silahlı İHA sayısı 241

adettir. Bu sayı toplama oranlandığında %3 düzeyinde bir orana denk gelmektedir [33].

Türkiye Savunma Sanayi Müsteşarlığı tarafından projesi yürütülmekte olan, Bayraktar Taktik İHA Sistemi MALE sınıfı tam otomatik uçuş kontrol özellikleri ile geliştirilmiştir. 2009 yılında resmi heyet huzurunda uçuş testleri sergilenmiştir. Hangardan çıktığı andan itibaren tam otomatik taksi, kalkış, uçuş, iniş, frenleme ve tekrar hangara dönüş gibi özellikler sergilenmiştir. 2012 yılında geliştirilen Bayraktar TB-2 Türk Silahlı Kuvvetleri tarafından operasyonel olarak kullanılmaktadır. İHA ve SİHA olarak kullanılabilen Bayraktar TB-2 Mini Akıllı Mühimmat-MAM ve Harp başlığı ile güçlendirilmiş hali olan MAM-L mühimmatları ile Uzun Menzilli Tank Savar-UMTAS füzelerini kullanabilmektedir. Mühimmat yüklü Bayraktar TB-2 S/İHA Şekil 2.1'de gösterilmektedir [34].

Şekil 2.1: Bayraktar TB-2 S/İHA

Tablo 2.2: Üretilen ve Kullanılan Başlıca İHA Modelleri

Kategori	İsim	İrtifa (Ft.)	Uçuş Süresi	Faydalı Yük (kg)	Ağırlık (kg)	Üretici Ülke
Mikro	Francopter		25 dk.		1,5	
	Black Widow	769	2 dk.		0,05	ABD
	MicroStar				11,8	ABD
	Wasp	50-1000	50 dk.			ABD
	gMAV / T-Hawk	10000	50 dk.			ABD
	QuattroCopter		10 dk.	0,28	0,94	
Mini	ScanEagle	19500	24+	20	13,1	ABD
	Skylark	15000	2		5,5	İsrail
	DragonEye RQ-14A	1000	1		2,7	ABD
	Pointer 2 FQM-151A	985	1		4,3	ABD
	RoboCopter 300			294	500	Japonya
	Raven (RQ-11)	15000	1,5		1,9	ABD
	R-Max Type 2		1	30	58	Japonya
	Efe	12000	1,5		4,1	Türkiye
	Gözcü (Bayraktar Mini)	12000	1		3,5	Türkiye
	Aladin	14760	>1		<4	Almanya
Küçük	SUAZ AECV Puma	100-500	2			ABD
	Hermes 900	15000	12-15	55	110	İsrail
	Scorpio 6/30	>6560	1		13	Fransa
	Luna	16400	6-8		<40	Almanya
	SilverFox	12000	>8	2,3	11,8	ABD
	Neptune	8000	4	92	613	ABD
	GoldenEye 80	15000	3	12	57	ABD
	Firebird	30000	40	14	90	ABD
Taktiksel	Sperwer	15000	>6	45	285	Fransa
	Watchkeeper	16000	17	150	450	İsrail/Fransa
	Hunter	20000	21	100	785	İsrail
	Aerostar	18000	<12	50	200	İsrail
	Falco	20000	14	70	420	İtalya
	Shadow 200	15000	9	82	131	ABD
	Eagle Eye	20000	6	533	828	ABD
	Karayel	18000	10			Türkiye
	Çaldıran (Bayraktar-Çaldıran)	30000	16	40	350	Türkiye
	Mücke	11500	3,5	35	130	Almanya
Operatif (MALE)	Heron Tp	45000	36	1000	3650	İsrail
	Predator	25000	40	204	567	ABD
	Darkstar	65000	12	1880	1980	ABD
	Grey Eagle (MQ-1C)*	25000	40	1452	363	ABD
	Dominator	30000	28	410	790	İsrail
	ANKA	30000	24	200	1368	Türkiye
	Hermes 1500	33000	>26	350	1300	İsrail
Stratejik (HALE)	Global Hawk (RQ-4)	60000	28	1361	14630	ABD
	EuroHawk	65000	>30	1360		ABD/Almanya
	Predator (MQ-1)*	25000	24+	205	1021	ABD
	BAMS (RQ-7)	60000	34+	1450	14630	
	Raper (MQ-9), Predator B*	50000	24	1700	4763	ABD
	Pathfinder Plus	81000	15	67,5	247,5	ABD
	Global Observer	70000	2 Hafta	454	4536	ABD
	Helios	100000	40	330	600	ABD
	Mercator	70000	82	2,5	53	İngiltere
Saldırı	Pegasus X-47A	40000+		472	1740	ABD
Rotorlu (DK-İHA)	Fire Scout (MQ-8)	20000	6+	273	1429	ABD
	Malazgirt	12000	1,5	2	12	Türkiye
	A160T Hummingbird	30000	20+	1850	1134	ABD
* Silahlı İHA olarak bilinmektedir.						

2.1.2 İHA'lar İçin Rota Planlama

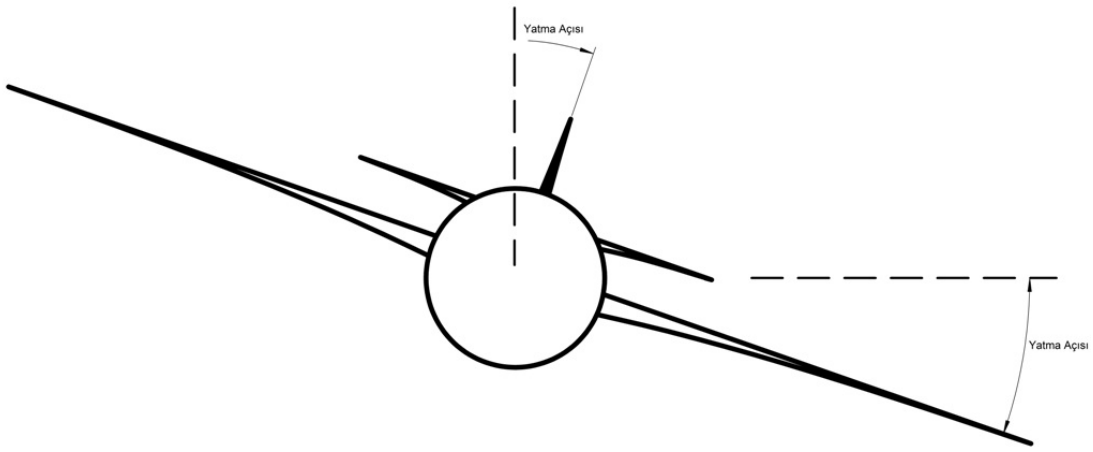
Genel olarak rota planlamasının karmaşık bir süreç olduğundan bir önceki bölümde bahsedilmişti. Özel olarak yukarıda verilen bilgiler ışığında İHA ele

alındığında otonomi ile birlikte rota planlama bir zorunluluk haline gelmektedir. Özellikle alçak irtifa ve kısa menzilli İHA'lar için planlanan rotanın optimize edilmesi maliyeti minimize edeceğinden önemli bir ihtiyaçtır.

Bu çalışmada kısa menzilli İHA'ların görev alanları için rota planlanması ve planlanan rotanın yumuşatma yöntemleri kullanılarak İHA'nın uçuş kabiliyetlerine uygun hale getirilmesi konusu ele alınmıştır. Örneğin, bir havalimanının güvenliğinin İHA yardımı ile sağlandığı durum göz önünde bulundurulabilir. Bu durumda havalimanının sınır güvenliği devriye görevi yapan İHA'lar ile birlikte ısı ve hareket sensörlerinden oluşan yardımcı yer sistemleri kullanılarak icra edilmektedir. Sınırlı havada kalış süresi ve menzile sahip İHA'ların bu görevi optimize edilmiş bir rota üzerinde uçarak gerçekleştirmeleri; zaman, enerji ve maliyet açısından önemli avantaj kazandıracaktır.

İHA'nın uçuş rotası planlanırken hareket halinde olan İHA'nın minimum dönüş yarıçapına uygun bir rota olması gerekmektedir. Özel olarak DK-İHA'ların havada sabit durduğunda istediği yönde hareket edebileceğinden bahsedilmişti, ancak görev icrasında hareket halinde olacaklarından hızlarına bağlı olarak manevra sırasında bir minimum dönüş yarıçapına ihtiyaç duyarlar.

İHA'lar dönüş esnasında dönüşü gerçekleştirmek istedikleri yönde Şekil 2.2'de gösterildiği gibi belli bir yatma açısına göre yana yatarak dönüş icra ederler [35].



Şekil 2.2: Dönüş Sırasında İHA'nın Yatma Açısı

Bir uçuş testinden üretilen sürtünme kutupları kullanılarak İHA için minimum dönüş yarıçapı; maksimum dönüş hızı ve titremeden gerçekleştirebileceği maksimum yatma açısı bulunabilir. Dönüş yarıçapı R , dönüş hızı ω ve yatma açısı ϕ Anderson tarafından tanımlanan Denklem 2.1, Denklem 2.2 ve Denklem 2.3'te belirtilen formüller ile hesaplanır [36].

$$R = \frac{V^2}{g\sqrt{n^2 - 1}} \tag{2.1}$$

$$\omega = \frac{g\sqrt{n^2 - 1}}{V^2} \tag{2.2}$$

$$\phi = arccos\frac{1}{n} \tag{2.3}$$

Burada; V havadaki hız, g yer çekimi ivmesi, W'nin ağırlık ve L nin kaldırma kuvveti olduğu durumda, yük faktörü $n = \frac{L}{W}$ olarak tanımlanmıştır. Bu denklemlere göre teorik olarak yük faktörü $n = 1,5$ ve hava hızı $V = 16$ m/sn. olması durumunda dönüş yarıçapı $R = 23$ m ve yatma açısı $\phi = 48,2°$ olarak hesaplanır [37].

Küçük İHA'lar üzerine yapılan bir çalışmada İHA için yatay olarak minimum dönüş yarıçapı Denklem 2.4 ile tanımlanmıştır [38].

$$R_{min} = \frac{2}{\rho g}\left(1 - \frac{1}{n_{maks}^2}\right)^{-0,5}\frac{W}{S}\frac{1}{CL_{maks}} \tag{2.4}$$

Burada; ρ hava yoğunluğu, g yerçekimi ivmesi, CL_{maks} maksimum kaldırma katsayısı, n_{maks} maksimum yapısal yük faktörü, $\frac{W}{S}$ kanat yükü (aracın ağırlığı W'nin kanat yüzeyi S'ye oranı) olarak belirlenmiştir. Maksimum yük faktörü küçük İHA yapısının dayanabileceği maksimum kaldırmanın, İHA'nın ağırlığı olan W 'ye bölünmesi ile elde edilen oranı temsil eder. Bu durumda Denklem 2.4'e göre dönüş yarıçapını azaltmanın üç yolu vardır; yüksek CL_{maks}, yüksek n_{maks} ve düşük $\frac{W}{S}$. İncelenen çalışmalar sonucunda İHA'lar belirlenmiş sabit değerli minimum dönüş yarıçaplarının olmadığı görülmüştür. Bunun yanında uygulanan yumuşatma yöntemlerinin sağlıklı karşılaştırılabilmesi için bir minimum dönüş yarıçapına ihtiyaç duyulmaktadır. Çalışmada Küçük Sınıf İHA'lar baz alınacaktır. Küçük Sınıf İHA'lar, taşınabilir olmaları, işletme maliyetlerinin ucuz olması ve sessiz olmaları avantajlarına sahiptir [39].

İncelenen çalışmalarda kullanılan minimum dönüş yarıçapı olarak 40 m ile 150 m arası değerlerin belirlendiği görülmüştür [39–42].

İHA rota planlamasının kısıtları göz önünde bulundurulduğunda, uçulabilir rota planlaması matematiksel olarak belli oranda eğrilik içeren eğriler tarafından hesaplanacaktır. İHA rota planlamasına, rotanın yumuşatılması için kullanılan başlıca yöntemlerin; Bezier Eğrisi, B-Spline Eğrisi ve Dubins Yolu yöntemleri olduğu incelenen çalışmalar sonucunda görülmüştür. İncelenen çalışmaların özet bilgisi Bölüm 2.2'de sunulmuştur.

2.2 İlgili Çalışmalar

İHA nın rota planlaması farklı dinamik değişkenler içeren karmaşık bir süreçtir. Birçok bilim dalı kendisine bakan yönü ile rota planlaması konusunda araştırmalar yapmaktadır. Matematik anabilim dalı olarak bize bakan yönü ise planlanan rotanın yumuşatılması kısmıdır. İHA rotasının planlama işleminde öncelikle verilen noktaların ziyaret edilme sırası belirlenir. Daha sonrasında sırası belirlenen noktalar İHA nın yaptığı nonholonomik hareket göz önünde bulunarak eğriler yardımı ile yumuşatılır. Yumuşatma yöntemleri incelenirken en kısa yol uzunluğunu vermesi, belirtilen kontrol noktalarının üstünden geçmesi veya yakınsaması yönleri göz önünde bulundurulmuştur. Yumuşatma kısmında en çok kullanılan üç yöntem olan Bezier Eğrileri, B-Spline Eğrileri ve Dubins Yolu ile ilgili çalışmalar incelenmiştir.

İHA yol planlama problemini çözmek için geliştirilen yöntemlerden biri olan Meyve Sineği Optimizasyonu Algoritması (Fruit Fly Optimization Algorithm-FOA) [43]'te önerilmiştir. Geliştirilen algoritma, kodlanmış faz açısı ve mutasyon adaptasyon mekanizmalarının temel FOA'ya eklenmesi ile oluşturulmuştur ve (phase angle-θ-Mutation Adaptation Fruit Fly Optimization Algorithm) θ-MAFOA olarak adlandırılmıştır. Mutasyon adaptasyon mekanizmaları, FOA'nın sömürü ve keşif kabiliyetlerinin dengelenmesi için benimsenmiş olup, meyve sinekleri için faz açısı tabanlı kodlanmış strateji yakınsama sürecindeki yüksek performansın elde edilmesine yardımcı olmuştur.

Daha sonra önerilen θ-MAFOA, çeşitli yeryüzü savunma silahları ile karmaşık üç boyutlu (3B) arazi ortamlarında İHA'ya en uygun yolu bulmak için kullanılmıştır. Başlangıç ve bitiş noktalarını birleştiren B-Spline Eğrisi, yumuşatılmış bir yolu temsil etmek için kullanılmıştır. B-Spline Eğrisi, global ve pürüzsüz bir doğaya sahiptir ve bu nedenle, sonuçtaki İHA yolu, otonom navigasyon kontrolörleri tarafından sorunsuz ve kolayca üretilir. Çalışmada İHA yollarının maliyetini değerlendirmek için performans kriterleri ve kısıtları göz önünde bulundurulmuştur.

İHA için çözülen rota planlama problemi, birçok noktada benzerlik gösteren silahlı insansız hava araçlarının rota planlaması için de kullanılmaktadır. Silahlı İnsansız Hava Aracı (SİHA) için üç boyutlu yol planlaması problemini ilk seferinde optimize etmek için Yarasa Algoritmasının (Bat Algorithm-BA) geliştirilmiş bir versiyonu bir Diferansiyel Evrim (Differential Evolution-DE) ile birleştirilmiş, ve geliştirilen algoritma (Improved version of Bat Algorithm) IBA [44]'te önerilmiştir. IBA'da, DE yarasa popülasyonunda en uygun bireyi seçmektedir. Önerilen IBA'yı kullanarak seçilen kontrol noktaları birleştirilerek güvenli bir yol elde edilmiştir. Elde edilen yolu daha iyi hale getirmek ve SİHA'nın gerçek uçuşu için daha uygun hale getirmek için B-Spline Eğrileri kullanılmıştır. Çalışmada IBA'nın performansı 3B yol planlama probleminde temel BA ile karşılaştırılmıştır. Elde edilen sonuçlara göre IBA'nın temel BA modeline kıyasla 3B SİHA yol planlama problemleri için daha iyi bir teknik olduğu sonucuna varılmıştır.

Yol planlama problemi non-holonomik araçlar üzerinde çalışma yapan araştırmacılar arasında popüler bir alandır. Reinis C. ve arkadaşlarının yaptığı çalışmada, çeşitli tiplere uygulanabilen Bezier Eğrilerine dayalı eğrilik kontrolü ile pürüzsüz bir yol planlayıcısı geliştirilmiştir [45]. Teorik altyapı ve deney sonuçları, kontrol noktaları arasındaki eğriyi düzenlemek için B-Spline yöntemiyle karşılaştırılmıştır. Düzgün bir yörünge için, eğrilik kontrolü ile daha kısa bir yol uzunluğu, kontrol noktalarının optimal konumlarının bulunduğu Bezier Eğrileri ile elde edilmiştir. Bezier Eğrileri parçalı bir şekilde hesaplanmış ve bütün yolu elde etmek için birleştirilmiştir. Geliştirilen yöntem eğrinin kolay kontrolünü, çarpışma doğrulamasını ve hızlı hesaplama yapmayı sağlamaktadır.

Elde edilen deney sonuçlarına göre benzer yöntemlere kıyasla daha iyi performans gösterdiği belirtilmiştir.

Non-holonomik hareket eden araçlardan biri de robotlardır. [46]'da mobil robotlar için, optimum robot yolunu elde etmeye yönelik Modifiye Genetik Algoritma (Modified Genetic Algorithm-MGA) kullanılarak dinamik alanda Bezier Eğrisi tabanlı yol planlama yapılmıştır. Önerilen MGA, standart Genetik Algoritma üretilen çözümlerinin çeşitliliğini ve MGA'nın arama yeteneklerini artırmayı amaçlamaktadır. Önerilen yöntemde, robotun yolu engellerin konumlarına göre dinamik olarak belirlenmiştir. Başlangıç noktası ile hedef nokta arasındaki mesafenin optimize edilmesi amacıyla, MGA, Bezier Eğrisinin kontrol noktası olarak kullanılmak üzere en uygun noktaların belirlenmesi için kullanılmıştır. Seçilen kontrol noktaları kullanılarak, başlangıç ve bitiş noktaları arasındaki toplam mesafeyi en aza indiren optimum yumuşatılmış yol seçilmiştir. Geliştirilen model, farklı ölçekler, farklı engeller ve altı kıyaslama haritası ile farklı ortamlarda test edilmiştir. Elde edilen sonuca göre, önerilen yöntemin robotun zorlu ortamlarda enerji tüketimini önlemek için etkili bir yol sağladığıdır. Önerilen algoritmanın yerel optimum sorununu çözemediği ve yerel araştırma konusunda geliştirilmesi gerektiği belirtilmiştir.

Birden fazla İHA bulunması durumunda her bir İHA için farklı hedeflere yönelik ayrı yollar planlanır. Weinan W. ve arkadaşları tarafından geliştirilen yöntem ile çoklu heterojen insansız hava araçlarının kooperatif misyon planlaması için hızlı ve eşleşmiş bir çözüm üzerinde çalışılmıştır [47]. İHA filosunun çoklu hedefler üzerine hareket etmesi ve ardışık görevleri yerine getirmesi için en uygun çözüm yolunu tanımlaması gereken bir problem çözümü incelenmiştir. Hızlı ve uygulanabilir bir çözüm elde etmek için, problemin görev atamasını ve yörünge oluşturma yönlerini bütünleştiren birleştirilmiş ve dağıtılmış bir planlama yöntemi geliştirilmiştir. Belirli kısıtlamalar ve serbest bir Dubins Yolu ile kooperatif misyon planlama problemi yeniden formüle edilmiştir. En uygun çözümü araştırmak için Dağıtılmış Genetik Algoritma (Distributed Genetic Algorithm-DGA) önerilmiştir. GA'yı oluşturan kromozomal genler, İHA'ların heterojen özelliğine uyum sağlamak üzere modifiye edilmiştir. Önerilen görev planlama yöntemini doğrulamak için 6 derecelik Serbestlik Dereceli (Degrees of

Freedom-DOF) sabit bir İHA modeli ve yol takip yöntemi kullanılmıştır. Simülasyon sonuçlarına göre önerilen yaklaşım ile uygun çözümler elde edildiği ve gerçek bir görevde kullanım potansiyeli ile işletme oranını önemli ölçüde geliştirdiği gösterilmiştir. Ancak önerilen yöntem belli bir yükseklikteki bir düzlemle sınırlı olduğu ve gerçek bir görevde kullanılamayacağı belirtilmiştir. Önerilen algoritmanın verilen senaryolar ile sınırlı olduğu ve geliştirilmesi gerektiği görülmüştür.

İHA yol planlama probleminde sabit hedefler de olduğu gibi hareketli hedefler üzerinde yapılan çalışmalar mevcuttur. [48]'de Hareketli Hava Hedefini (Moving Air Target-MAT) izleyen insansız hava aracı için online hareket planlama yöntemi konusu çalışılmıştır. Bir İHA'nın algılama aralığı, İHA'nın ileri görüş açısında yer alan yelpaze şeklinde bir alanla sınırlıdır. Bu çalışmada, güvenlik mesafesi kısıtlamaları ve sensör algılama aralığı kısıtlamaları dikkate alınarak, İHA kullanarak MAT algılama ve izleme için bir yöntem önerilmiştir. Çalışmada incelenen optimizasyon problemini çözmek için Diferansiyel Evrim tabanlı bir yöntem geliştirilmiştir. Ayrıca, DE'nin verimliliğini artırmak için Azalan Ufuk Kontrolü (Receding Horizon Control-RHC) benimsenmiştir. Ek olarak, oluşturulan yolların düzgün ve akıcı olmasını sağlamak için, eğrilik kısıtlamaları dikkate alınarak Dubins yol planlama yöntemi kullanılmıştır. Sensör algılama hatasının yol planlamasına etkisi göz önüne bulundurularak analiz edilmiştir. Yaygın olarak kullanılan ve rekabetçi bir optimizasyon yöntemi olan Genetik Algoritma (Genetic Algorithm-GA) kullanılmıştır. Sonuçların hesaplanması ve karşılaştırılması ile, DE'nin, MAT'nin tespitinde ve izlenmesinde GA'dan daha iyi performans gösterdiği ve mümkün olan en verimli yolu üretebildiği gösterilmiştir. Daha sonraki çalışmalarda önerilen yöntemin çoklu İHA'lara ve çoklu MAT'lere genişletilerek geliştirilebileceği belirtilmiştir.

İHA'lar kanat özelliklerine ve motor yapısına göre farklı görevlerde kullanılmaktadır. Chen Y. ve arkadaşları tarafından döner kanatlı İHA'lar için gerçek ve sahte 3B uzaylar göz önünde bulundurularak yarı-optimal yörüngeleri hesaplamak için Kurt Paketi Arama (Wolf Pack Search-WPS) algoritması uygulanmıştır [49]. Çalışmanın amacı, yol uzunluğunun en aza indirilmesi, yakıt maliyetinin en aza indirilmesi ve planlama yolunun en iyi düzgünlüğünü içeren

çok amaçlı maliyet fonksiyonunun en aza indirilmesidir. GA'da kullanılan kromozom ve gen kavramları geleneksel WPS algoritmasına dahil edilmiştir. Bir kromozomun bir kurdu temsil ettiği prensibine dayanarak, kurt paketinin operasyonları bir popülasyonun güncellenmesine dönüştürülmüştür. Daha sonrasında, GA'daki çaprazlama ve mutasyon operatörleri, modifiye edilmiş WPS algoritmasında tanıtılmıştır. İHA'nın kinematik ve dinamik özellikleri göz önünde bulundurularak, doğru parçaları Kübik B-Spline Eğrisi ile düzeltilmiştir. Kübik B-Spline Eğrisinin kontrol noktaları sabit kanatlı İHA'nın hızlanma limitine göre belirlenmiştir. Simülasyon sonuçlarına göre, modifiye edilmiş WPS algoritmasının farklı 3B uzay türlerinden bağımsız olarak 3B İHA yolu planlama problemini başarıyla çözebileceği gösterilmiştir. Ayrıca, bu yaklaşımın her türlü kısıtlama dikkate alındığında döner kanatlı İHA'lar ve sabit kanatlı İHA'lar için etkili olduğu ve üretilen yolun uçulabilir olduğu belirtilmiştir. Modifiye edilmiş WPS algoritmasının geleneksel WPS algoritması, GA ve rastgele arama yöntemine göre daha üstün olduğu gösterilmiştir.

Çoklu insansız araçların eş zamanlı olarak varış noktasında olmaları için geliştirilen Zamansal-Mekansal (Temporal-Spatial - T-S) Bezier Eğrisi [50]'de incelenmiştir. Eş zamanlı varış süreci tanıtılmış ve formüle edilmiştir. Öncelikle T-S Bezier Eğrisi, Bezier Eğrisinin tek boyutlu bir değişkeni olarak zaman göz önüne alınarak oluşturulmuştur. İkinci olarak, en düşük eğrilik yarıçapı, teğet hızlanma ve hız kısıtlamaları en uyguna yakın T-S Bezier Eğrisini tasarlamak için ele alınmıştır. Ardından, çoklu insansız araçlar için en uyguna yakın eş zamanlı varış zamanı hesaplanmış ve elde edilmiştir. Son olarak bir T-S Bezier Eğrisi kullanarak eş zamanlı varış planlaması özetlenmiştir. Önerilen yöntemin etkinliğini göstermek için simülasyonlar ve karşılaştırmalar yapılmıştır. Sayısal sonuçlara göre, T-S Bezier Eğrisinin çoklu insansız araçların eşzamanlı gelişini sağlamak için etkili ve uygulanabilir olduğu ifade edilmiştir.

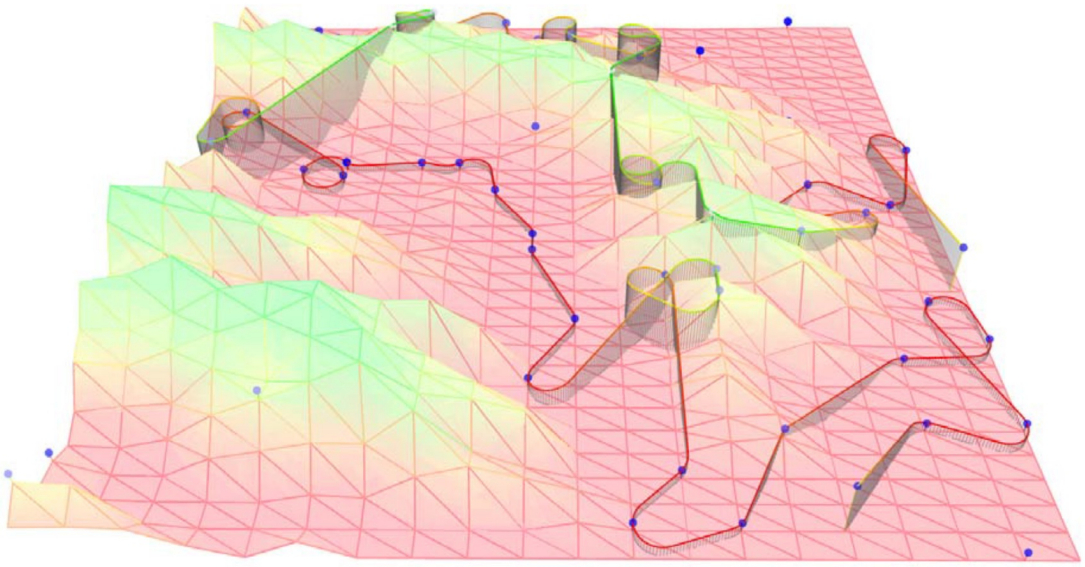
Yol planlama problemi Otonom Sualtı Aracı (Autonomous Underwater Vehicle-AUV) için de geçerli olan bir çalışma alanıdır. [51]'de okyanus akımına göre AUV'a yönelik küresel yol planlaması üzerine bir araştırma yapılmıştır. Çalışmada, en uygun çözüm arayışındaki düşük verimlilik ve küresel yol planlamasında uzun arama süresi gibi sorunlara yönelik yeni bir

Genetik-Karınca hibrit algoritması önerilmiştir. Akıntının yetersiz kabul edildiği durum göz önüne alındığında, B-Spline fonksiyonuna dayanan kılavuzları olan bir okyanus akıntı modeli esas alınmıştır. Sonuçta, hibrit algoritmada entegrasyon verimliliğini artırmak için yeni bir optimal entegrasyon stratejisi ortaya konmuş ve değerlendirme fonksiyonu, mevcut olanla birlikte minimum enerji tüketimine dayanarak belirlenmiştir. Simülasyon deneyleri ile, arama hassasiyeti korunurken mevcut hibrit algoritmalarla kıyaslandığında, yeni algoritmanın arama süresinin ve iterasyonların sayısının belirgin bir şekilde azalması avantajına sahip olduğu gösterilmiştir. Gerçek hayatta akıntının etkisi olacağı düşünüldüğünde çalışmanın geliştirilmesi gerektiği değerlendirilmektedir.
[52]'de Parçacık Sürüsü Optimizasyonu'na (Particle Swarm Optimization-PSO) dayalı yol planlama problemlerinde kullanılan üç farklı eğri karşılaştırılmıştır. Yol planlama probleminde en kısa yol uzunluğu, güvenlik, fizibilite, akıcılık vb. kriterleri karşılayan çarpışmasız yol bulma amaç olarak belirlenmiştir. Burada belirtilen kısıtlamalara uygun eğriler ve algoritmalar tanımlanmıştır. Yaygın olarak kullanılan üç eğri, Ferguson Eğrisi, $\eta 3$ Eğrisi ve Bezier Eğrisi, bir dizi yol planlama problemini çözme performanslarına göre karşılaştırılmış ve tartışılmıştır. Bu eğriler için gerekli parametrelerin optimize edilmesi için Dinamik Çoklu Sürü Parçacık Sürüsü Optimizasyonu (Dynamic Multiswarm Particle Swarm Optimization - DMS-PSO) kullanılmıştır. DMS-PSO'nun küresel aramada geleneksel PSO'dan daha iyi olduğu sonucuna varılmıştır. Elde edilen sonuçlara göre, Bezier Eğrisinin bu çalışmada ele alınan belirli yol planlama problemlerinin çözümü için diğerlerinden daha uygun olduğu belirtilmiştir. Deney koşullarının sınırlandırılması için, bu yazıda sadece yol uzunluğu ve güvenlik kriterleri karşılaştırılmıştır. Çalışmanın geliştirilmesi adına daha fazla kriterin karşılaştırılmasının uygun olacağı değerlendirilmektedir.
Gurkan K. tarafından yapılan çalışmada Bezier Eğrilerine dayanan yol planlaması ile nonholonomik mobil robot kontrol simülasyon sonuçları sunulmaktadır [53]. Çalışmanın amacı, MATLAB / Simulink ortamını kullanarak Diferansiyel Sürüş Tekerlekli (Differential Drive Wheeled-DDW) robot için bir simülasyon modeli geliştirmektir. Kesirli mertebeli Oransal İntegral Türev (Proportional Integral Derivative-PID) yapısı, DDW robotunun

izleme performansını artırmak için kullanılmıştır. Bezier Eğrileri, bu çalışmada yol planlamasına katkıda bulunmak için kullanılmıştır. Elde edilen simülasyon sonuçları çeşitli yol senaryoları için karşılaştırılmıştır. Önerilen kontrol şemasının istenen yolu izlemesi için uygun performans sergilediği, her iki yol senaryosunda hedefe ulaşmada iyi sonuçlar gösterdiği belirtilmiştir. Sonraki çalışmalarda, eş zamanlı lokalizasyon ve haritalama ile gerçek zamanlı platformlarda kesirli mertebeli lineer olmayan kontrol algoritmalarının ve görüntü işleme tekniklerinin kullanılması planlanmaktadır.

Hava sahası koordinasyonunda bilgi taşıyıcı İHA'nın yol planlaması [54]'te ele alınmıştır. Bir İHA ve çoklu İnsansız Kara Araçları'ndan (İKA) oluşan heterojen araçlar içeren bir ekip için, bir İHA planlama yolu çalışılmıştır. İKA'lar, mobil erişim düzenekleri olarak kullanılırken, İHA, İKA'lar arasında bilgi paylaşımını sağlamak için bir bilgi taşıyıcı olarak hizmet vermektedir. İHA, bilgiyi toplamak için periyodik olarak her İKA üzerinden uçmak ve daha sonra bilgiyi diğer İKA'lara iletmek zorundadır. Bilgi taşıyıcı mekanizmasındaki İHA yolu planlama problemi, Komşularla Dinamik Dubins Gezgin Satıcı Problemi (Dynamic Dubins Traveling Salesman Problem with Neighborhood-DDTSPN) olarak formüle edilmiştir. Bu problemin amacı, İHA'nın istenen tüm İKA'lara bilgiyi iletmesini sağlayan en kısa rotayı bulmak olarak belirtilmiştir. Bu problemi çözerken, problemi iki alt probleme ayırmak için bir ayrıştırma stratejisine dayanan bir algoritma önerilmiştir. Birincisi, İHA'nın ziyaret etmesi için tüm İKA'ların sırasının belirlenmesi ve ikincisi de İHA'nın her bir İKA için komşuları arasındaki erişim yerlerinin optimize edilmesi olarak tanımlanmıştır. Temsili senaryolardaki deneylerde, geliştirilen algoritmanın, bilgi taşıyıcı İHA'nın bilgiyi toplamak ve iletmek için İKA'ların etkili iletişim aralığında güvenilir ve hızlı bir şekilde uçtuğu gösterilmiştir. Bilgi toplayıcı İHA'nın İKA'ların hareket parametreleri hakkında önceden bilgi sahibi olmaması durumunda çalışması için algoritmanın geliştirilmesi gerekmektedir.

Sınırlı bir seyahat bütçesi olan sabit kanatlı insansız hava aracı için Şekil 2.3 ile örneği gösterilen veri toplama planlama problemi [55]'te ele alınmıştır.



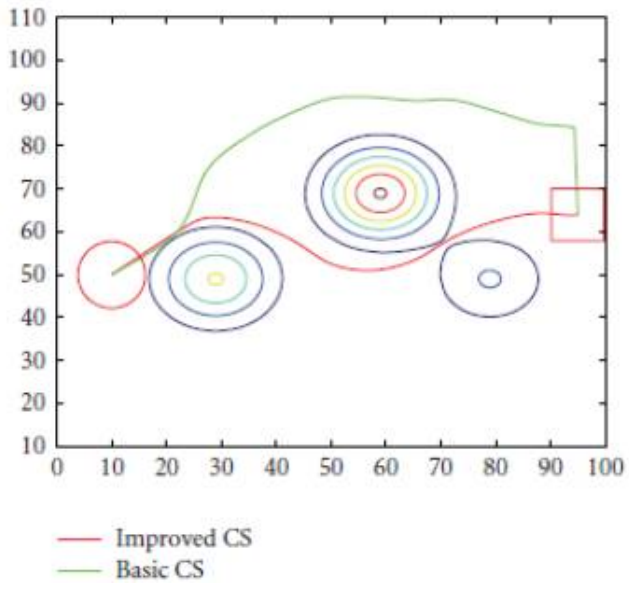
Şekil 2.3: Dubins İHA Oryantiring Problemi Örneği.

Problemi, üç boyutlu uzaya ve eğrilik kısıtlaması olan araçlara genişletmek için Dubins uçak modelinin kullandığı Oryantiring Problemi'nin (Orienteering Problem-OP) bir değişkeni olarak formüle edilmiştir. Önerilen Dubins Uçak Oryantiring Problemi (Dubins Airplane Orienteering Problem-DA-OP), sınırlı seyahat bütçesini aşmayan, yörüngenin belirli hedef konumlarının bir alt kümesini ziyaret eden en faydalı veri toplama yörüngesini bulmayı amaçlamaktadır. Orijinal OP formülünün aksine, önerilen DA-OP, ziyaret edilecek dizinin belirlenmesi ile birlikte ziyaret edilecek hedeflerin bir alt kümesini belirleyen kısmı birleştirir. Bunun yanında Dubins uçağı için en kısa rotayı bulmada sürekli optimizasyonla ilgili zorlukları da içermektedir. Problem olası hedeflere yaklaşmakta olan açılardan örnek alınarak ele alınmakta ve Rastgele Değişken Komşuluk Arama (Randomized Variable Neighborhood Search-RVNS) yöntemi ile bir çözüm bulunabilmektedir. Önerilen çözümün bir fizibilitesi, OP değişkenlerinin hedeflerin çeşitli yüksekliklerine sahip senaryolarına yönelik değiştirilmiş karşılaştırmaları üzerine ampirik bir değerlendirme ile kanıtlandığı belirtilmiştir.

Üç boyutlu ortamda İHA'lar için koordine edici bir yol planlama ve yumuşatma algoritması [56]'da önerilmiştir. 3B ortamlarda İHA'lar için yollar yumuşatılmış, uçulabilir ve güvenli olmalıdır. Çoklu İHA'ların çeşitli kısıtlamaları olan,

koordineli yol planlama problemini çözmek için, evrimsel bireyler olarak bir grup İHA yolu kullanan Diferansiyel Evrim'e dayalı çok amaçlı bir optimizasyon yaklaşımı ele alınmıştır. Ayrıca DE'nin etkinliğini arttırmak için yeni bir diferansiyel strateji benimsenmiştir. İHA kısıtlamalarına ve koordineli olma kısıtlamalarına uyacak şekilde, pareto (çok amaçlı) optimal çözümünü ölçmek için kooperatif bir uygunluk fonksiyonu verilmiştir. Daha sonra, İHA'lar için bir grup yuvarlatılabilir yumuşatma yolu elde etmek amacıyla bir B-Spline Eğrisinin kontrol noktaları ters olarak hesaplanmıştır. Simülasyon sonuçlarının, önerilen yaklaşımın koordineli-işbirlikçi doğruluğu etkili bir şekilde geliştirdiği ve İHA'ların yollarını yumuşatmada iyi çalıştığı belirtilmiştir.

Gaige W. ve arkadaşları tarafından SİHA 3B yol planlama problemini çözmek için, Şekil 1.2'de örneği gösterilen, yeni bir Hibrit Metaheuristik-Üstsezgisel Diferansiyel Evrim (Differential Evolution-DE)/Guguk Kuşu Arama (Cuckoo Search-CS) algoritması önerilmiştir [57].



Şekil 2.4: SİHA Rota Planlamasının Karşılaştırılması.

SİHA için 3B yol planlaması, karmaşık savaş alanı ortamlarında farklı türdeki kısıtlamaların göz önünde bulundurulmasıyla öncelikli olarak uçuş rotasını optimize eden karmaşık yüksek boyutlu bir optimizasyon problemidir. DE, guguk kuşunun yuvada güncellenmesi sırasında geliştirilmiş CS modelinin guguk

kuşunun seçme sürecini optimize etmesi için uygulanmaktadır. Guguk kuşlarının, en uygun SİHA yolunu araştırırken birer ajan olarak hareket edebileceği değerlendirilmiştir. Bunun ardından, SİHA için tehdit alanlarından kaçınarak ve minimum yakıt harcayarak, seçilen kontrol noktalarının koordinatlarını bağlayarak güvenli yol bulunur. Bu yeni yaklaşım, temel CS'nin güçlü sağlamlığını korurken küresel yakınsama hızını artırabileceği değerlendirilmiştir. Optimize edilmiş SİHA yolunu daha uygun hale getirmek için, yolun yumuşatılması B-Spline Eğrisi ile sağlanmıştır. B-Spline rota yumuşatma algoritmasının düşük bir hesaplama yüküne sahip olduğu ve gerçek zamanlı olarak çalıştırılabileceği belirtilmiştir. Önerilen hibrit metaheuristik DE/CS yönteminin performansını kanıtlamak için temel CS algoritması ile karşılaştırılmıştır. Simülasyon sonuçları, önerilen yaklaşımın SİHA 3B yol planlamasında temel CS modelinden daha etkili ve uygulanabilir olduğu gösterilmiştir.

[58]'de İHA için odaklı dinamik A* algoritmasına dayalı bir yol planlama algoritması, 3B dinamik ortamlar altında sunulmuştur. Problem iki adımda ele alınmaktadır: Bir olası çevre haritasından alınan bilgilere dayanarak, öncelikle A* algoritmasına göre hareketsiz tehditlerden kaçınan bir yol planlanmaktadır. Ardından yolu takip ederken İHA haritayı günceller ve sensör bilgisi ile yolu düzeltir. Yolun düzgünlüğünü artırmak için, B-Spline yol yumuşatma tekniği benimsenmiştir. Simülasyon sonuçları, algoritmanın verimliliğini ve geçerliğini göstermiştir. Çalışmanın geliştirilmesi gereken kısımları bulunmaktadır. Lokal yamanın karakteristiğinden dolayı, dinamik engel çok büyük ise veya İHA'nın algılama yarıçapı çok küçük ise algoritma bir yol bulmak zorunda değildir. Bu durumda, mevcut düğümden hedefe yeni bir yolun yeniden planlanması için A* algoritması çağrılması gerektiği belirtilmiştir. Daha büyük haritalarda, ilk pürüzlü yolu planlamak için düşük çözünürlüğe sahip ızgaralar benimsenmesi gerekmektedir.

Tamamen bilinmeyen karmaşık ortamlarda çalışan İHA'nın otonom navigasyonunu geliştirmek için gerçek zamanlı üç boyutlu yumuşatılmış yol planlama algoritması [59]'da sunulmuştur. Bir Hızlı Rastgele Ağaç (Rapidly Random Tree-RRT) tabanlı yol noktası oluşturma algoritması, öncelikle

çarpışma içermeyen yol noktaları oluşturmak için önerilmektedir. Yeni bir yol noktası belirlendikten sonra sunulan yol düzeltme tekniği, bir İHA'nın uçabilmesinin mümkün olduğu düzgün bir yol üretmek için kullanılmaktadır. Özellikle, İHA'nın hareket kısıtlamaları ve ardışık Bezier Eğrileri arasındaki süreklilik gereksinimi göz önüne alındığında, PSO tabanlı parçalı Bezier Eğri uydurma yoluyla yol yumuşatma gerçekleştirilmiştir. Son olarak, birden fazla çözüm mevcut olduğunda optimum bir yol aramak için bir yol seçim stratejisi de tanıtılmaktadır. Simülasyon sonuçları ile önerilen gerçek zamanlı 3B yumuşatılmış yol planlama algoritmasının üstünlüğü gösterilmektedir. Çalışmanın geliştirilmesi açısından, yol planlaması sırasında daha fazla kinematik ve dinamik kısıtlar dikkate alınarak, aynı zamanda algılama bilgilerinin kaybolması ve kör sokaklarda sıkışıp kalması gibi olağanüstü durumlarla başa çıkması için çeşitli stratejiler geliştirilmesi planlanmaktadır. Dahası, önerilen tekniklerin üstünlüğünü daha da doğrulamak için gerçek İHA'lara yerleştirilmesi düşünülmektedir.

Kenneth R.S. ve arkadaşları tarafından Bezier Eğrilerini kullanarak nonholonomik mobil robotlar için gerçek zamanlı yumuşatılmış yörünge üretimi konusu üzerinde çalışılmıştır [60]. Beşinci derece Bezier Eğrileri kullanılarak yumuşatılmış eğriler üretmek için basit ve verimli bir algoritma geliştirilmiştir. Geliştirilen algoritma C^2 sürekliliğine sahip eğriliği ile yumuşak hareket yörüngeleri üretmektedir. Oluşturulan yörünge, bir mobil robotun düzgün hareketini garanti eden tüm temel kriterleri gözlemlemektedir. Operatörün görünürlüğü için tavan kameralı uzaktan işletilen tekerlekli bir robot iç mekan ortamında düşünülmektedir. Hareket yörüngesi, kullanıcı tarafından belirlenen noktalar ve yol genişliği ile sınırlandırılmıştır. Sadece bu iki girdiye dayanan yörüngenin otomatik olarak oluşturulması için bir yöntem geliştirilmiş ve önerilmiştir. Mobil robotun izlenebilirliğini geliştirmek için, Bezier alt bölümünün benimsenmesi keskin köşelerde eğriliğin önemli ölçüde geliştiğini göstermektedir. Önerilen yöntem ayrıca, yörüngesinin başlangıcında ve sonunda ikinci türevin keyfi olarak ayarlanmasına izin vermektedir. Bu özellik ile Bezier alt bölümünün kombinasyonu, engelden kaçınma yörüngesinin oluşturulması için uygun bulunmuştur. Robot bir engelle karşılaştığında, yörünge bölünecek ve

engelleri önleyen yeni bir Bezier Eğrisi düzgün bir şekilde bağlanacaktır. Önerilen algoritmanın gerçek zamanlı engelden kaçınma yörüngesinin oluşturulması için kullanılabileceği belirtilmiştir. Ancak çalışma alanının doğru bilinmesi gerektiği için bu uygulamada ele alınmamıştır. Gerçekleştirilen simülasyon ve deneysel sonuçların önerilen yöntemin etkinliğini gösterdiği belirtilmiştir.

Homojen İnsansız Hava Aracı Sistemleri (Unmanned Aircraft Systems-UAS) takımı için statik engellerin varlığında sürekli bir gözetim uygulaması için düzlemsel bir yol planlayıcısı [61]'de geliştirilmiştir. UAS'da bulunan ekibin rolü, belirlenen süre içerisinde kullanıcı tarafından belirlenen bir alanın tam kapsamını (sensör verisi veya görüntü elde edilmesi) sağlamaktır. Önerilen yöntemde iki aşamalı bir planlayıcı seçilmesi öngörülmüştür. Böylece araç manevrası kısıtlamaları hesaplama açısından verimli bir şekilde karşılanabilecektir. Yolun belirlenmesi için, bir grafik arama ve B-Spline parametrik eğri yönteminin kombinasyonu kullanılmaktadır. Diğer parametrik polinom temelli eğri yaklaşımları yerine B-Spline yönteminin seçilmesi için gerekçeler çalışmada ifade edilmiştir. Çözümün optimal olmadığı, ancak engeller arasında gezinme için ve yeterli görüntü/veri çözünürlüğü, kapsama alanı ve yenileme zamanlaması sağlaması açısından UAS için kısıtları karşıladığı belirtilmiştir. Geliştirilen algoritmanın yol navigasyon sistemlerine entegre edilebileceği ve UAS üzerine gömülü sistemlerde kolayca çalışabilecek bir formatta ayrıntılı olarak geliştirilebileceği belirtilmiştir. Önerilen yöntemin geliştirilmesi gerektiği görülmüştür.

[62]'de sınırlı eğriliği olan araçlar için zaman optimizasyonlu yol problemi ele alınmıştır. TSP'de görevlendirilen satıcı sayısı k olarak tanımlanmıştır. Önerilen çalışmada k-Dubins TSP (k-DTSP), sınırlandırılmış eğriliği olan (Dubins aracı) çoklu robotlar için çevredeki kontrol noktaları arasında verimli yollar planlanması amaçlanmaktadır. Robotun başlığının ayrıştırılmasına dayanan bir kombinatoryal yaklaşım ile üstlenilen problem için doğrusal olmayan bir matematiksel formül önerilmektedir. Yöntemin temel amacı, en uzun turun süresini en aza indirgemek ve böylece tüm noktaları ziyaret etmek için gereken toplam süreyi azaltmaktır. Simüle edilmiş bir ortamda gerçekleştirilen çok

sayıda deneme ile farklı senaryolar için önerilen tekniğin performansının literatürde bulunan klasik yöntemlere dayanan En İyi Alternatif Algoıritması'nı (Best Alternating Algorithm) geride bıraktığı hakkında istatistiksel bilgi sağlamıştır. Kısıtların artırılması ve heterojen robotlar üzerinde uygulanması için önerilen yöntemin geliştirilmesi gerekmektedir.

Çoklu yer hedeflerine saldıran çoklu İHA için bir görev planlama yöntemi [63]'te sunulmuştur. Görev planlama yöntemi, karınca kolonisi optimizasyonuna dayalı misyon tahsisini ve geliştirilmiş Dubins Yoluna dayalı yol planlamayı içermektedir. İlk olarak çoklu hedeflere saldırma problemi, bir optimizasyon problemine dönüştürülmüştür. İkinci olarak, yöntemi gerçek savaş alanı ortamına daha uygun hale getirmek için optimizasyon probleminde bazı ortak kısıtlamalar dikkate alınmıştır. Misyon tahsisinin kısıtları temel olarak, İHA'nın yönetici kabiliyetlerini, uçuş menzillerini, hedefin tehdit derecesini ve saldırının yararını vb. içermektedir. Yol planlamasının kısıtları minimum dönüş yarıçapını ve yol uzunluğunu içermektedir. Çoklu performans endekslerinin optimizasyon hedefini karşılamak için kapsamlı performans endeksi tasarlanmıştır. Ayrıca her İHA'nın uçuş yollarını oluşturmak için Dubins Yolu benimsenmiştir. Bununla birlikte, yol planlaması problemine uymak için Dubins Yolunun bir miktar iyileştirmesi yapılmıştır. Geliştirilmiş Dubins Yolu, orijinal Dubins Yoluna göre daha kısadır ve hesaplanması daha kolaydır. Son olarak, çoklu yer hedeflerine saldıran çoklu İHA'ların bir örneği, misyon planlama stratejilerinin fizibilitesini doğrulamak için gerçekleştirilmiştir. MATLAB simülasyonlarının sonuçları, stratejinin misyon planlama problemlerine uygun olduğunu göstermektedir. Önerilen yöntemin gerçek hayatta uygulanabilmesi için geliştirilmesi gerektiği görülmüştür.

Chen G. ve arkadaşları tarafından saldırgan ortamdaki çoklu İHA'lar için kendinden organize çevrimiçi bir arama ve saldırı algoritması sunulmuştur [64]. Dağıtılmış Arama-Saldırı Görevi Kendinden Organizasyon Algoritması (Distributed Search–Attack Mission Self-Organization Algorithm-SAMSOA), küresel optimizasyon problemini birkaç yerel optimizasyon problemlerine ayırmaktadır. Her bir İHA bir alt sistem olarak kabul edilmekte ve yerel optimizasyon problemini çözmek için ayrı bir işlemci olarak atanmaktadır. Bu

arada İHA'lar arasındaki bilgi alışverişi, her bir alt sistemin çoklu İHA sistemi için en uygun kararı vermesine yardımcı olabilecektir. SAMSOA algoritması, normal uçuş modundan ve tehdit önleme modundan oluşmaktadır. Normal uçuş modunda, arama saldırı misyonu gözetim kapsama oranını en üst düzeye çıkarmak ve hedeflerin mevcudiyet süresini en aza indirmek için modellenmiştir. Ardından geliştirilmiş Dağıtılmış Karınca Kolonisi Optimizasyonu Algoritması, yol noktalarını oluşturmak için tasarlanmıştır. Son olarak, yol noktalarını yumuşatılmış bir şekilde bağlamak için Dubins Yolu kullanılmıştır. Tehdit önleme modunda, bir tehditten kaçınmanın yolu, Dubins Yoluna dayanarak oluşturulmuştur. Önerilen SAMSOA algoritmasının çevrimiçi uygulamasını doğrulamak için bir dizi simülasyon gerçekleştirilmiştir.

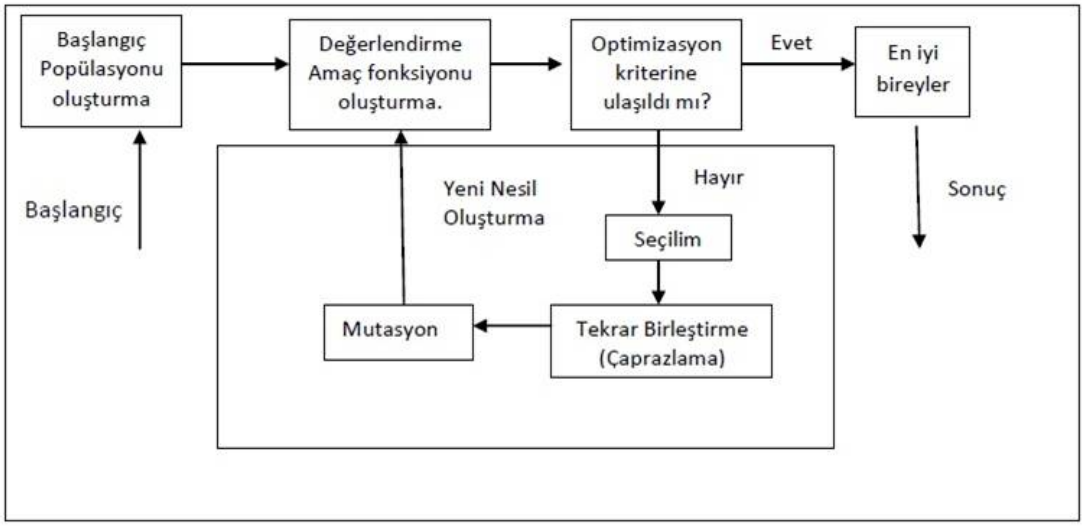
Otonom araçlar için başlangıç durumları ve hedef durumlar arasında yerel yörünge oluşturmak için, Kuartik Bezier Eğrisine dayanan uygun bir yörünge oluşturma algoritması [65]'te önerilmiştir. Problemi basitleştirmek için yörünge oluşturma; yörüngeyi şekillendirmek ve yörüngeyi yürütmek için doğrusal hız profili oluşturmak üzere eğrilik profili oluşturmak olarak ayrılmıştır. Sürekli ve sınırlı eğrilik profili oluşturmak için, belirli özellikler nedeniyle Kuartik Bezier Eğrisi uygulanmıştır. Eğrilik profili oluşumu, Kuartik Bezier Eğrisinin spesifik özellikleri nedeniyle sadece 3 parametre ile bir optimizasyon problemine dönüştürülmüştür. Özel hedef fonksiyonu ile ilgili optimal çözümü bulmak için sıralı Kuadratik programlama kullanılmıştır. Kaymayı önlemek ve hız-süreklilik ve hızlanma limitlerini sağlamak için doğrusal hız profili oluşturma çerçevesi de önerilmektedir. Bir örnek olarak, sabit ivmeli basit bir profil de sağlanmıştır. Yeteneği ve gerçek zamanlı performansı doğrulamak için şerit tutma ve değiştirme ve yol takibinde simülasyon yapılmıştır. Simülasyon sonuçlarında, önerilen algoritma ile eğrilik sürekliliğinin ve sınırlarının sağlanabileceği gösterilmektedir. Lineer hız profilinin, hız, ivme ve yana kayma kısıtlamaları dikkate alınarak oluşturulabileceği belirtilmiştir. Gerçek zamanlı performansı değerlendirilmiş olup karayolu otonom araçlarına uygulanabilirliği önerilmektedir. Ayrıca, algoritmanın başlangıç parametrelerine karşı duyarsız olduğunun kanıtlandığı ifade edilmiştir.

Yapılan literatür araştırması sonucunda, İHA rota planlama probleminin çözümünde yaygın olarak evrimsel algoritmalar kullanılarak yolun optimize edildiği görülmüştür. Optimize edilen yolun yumuşatılması için Bezier Eğrileri, B-Spline Eğrileri ve Dubins Yolu yöntemlerinin sıklıkla tercih edildiği sonucuna ulaşılmıştır. Bununla birlikte bu üç yöntemin karşılaştırılmasına yönelik bir çalışmaya rastlanmamıştır. Yumuşatma yöntemlerinin karşılaştırılması İHA'nın sınıfı, kısıtları ve görev tanımına göre ele alınacaktır.

BÖLÜM 3

EVRİMSEL ALGORİTMALAR

Charles Darwin tarafından 1859 yılında ilkeleri belirlenen Evrim Teorisi'nin [66], 1950'lerden itibaren bilgisayar bilimcileri tarafından optimizasyon problemlerinin çözümünde kullanılabileceği fikri geliştirildi [67]. Geliştirilen fikir, doğal genetik çeşitlilik ve doğal seleksiyondan esinlenen operatörleri kullanarak belirli bir probleme aday çözümler popülasyonunu değiştirmektir [68]. Evrimsel algoritmalar, bir çözümün daha iyi üretilebilmesi ve daha iyi tahmin edilebilmesi için "en uygun" olanın hayatta kalma ilkesini uygulayan potansiyel çözümler popülasyonunda çalışır. Her jenerasyonda, bireyleri, doğal genetikten alınan operatörleri kullanarak onları bir araya getiren problem alanındaki uygunluk düzeylerine göre seçme süreci ile yeni bir dizi küme oluşturulur. Bu süreç, doğal adaptasyonda olduğu gibi, çevreye daha iyi uyan bireylerin popülasyonlarının, yaratılmış oldukları bireylerden daha fazla evrim geçirmesine yol açar [69]. Şekil 3.1'de basit bir evrimsel algoritmanın yapısı gösterilmektedir. Geleneksel arama yöntemlerinde, problem için bir çözüm adayı önerilir ve bu aday değiştirilerek daha iyi çözümler elde edilmeye çalışılır. Evrimsel algoritmalarda ise bunun aksine, aday çözümleri içeren bir popülasyon oluşturulur ve zamanla bu popülasyon evrimleştirilir. Adaylardan her birinin çözüme yakınlığı, uygulama içinde fonksiyon olarak ifade edilir. Belirlenen çözüm adayı bir kurallar grubundan oluşabileceği gibi parametreleri veya bilgisayar programını da temsil edebilir. Bütün durumlarda da, algoritma her çözüm adayının ne kadar uygun olduğunu hesaplar ve ebeveyn seçimi için uygun



Şekil 3.1: Tek Bir Nüfus Evrim Algoritmasının Yapısı

adayları saklarken diğerlerini popülasyondan çıkartır. Bir sonraki adımda ise daha iyi bir nesil oluşturmak için seçilen ebeveynlere mutasyon ve yeniden yapılandırma uygular. Süreçte en uygun adaylar sonraki nesillere saklandığı için her defasında daha uygun çözümler oluşturarak tekrarlanır.

Belirlenen bir probleme evrimsel algoritma uygulayabilmek için çözüm adaylarının temsil edilme şekli, uygunluğun hesaplanması için bir fonksiyon ve genetik işlemlerin uygulanma yöntemleri belirlenmelidir. Problemin uygun bir zaman içerisinde çözümü için bunların iyi belirlenmesi ve uyum içinde çalışmaları önem arz etmektedir.

Evrimsel Algoritmalar günümüzde ortaya çıkan karmaşık problemlerin çözümleri için hızlı ve doğru sonuçlar üretmektedir.

Çalışmamızda ele aldığımız rota planlamanın birinci kısmı bir tür TSP olmakla birlikte, NP-Hard türünde matematiksel yöntemlerle çözülemeyeceği düşünülen bir problemdir. Bu problemin çözüm yöntemi için yaptığımız literatür araştırmamızda Karınca Kolonisi Optimizasyonu, Yapay Arı Kolonisi Optimizasyonu, Parçacık Sürüsü Optimizasyonu, Diferansiyel Evrim Algoritması ve Genetik Algoritma yöntemlerinin kullanıldığı görülmektedir [70–73]. Bu yöntemler matematiksel yöntemler olmayıp, daha çok sezgisel yöntemler olarak ifade edilmektedirler. Matematikte kullanılan iterasyon yöntemi ile benzerlik göstermektedirler.

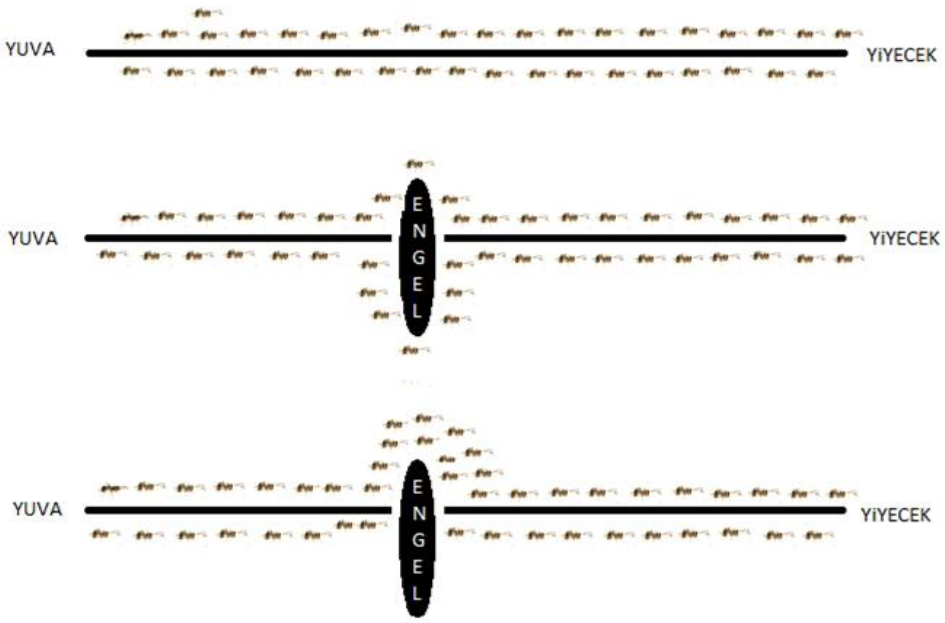
3.1 Karınca Kolonisi Optimizasyonu

Sezgisel yöntemlerden birisi olan, Karınca Kolonisi Optimizasyonu (Ant Colony Optimization-ACO) ilk kez 1991 yılında Marco Dorigo ve arkadaşları tarafından TSP ye bir çözüm yöntemi olarak önerilmiştir [74]. Geliştirilen yöntem TSP üzerine uygulanmış ve olumlu sonuçlar elde edilmiştir. Doğada koloniler şeklinde yaşamlarını sürdüren karıncalar, dünya yüzeyi ile kıyaslandığında çok küçük canlılardır. Bununla birlikte yapılan araştırmalarda yiyeceklerini yuvalarına olabilecek en kısa yoldan ulaştırmayı başarabildikleri görülmektedir. Araştırmacılar karıncaların koloni içerisinde görev dağılımı yaptığını, bu kapsamda öncü olarak belirlenen karıncaların doğaya yiyecek bulmak için rastgele dağıldığını, yiyeceği bulan öncü karıncanın yuvaya taşıma esnasında diğer karıncalar tarafından algılanabilecek bir iz bıraktığını saptamıştır. Kendine has kokusu olan ve doğada zamanla buharlaşan feromon adlı bu salgı diğer karıncaların bu kokuyu takip ederek yiyeceğe ulaşmalarını sağlamaktadır [75]. Doğada feromon salgısı ile karşılaşan bir karınca bunun bir yiyeceğe giden yol olduğunun bilincindedir ve miktarı daha fazla olan bir feromon ile karşılaşmadığı sürece bu izi takip edecektir. Yiyeceğe ulaşan her karınca kendi feromonunu takip ettiği feromonun üzerine ekleyerek yiyeceği yuvaya ulaştıracaktır. Eklenen her karınca yiyeceğe ulaşan yolda bulunan feromon miktarının artmasını sağlayacağından, kokunun miktarı giderek artacak ve daha çok karınca tarafından yolun tercih edilmesini sağlayacaktır.

Bu esnada daha kısa bir yoldan yiyeceğe ulaşmış bir karıncanın izleri daha taze ve feromonu daha az buharlaşmış olacağından yolda karşılaştığı diğer feromon izlerinin yoğunluğu karıncanın yuvasına daha yakın olan yolu bulmasını sağlayacaktır. Feromon yoğunluğu fazla olan, yani yuvaya yakın olan yolun her seferinde tercih edileceği ve diğer yollarda bulunan feromonun zamanla buharlaşacağı düşünüldüğünde; bir süre sonra aynı yönde yiyecek aramaya çıkan tüm karıncalar tek bir yol üzerinde yoğunlaşacaktır.

Karıncaların kullandığı feromon izi yöntemi, bulunan yiyeceği yuvaya taşırken

belirlenen güzergahta bir engel ile karşılaşılması durumunda yuva ile yiyecek arasındaki en kısa yolun bulunmasını ve diğer yollara tercih edilmesini Şekil 3.2'de gösterildiği gibi sağlamaktadır [75]. Yol güzergahında bulunan engel karşısında ilk önce rastgele tercih yapacak olan karıncalar kısa yoldan izlerin daha hızlı oluşması ve feromon yoğunluğunun daha çok artması üzerine kısa olan yolu tercih edecektir. Feromon salgısı sayesinde karıncalar yuvaları ile yiyecek arasındaki en kısa yolu birbirleri ile doğrudan iletişimleri olmadan bulabilmektedir [76].



Şekil 3.2: Feromon Etkisi Yol Belirleme.

ACO algoritmasının yola çıkış fikri incelendiğinde çok basit bit mantık içerdiği görülmektedir. Ancak bunun bir yapay zeka ile birleştirilmesi ve matematiksel olarak ifade edilmesi görüldüğü gibi kolay değildir. Sezgisel algoritmaların genelinde karşılaşılan sorun problemin optimal çözümüne çok yakın sonuçlar ortaya çıkarırken geçen sürenin fazlalığıdır. ACO algoritmasının sözde kodu aşağıda belirtilmiştir [77].

Algorithm 1 ACO Algoritması

```
1: procedure ACO(a, b)
2:     Parametreleri oluştur, feromon izlerini başlat
3:     while sonlandırma koşulu oluşmadıysa do
4:         ÇözümlerOluştur
5:         YerelAramayıUygula                    ▷ isteğe bağlı
6:         İzleriGüncelle
7:     end
8:     end ACO algoritması
```

Uygulanan işlem adımlarının açıklaması aşağıda belirtilmiştir:

ÇözümlerOluştur: Belirtilen problem örneğinde, her karınca bir şehirden başlar, daha sonra şehirleri birer birer dolaşır. Her adımda karıncalar bu olasılıktaki şehirlerden birini seçmek için olasılık dağılım fonksiyonunu hesaplar. Bu rastgele seçime rastsal orantısal geçiş kuralı denir [77] (aynı zamanda sözde rastgele orantısal kural denir) ve iki değerin kombinasyonuna bağlıdır:

Hareketin cazibesi, bu sezgisel bir işlevdir (probleme bağlıdır), o hareketin öncelikli olmasını istemektedir.

Yapay feromonun iz seviyesi, istenilen hareketin sonrasında geleni gösterir, çünkü mevcut şehirde iken bir sonraki olası şehrin seçilmesinin öğrenilen istekliliğini temsil eder.

Bu sayede, her karınca aşamalı olarak, çözülecek örnek için bir çözüm oluşturur.

YerelAramayıUygula: İz seviyesi güncellenmeden önce, geçerli iterasyonda oluşturulan her bir çözüm için yerel arama yöntemleri uygulanabilir. Bu süreç opsiyoneldir ve her problem için değişik sonuçlar doğursa da, karıncalar tarafından elde edilen çözümleri geliştirdiği görülmüştür ve yeni geliştirilen ACO algoritmaları tarafından kullanılmıştır.

İzleriGüncelle: Çözümler oluşturulduktan ve hesaplandıktan sonra iyi çözümlerin bulunduğu yollarda feromon seviyesi artarken (feromon birikmesi), kötü çözümlerin bulunduğu yollarda feromon seviyesi azalır (feromon buharlaşması).

Karınca Sistemi (Ant System):

Belirtilen sistemdeki karıncalar aşağıdaki özelliklere sahiptir [74]:

1- i şehrinden j şehrine giderken arkalarında iz olarak adlandıracağımız bir madde bırakırlar, (i,j) yolu,

2- Karıncaların TSP kuralına uygun tur yapmasını sağlamak için, daha önce ziyaret edilmiş şehirlere tekrar uğranması engellenmektedir.

3- Şehirlerin arasındaki iz miktarındaki artışa ve uzaklığa bağlı olan bir olasılık fonksiyonu ile gitmek için şehir seçilir,

4- $\tau_{ij}(t)$,t zamanda (i,j) yolundaki izin yoğunluğu olsun. Algoritmanın her yinelemesinde iz yoğunluğu Denklem 3.1'de ifade edilen formülle belirtilir [74].

$$\tau_{ij}(t+1) = \rho\tau_{ij}(t) + \Delta\tau_{ij}(t, t+1) \tag{3.1}$$

Burada $\rho, 1-\rho$ izinin buharlaşmasını temsil eden katsayıdır. Birim uzunluk başına düşen feromon miktarını veren formül Denklem 3.2 ile belirtilmiştir.

$$\Delta\tau_{ij}(t, t+1) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t, t+1) \tag{3.2}$$

$\Delta\tau_{ij}^{k}(t, t+1)$, t ve t+1 zamanı arasında k-ıncı karınca tarafından (i,j) yoluna döşenen izi bırakan maddenin (gerçek karıncadaki feromon) birim uzunluk başına miktarıdır. İzin sınırsız miktarda birikmesini önlemek için $\rho < 1$ olarak seçilmelidir. Başlangıçtaki izin yoğunluğu, $\tau_{ij}(0)$, keyfi olarak seçilen küçük değerlere ayarlanmalıdır.

i, ve j şehirleri arasındaki yolun uzunluğu d_{ij} olarak adlandırılır; Öklid düzleminde gerçekleştirilen TSP de i ve j şehirleri arasındaki Denklem 3.3'te belirtilen Öklid uzaklığıdır.

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{3.3}$$

Görünürlük η_{ij} olarak adlandırılır ve $\eta_{ij} = \frac{1}{d_{ij}}$ olarak belirlenir. i şehrinden j şehrine geçiş ihtimali k-ıncı karınca için Denklem 3.4'te belirtilen formülle tanımlanır [77].

$$p_{ij}(t) = \begin{cases} \frac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{h\in\Omega}(\tau_{ih})^{\alpha}(\eta_{ih})^{\beta}} & \text{eğer } j\epsilon\Omega \\ 0 & \text{diğer durumlarda} \end{cases} \tag{3.4}$$

Ω ziyaret edilmeyen şehirlerin kümesidir. İzin yoğunluğu uyarlanabilir hafıza olarak yorumlanır ve sabit bir α parametresi ile düzenlenir. İzin görünürlüğü ise sezgisel fonksiyonu temsil eder ve sabit bir β parametresi ile düzenlenir. α ve β sabit ayarlanabilir parametreleri algoritmanın performansına önemli ölçüde etki etmektedir.

$\alpha = 0$ ise, en yakın şehirlerin seçilmesi ihtimali daha yüksektir; bu klasik stokastik aç gözlü bir algoritma oluşturur.

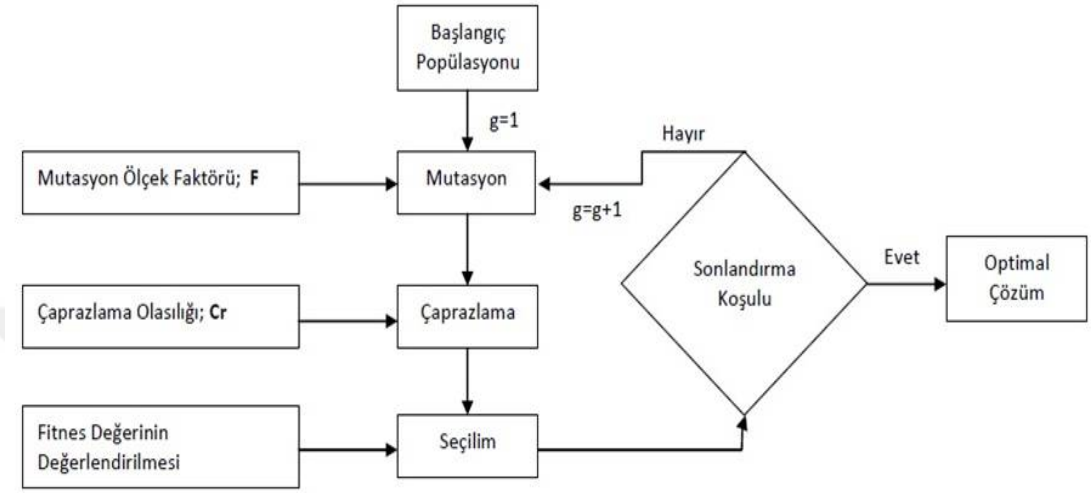
$\beta = 0$ ise sadece feromona bağlı bir algoritma çalışır, bu yöntem en uygun olmayabilecek hızlı bir yol seçimine neden olur [77].

Yapılan literatür araştırmasında ACO algoritmasının bir takım özel araçlar için oluşturulan yol planlamasında karşılaşılan TSP'lerin çözümünde kullanıldığı görülmüştür. Bu çalışmalar incelendiğinde ACO'nun sıklıkla karşılaşılan kullanım alanlarını sıralayacak olursak; şehir içi ve şehirlerarası yollarda TSP'ye bi takım kısıtlar (manzara, dinamik TSP, araç yönlendirme vb) eklenerek oluşturulan problemlerin çözümünde kullanıldığı [78–81], mobil robot-otonom robot yol planlaması ve çarpışmasız robot yol planlaması problemlerinin çözülmesinde etkili olduğu [82–86], insansız kara araçları yol planlaması problemi için model geliştirildiği [87], İnsansız hava araçları ve çoklu insansız hava araçları rota planlama problemi için uygulanabilir olduğu [88], bunun yanı sıra ACO'nun yakınsama hızını ve en uygun yolu bulması konusunda elde ettiği sonucu geliştirmeye yönelik çalışmaların devam ettiği [89–92] görülmektedir. Ayrıca ACO'nun bazı çalışmalarda diğer sezgisel algoritmalar ile birlikte kullanıldığı hibrit uygulamaları yaygın olarak bulunmaktadır [93, 94]. ACO algoritmasının hesaplama süresinin uzun olduğu incelenen çalışmalarda ön plana çıkmaktadır.

3.2 Diferansiyel Evrim (Gelişim) Algoritması

Popülasyon tabanlı sezgisel bir algoritma olan Diferansiyel Evrim Algoritması (Differential Evolution Algorithm-DEA) doğrusal olmayan problemlerin çözümlerinde sürekli parametrelerin söz konusu olduğu durumlar için Price ve

Storn tarafından 1995 yılında geliştirilmiştir [95, 96]. DEA işleyiş mantığı ve operatörleri (mutasyon, çaprazlama ve seçilim) itibariyle GA'yı temel alan popülasyon tabanlı bir sezgisel optimizasyon algoritması çeşididir [96]. DEA'nın işleyiş şeması Şekil 3.3'te gösterilmiştir.



Şekil 3.3: Diferansiyel Evrim Algoritması Şeması.

DEA daha çok sürekli parametrelerin olduğu durumlarda tercih edilmektedir. Programlama kodunun kısa olması da yine tercih edilme sebeplerinden bir tanesidir. Daha iyi çözümler üretmede ana farklılık, GA'nın çaprazlamaya dayandığı ve DEA'nın mutasyon operasyonuna dayanmasıdır [97].

3.2.1 Problem ve Parametreler:

DEA uygulanan problemde kullanılan parametreler aşağıda belirtilmiştir [96]:
Genel bir optimizasyon problemi aşağıdaki gibi tanımlanabilir [98] :

NP :Popülasyon büyüklüğü (kromozom sayısı) $NP \geq 4$, (1,2,3,. . . . . i)

D :Değişken sayısı (gen sayısı) (1,2,3. . . . . j)

Cr :Çaprazlama oranı [0.1,1.0]

G :Jenerasyon (1,2,3,. . . G_{max})

F :Ölçekleme faktörü

$x_{j,i,G}$	:G jenerasyonunda, i kromozomun j parametresi (gen)
$n_{j,i,G+1}$	:Mutasyon ve çaprazlamaya tabi tutulmuş ara kromozom
$u_{j,i,G+1}$	:$x_{j,i,G}$ G den bir sonraki jenerasyon için üretilen kromozom (child-trial)
$r_{1,2,3}$	:Yeni kromozomun üretilmesinde kullanılacak rastgele seçilmiş kromozomlar $r_{1,2,3} \quad \epsilon \{1, 2, 3...NP\}\, r_1 \neq r_2 \neq r_3 \neq i$
$x_j^{(1)}, x_j^{(u)}$	:Değişkenlere ait alt ve üst sınır değerleri.

minimuma indir	f(x)
kısıtlar	$g_t(x) \leq 0$

Burada f(x) amaç fonksiyonu, $g_t(x)$ kısıtlar kümesi ve $x \epsilon R^n$ gerçek değerli tasarım değişken vektörü olup, n tasarım değişkenlerinin sayısıdır. DEA terminolojisinde amaç fonksiyonu genellikle maliyet fonksiyonu (cost-function) olarak adlandırılır.

3.2.2 Başlangıç popülasyonu:

DEA de yeni kromozomların üretilebilmesi için mevcut kromozomun dışında üç adet kromozom gerekmektedir bu nedenle NP kullanıcı tarafından belirlenen kromozom sayısı üçten büyük olmalıdır. D tane parametreden oluşan bir optimizasyon görevi, D boyutlu bir vektörle temsil edilebilir [97]. NP adet kromozom bulunan ve D adet parametreden meydana gelen başlangıç popülasyonu (P0) Denklem 3.5'te belirtilen formül ile üretilir [96].

$$\forall i \leq NP \wedge \forall j \leq D : x_{i,j,G=0} = x_j^{(1)} + rand_j\,[0.1]\,.\left(x_j^{(u)} - x_j^{(1)}\right) \tag{3.5}$$

3.2.3 Mutasyon:

Başlangıç popülasyonu oluşturulduktan sonra, mutasyon işleminde mevcut kromozomun dışında rastgele üç kromozom ($r_{1,2,3}$ kromozomları) seçilir. F parametresi seçilen kromozomların ilk ikisinin farkı ile çarpılır. Bu çarpım daha sonra üçüncü kromozom ile toplanır. Bunun sonucunda çaprazlamada kullanılacak olan $n_{j,i,G}$ kromozomu elde edilmiş olur. $n_{j,i,G}$ kromozomunun elde

edildiği işlemler Denklem 3.6'da belirtilmiştir [96]:

$$\forall j \leq D : n_{i,j,G} = x_{j,r_3,G} + F.\left(x_{j,r_1,G} - x_{j,r_2,G}\right) \tag{3.6}$$

3.2.4 Çaprazlama:

Popülasyonun çeşitliliğini artırmak için ana vektör mutasyon sonucunda üretilen vektör ile çaprazlanarak deneme vektörü $u_{j,i,G+1}$ Denklem 3.7 ile üretilir [97].

$$u_{j,i,G+1} = \begin{cases} n_{j,i,G+1} & \text{eğer } (rnd_j \leq CR)\, veya \quad j = rn_i \\ x_{j,i,G} & \text{eğer } (rnd_j > CR)\, ve \quad j \neq rn_i \end{cases} \tag{3.7}$$

Burada $j = (1, 2, \ldots D)$; $r_j \in [0, 1]$ rastgele sayıdır, çaprazlama sabiti $CR\epsilon\,[0, 1]$ ve $rn_i \in (1, 2, \ldots, D)$ rastgele indekstir.

3.2.5 Seçilim:

Mutasyon ve Çaprazlama işlemlerinden sonra; popülasyon büyüklüğünü sonraki kuşaklar boyunca sabit tutmak için, ebeveyn veya yavru vektörden hangisinin bir sonraki jenerasyona aktarılacağının belirlenmesine yönelik seçilim yapılmalıdır. Seçilim işlemi Denklem 3.8 ile tanımlanır [99]:

$$x_{i,G+1} = \begin{cases} x_{u,G+1} & \text{eger } f\left(x_{u,G+1}\right) \leq f\left(x_{i,G}\right) \\ x_{i,G} & \text{eger } f\left(x_{u,G+1}\right) > f\left(x_{i,G}\right) \end{cases} \tag{3.8}$$

Burada f(x) en aza indirgenecek objektif fonksiyondur.

Yapılan literatür araştırmasında Diferansiyel Evrim Algoritmasının daha yaygın olarak sürekli parametrelerin olduğu problemlerde tercih edilen bir yöntem olduğu görülmektedir. Çok popülasyonlu stratejilerde ve çoklu İHA rota planlama problemlerinin çözümünde başarılı sonuçlar elde edildiği görülmüştür [100–103]. Ayrıca diğer evrimsel algoritmalar ile birlikte kullanılması sonucu elde edilen hibrit uygulamalar bulunmaktadır [104, 105]. Bunlara ek olarak DEA yı geliştirmeye yönelik çalışmalar devam etmektedir [106, 107].

3.3 Parçacık Sürü Optimizasyonu

Parçacık Sürü Optimizasyonu (Partical Swarm Optimizition-PSO); bir grup bilim adamının oluşturduğu, kuş sürüsü ve balık sürüsü içindeki canlıların çeşitli hareketlerinden oluşan bilgisayar simülasyonlarından etkilenen James Kennedy ve Russell Eberhart tarafından 1995 yılında geliştirilmiştir [108]. Sürü halinde hareket eden kuşların beslenme yöntemleri, yiyecek aramak ve bulduklarında sürünün diğer üyelerine haber vermek için kullandıkları, geliştirilen algoritmanın temel mantığını oluşturmaktadır. Genetik Algoritma ile benzer şekilde PSO'da başlangıç olarak rastgele seçilmiş bir çözüm popülasyonu kullanır. Ele alınan problemin çözüm adayları Genetik Algoritmada kromozom olarak adlandırılırken, PSO'da ise parçacık olarak adlandırılmaktadır [109]. Başlangıç popülasyonu olarak da adlandırdığımız parçacık yığını içindeki her birey D boyutlu üç bileşenden oluşur; burada D, arama alanının boyutudur [110]. i-inci parçacığın konumu $X_i = \{x_{i1}, x_{i2}, \ldots, x_{id}, \ldots\ldots, x_{iD}\}$ ile gösterilir, burada $x_{id}\epsilon[l_d, u_d]$, $d\epsilon[1, D]$, l_d, u_d d-inci boyutun sırasıyla alt (lower) ve üst (upper) sınırlarıdır. i-inci parçacığın en iyi önceki pozisyonu (best previous position) , yani en iyi fitness değerini veren pozisyonu, $P_i = \{p_{i1}, p_{i2}, \ldots, p_{id}, \ldots, p_{iD}\}$ ile gösterilir ve pbest olarak adlandırılır. Popülasyondaki en iyi parçacık g indisi ile gösterilir. Pg konumu, yani global en iyi, gbest olarak adlandırılır. i-inci parçacığın hızı $V_i = \{v_{i1}, v_{i2}, \ldots, v_{id}, \ldots, v_{iD}\}$ ile gösterilir, kullanıcı tarafından belirlenen maksimum hız ise $V_{max} = \{v_{max1}, v_{max2}, \ldots, v_{maxd}, \ldots, v_{maxD}\}$ ile gösterilir [111]. Parçacıklar Denklem 3.9 ve Denklem 3.10 ile belirtilen formüllere göre ayarlanır [112].

$$v_{id} = w * v_{id} + c_1 * rand() * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id}) \tag{3.9}$$

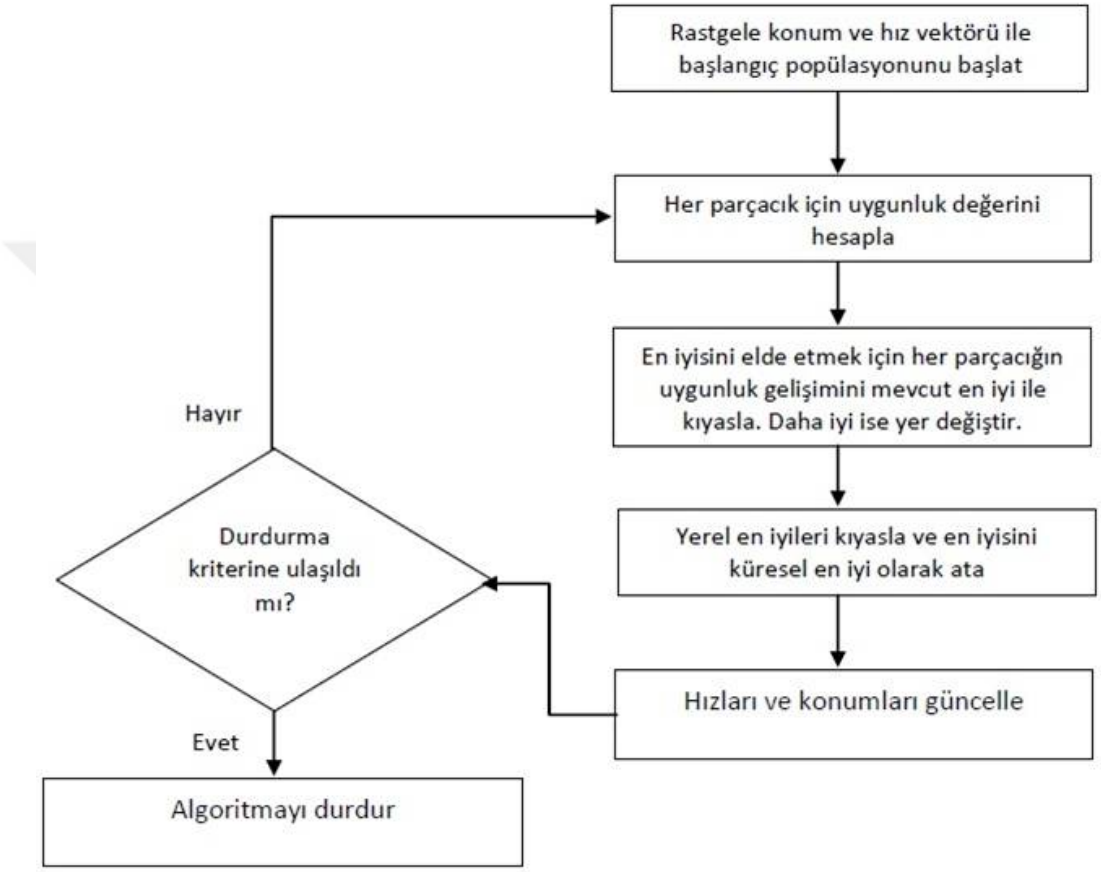
$$x_{id} = x_{id} + v_{id} \tag{3.10}$$

Burada c_1 ve c_2 iki pozitif ivme sabitleri, w eylemsizlik ağırlığı ve rand() ise [0,1] aralığında rastgele bir fonksiyondur.

Denklem 3.9'un ikinci kısmı parçacığın kendi düşüncesini temsil eden "biliş" kısmıdır, üçüncü kısmı ise parçacıklar arasındaki ilişkiyi temsil eden "sosyal"

bölümdür. Denklem 3.9 parçacığın yeni hızının önceki hızına göre durumunu ve mevcut konumunun kendi en iyi deneyimine (pozisyonuna) ve grubun en iyi deneyimine uzaklığını hesaplamak için kullanılır. Daha sonra parçacık Denklem 3.10'a göre yeni bir pozisyona doğru uçar.

PSO'ya ait akış diyagramı Şekil 3.4'te belirtilmiştir.



Şekil 3.4: PSO Algoritmasının Akış Diyagramı.

PSO algoritması kolay uygulanabilir olduğu ve ayarlanması için birkaç parçacık gerektiğinden yaygın olarak kullanılmaktadır ve hızlı bir şekilde geliştirilmektedir [113]. Ancak yakınsama hızının yavaş olması PSO için bir dezavantaj olarak gösterilebilir.

Yapılan literatür araştırmasında PSO'nun yaygın olarak diğer evrimsel algoritmalar ile birlikte hibrit olarak kullanıldığı görülmüştür [114–118]. PSO evrimsel algoritmasının dinamik tabanlı TSP'lerin çözümlerinde daha çok tercih edilen bir yöntem olduğu görülmüştür [114, 119–122]. İncelenen çalışmalar

sonucunda PSO'nun problem türüne göre farklı algoritmaları olduğu ve geliştirmeye açık bir yapısı olduğu gözlemlenmiştir [119, 121, 123–127].

3.4 Yapay Arı Kolonisi Optimizasyonu

Yapay Arı Kolonisi (Artificial Bee Colony-ABC) algoritması Derviş Karaboğa tarafından 2005 yılında bir bal arısı sürüsünün en yakın ve en karlı gıda kaynağını belirlemesi ve onu hasat etmesi davranışının özel akıllı bir davranış olduğu düşüncesinden hareket ederek geliştirilmiştir. ABC gerçek bal arılarının bu davranışını taklit eden bir algoritmadır. Genellikle çok boyutlu ve multimodal optimizasyon problemleri çözmek için tercih edilmektedir [128].

Bal arıları gıda kaynağına ulaşırken, kovanda halihazırda bir gıda kaynağını bulmuş olan, işi olan, toplayıcılar ve henüz bir gıda kaynağı bulamamış olan, işi olmayan, toplayıcıları kullanır. İşi olmayan toplayıcılar iki gruptan oluşmaktadır; bunlar bir gıda kaynağı bulmak için kovandan çıkan "keşifçiler" ve işi olan arıların paylaştığı bilgileri kullanarak yeni bir gıda kaynağı oluşturan "izleyici" arılardır. Arılar arasındaki kolektif bilginin oluştuğu yer, kovanın en önemli bölgesi olan dans alanıdır. Burada işi olan arılar kaynağın yeri ve kalitesi ile ilgili bilgileri izleyici arılara sallanma dansı ile aktarırlar. Muhtemelen çok sayıda dansı izleme şansı bulan izleyiciler en karlı kaynakta çalışmaya karar verir. Daha karlı kaynaklar hakkında daha fazla bilgi dolaşacağından daha karlı kaynakları seçme olasılığı daha fazladır. İşi olan toplayıcılar bilgilerini karlılığı ile orantılı bir şekilde paylaşırlar [128].

ABC algoritması gerçek bal arılarının bu davranışını simüle etmek için tasarlanmıştır. Geliştirilen modelde yapay arıların kolonisi üç arı grubundan oluşur: işçi arılar, izleyiciler ve keşifçiler. Koloninin ilk yarısı yapay işçi arılardan, ikinci yarısı ise izleyiciler tarafından oluşturulmaktadır. Her besin kaynağı için sadece bir tane işçi arı vardır, yani işçi arıların sayısı, kovan çevresindeki besin kaynakları sayısına eşittir. Besin kaynağı tükenen işçi arılar, bir keşifçi olurlar [128].

Algoritmanın ana adımları aşağıda verilmiştir [129]:

Algorithm 2 ABC Algoritması

1: **procedure** ABC(a, b)
2: Gözcüleri ilk besin kaynağına gönder
3: REPEAT
4: İşçi arıları besin kaynaklarına gönder ve nektar miktarlarını belirle
5: İzleyici arıların tercih ettikleri kaynakların olasılık değerlerini hesapla
6: İzleyici arıları besin kaynaklarına gönder ve nektar miktarlarını belirle
7: Arılar tarafından tüketilen kaynakların toplanma sürecini durdur
8: Yeni yiyecek kaynakları bulmak için keşifçileri rastgele olarak arama alanına gönder
9: Şimdiye kadar bulunan en iyi besin kaynağını ezberle
10: UNTIL ▷ gereksinimler karşılandı

Yapılan literatür araştırmasında ABC algoritmasının geliştirilmesi kapsamında yavaş yakınsama sorunun çözümüne yönelik çalışmalar yapıldığı görülmüştür [130–135]. Augerat ve ark. rota planlama problemi kapsamında Kapasitif Araç Yönlendirme Sorununun üstesinden gelmek için uygulanabilir bir yapay ABC algoritması geliştirmişlerdir [136]. Diğer evrimsel algoritmalar ile birlikte hibrit olarak kullanıldığı ve birlikte daha verimli sonuçlar elde edildiği görülmüştür [137, 138].

3.5 Genetik Algoritma

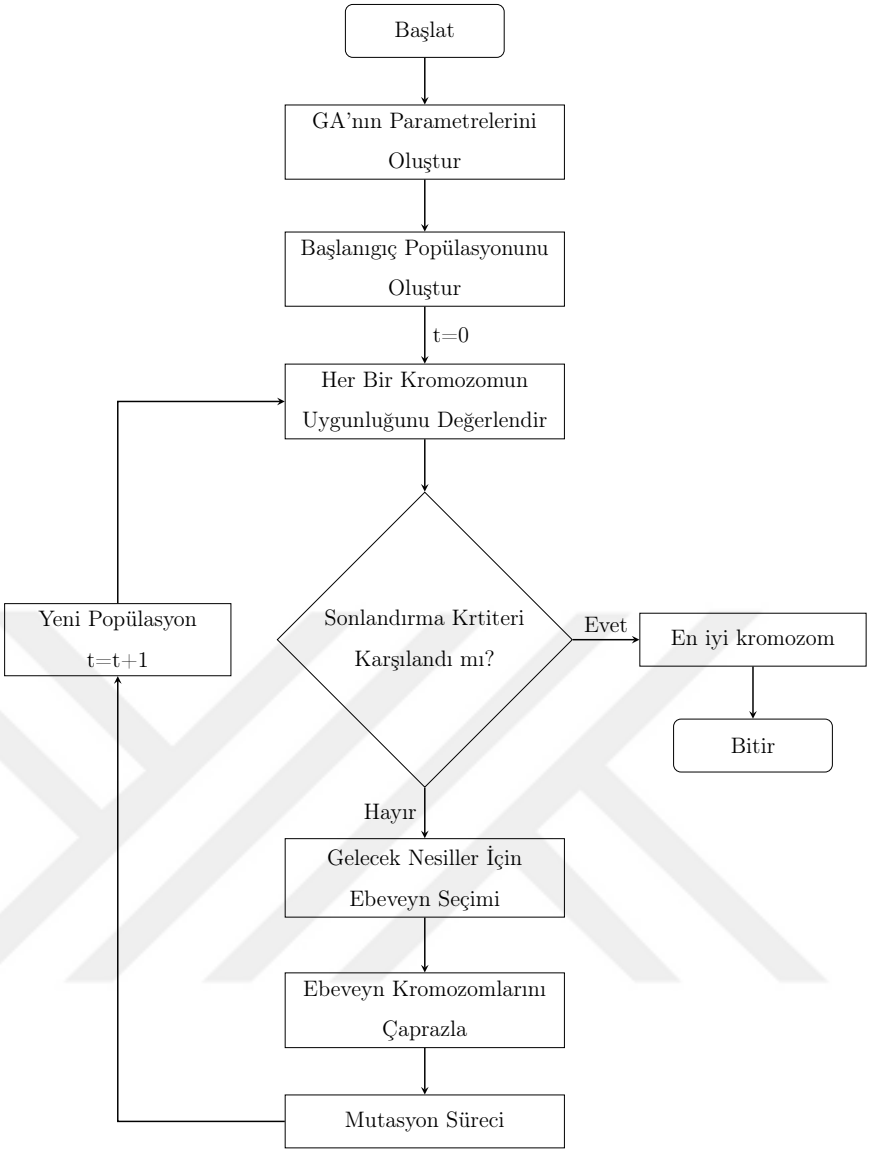
John Holland tarafından 1962 yılında gerçekleştirilen bir uyarlanabilir sistem mantıksal teorisi için taslak çalışması tarafından Genetik Algoritma'nın (Genetic Algorithm-GA) temelleri atılmıştır [139]. J.Holland'ın 1975 yılında yayınlanan Doğal ve Yapay Sistemlerde Adaptasyon (Adaptation in Natural and Artificial Systems) kitabında GA biyolojik evrimin bir uygulaması olarak sunulmuştur [140]. GA ismini genel olarak bir popülasyondaki bireylerin genetik değişim modellerine dayanmasından almaktadır [141]. Bir soruna en uygun çözümü arayan bilgisayar programı olarak evrim tabanlı Genetik Programlama

1992 yılında J.R.Koza tarafından geliştirilmiştir [142]. Holland'ın GA'sı, bir kromozom popülasyonundan (örneğin sıfır ve birlerden oluşan bir diziden) bir çeşit doğal seleksiyonu, genetikten esinlenen çaprazlama, mutasyon ve inversiyon (ters çevirme) operatörleri ile birlikte kullanarak yeni popülasyona taşınması için bir yöntemdir [143].

GA'nın çalışma prensibi doğayı taklit etmektir. GA'nın çalışma yönteminin daha iyi anlaşılması için bir örnek üzerinde anlatalım. Örneğin, herhangi bir zamanda var olan bir tavşan popülasyonunu ele alalım. Bazı tavşanlar diğerlerinden daha hızlı ve daha zekidir. Bu daha hızlı ve daha zeki tavşanların tilkiler tarafından yenilme olasılığı daha düşüktür. Bu nedenle bunların çoğunluğu hayatta kalırlar ve üremeye devam ederler. Bunun yanında yavaş ve daha düşük zekaya sahip tavşanlardan da şanslı olan bazıları hayatta kalırlar. Bu hayatta kalan tavşanlar üremeye başlar. Üreme, tavşanların genetik olarak karışımına yol açar, bazı yavaş tavşanlar hızlı olanlarla, bazı hızlı olanlar hızlı olanlarla, bazı daha az zeki olanlar daha zeki olanlarla vb. Bunun da ötesinde doğa her seferinde tavşan genetik materyalinin bir kısmını mutasyona uğratarak vahşi tavşanın içine koyar. Sonuçta ortaya çıkan yavru tavşanlar ortalama olarak orijinal popülasyonda olduğundan daha hızlı ve daha akıllı olacaktır, çünkü daha hızlı ve daha akıllı ebeveynler tilkilerden kurtulmuşlardır. Bunun yanı sıra benzer bir süreç tilkiler için de söz konusudur. Aksi takdirde zamanla tavşanlar tilkilerin yakalaması için aşırı hızlı ve zeki hale gelirlerdi [144].

3.5.1 Genetik Algoritmanın Yapısı

Klasik GA, çalışma prensibi olarak EA'nın çalışma yöntemini temel almaktadır. Başlangıç olarak GA'nın parametreleri belirlenir. Ardından başlangıç popülasyonu rastgele olarak oluşturulur. Oluşturulan ilk jenerasyon kullanılarak, çaprazlama, mutasyon ve başarısız nesillerin çaprazlamadan çıkartılması sureti ile en iyi çözüm yolu iteratif şekilde aranmaktadır. GA'nın çalışma prensibi Şekil 3.5'te gösterilmiştir [145].



Şekil 3.5: Genetik Algoritma Çalışma Prensibi.

İteratif yöntemlerin genel sorunu olan "zaman" burada da karşımıza çıkmaktadır. Genetik Algoritmanın daha kısa sürede sonuç vermesi konusunda da çalışmalar bulunmaktadır [141].

3.5.2 Kodlama

Teknolojideki gelişmelere paralel olarak birçok problemin çözümü bilgisayar programları yardımı ile yapılmaktadır. İlk olarak problemin bilgisayar

tarafından anlaşılabilir hale getirilmesi gerekmektedir. Bu işleme problemin modellenmesi denir. Belirlenen bir amaç fonksiyonu bilgisayara tanıtılır ve veriler bilgisayar ortamına aktarılır. Genetik algoritmada popülasyonda bulanan her birey kromozomlar olarak ifade edilmektedir. Bu çalışmada her bir kromozom TSP için olası bir çözümü temsil etmektedir. Kromozomları ifade etmek için en çok kullanılan iki yöntem vardır Bunlardan biri değişkenleri 2 tabanlı diziler ile kodlanmış olan ikili kodlama yöntemi ve diğeri ise sürekli değişkenler kullanan yöntemdir. Niceleme sürekli bir değer aralığını birbiriyle örtüşmeyen alt gruplara ayırır. Değişkenlerin ikilik sistem ile ifade edilebilmesi için sürekli değerlerin ikilik sisteme dönüştürülmesi ve gerektiğinde bu işlemin tersi uygulanmalıdır. n'inci değişken p_n'nin ikili olarak kodlanması ve kodun çözülmesi için matematiksel formüller aşağıda gösterilmiştir [146].
Kodlama için formüller Denklem 3.11 ve Denklem 3.12 ile belirtilmiştir.

$$p_{norm} = \frac{p_n - p_{lo}}{p_{hi} - p_{lo}} \tag{3.11}$$

$$gen\,[m] = yuvarla\left\{p_{norm} - 2^{-m} - \sum_{p=1}^{m-1} gen\,[m]\,2^{-p}\right\} \tag{3.12}$$

Kodun çözülmesi için Denklem 3.13 ve Denklem 3.14 ile belirtilmiştir.

$$p_{nic} = \sum_{m=1}^{N_{gen}} gen\,[m]\,2^{-m} + 2^{-(M+1)} \tag{3.13}$$

$$q_n = p_{nic}\,(p_{hi} - p_{lo}) + p_{lo} \tag{3.14}$$

Burada;

p_{norm}	=	Normalize edilmiş değişken, $0 \leq p_{norm} \leq 1$
p_{lo}	=	En küçük değişken değeri
p_{hi}	=	En büyük değişken değeri
$yuvarla\,\{\cdot\}$	=	En yakın tam sayıya yuvarla
p_{nic}	=	p_{norm}'un nicelenmiş versiyonu
q_n	=	p_n'nin nicelenmiş versiyonu

Sürekli değişkenler kullanılan yöntemde; bir kromozom optimize edilecek değişkenlerin bir dizisi olarak tanımlanır. Ziyaret edilmesi planlanan kontrol noktası sayısı aynı zamanda problemin boyutunu belirtir. N_d adet kontrol noktası olması durumunda kromozom 1 x N_d boyutunda bir dizi ile temsil edilir. İHA yollarını tanımlamak için tam sayı kodlanmış kromozom yapısı kullanılır [10].

3.5.3 Başlangıç Popülasyonunun Oluşturulması

GA'nın başlangıç popülasyonu rastgele olarak oluşturulur. Bu popülasyonun her bir üyesi probleme olası bir çözümü kodlar. Bu kodlama ile oluşan değişkenler dizisi kromozomları oluşturur. Başlangıç popülasyonu oluşturulduktan sonra her kromozom uygunluk fonksiyonuna göre bir uygunluk değeri ile değerlendirilir ve bulunan uygunluk değeri o kromozoma atanır. Başlangıç popülasyonu rastgele olarak üretilirken dikkate alınması gereken bazı faktörler vardır. Bunlar: arama alanı, uygunluk fonksiyonu, çeşitlilik, problem zorluğu, seçim baskısı ve birey sayısıdır [147].

3.5.4 Uygunluk Fonksiyonu

Uygunluk fonksiyonu giriş değişkeni olarak belirlediğimiz kromozomları kullanarak bize bir çıktı hesaplayan fonksiyondur. Burada amaç girdileri değiştirerek istenilen çıktıya ulaşmaktır. Optimizasyon hesaplamalarında uygunluk fonksiyonu, hesaplanan veriyi maksimize etme veya minimize etmek için kullanılmaktadır. TSP hesaplamalarında amaç en kısa yolu bulmak olduğu için uygunluk fonksiyonu yerine maliyet fonksiyonu ifadesi de kullanılmaktadır. Optimize edilecek problemin boyutu N_d olsun. Bu durumda $p_1, p_2, \ldots p_{N_d}$ ile verilen N_d değişkeni olan kromozom Denklem 3.15'te belirtilen N_d elemanlı satır vektörü olarak yazılır [146].

$$kromozom = [p_1, p_2, p_3, \ldots, p_N] \tag{3.15}$$

Her kromozomun, uygunluk fonksiyonu f de $p_1, p_2, \ldots p_{N_d}$ noktalarının Denklem 3.16'da belirtilen şekilde hesaplanması ile bulunan bir uygunluk değeri vardır.

$$uygunluk = f(kromozom) = f\left(p_1, p_2, p_3, \ldots, p_{N_d}\right) \tag{3.16}$$

3.5.5 Gelecek Nesiller İçin Ebeveyn Seçimi

Bu aşamada popülasyonda bulunan kromozomlar sonraki nesilleri oluşturmaları için seçilmektedir. Kromozom ne kadar uygun olursa sonraki nesiller için tekrar seçilmesi olasılığı o kadar fazla olur [143]. Ebeveyn kromozomları uygunluk değerleri hesaplanır ve uygunluğu en çok olandan en az olana doğru sıralanır. Bu sıralamada en iyi olanlar sonraki nesiller için seçilirken geriye kalan kromozomlar silinir [146].

3.5.6 Ebeveyn Kromozomlarını Çaprazlama

Silinen kromozom sayısı kadar yeni kromozom (yavru) tutulan ebeveynlerden üretilir. GA'da ebeveynlerden yavru üretilmesi süreci hayvanların çiftleştirilmesine benzemektedir. Elde edilecek olan yavrular (yeni nesiller) ebeveynlerin özelliklerine göre belirlenmektedir. Çaprazlama yapılacak ebeveynlerin seçiminde yaygın olarak kullanılan yöntemlerden kısaca bahsedelim;

Rastgele eşleştirme yönteminde operatör rastgele bir konum seçer ve bu konumdan önceki ve sonraki kromozomların alt dizilerini değiştirerek iki yavru oluşturur. Örneğin iki dizi 101010101 ve 100011111 iki yavru üretmek için her biri dördüncü konumdan sonrasında çaprazlanırsa 101011111 ve 100010101 elde edilir [143]. TSP'de her şehir bir kez ziyaret edilmesi gerekmektedir. Bu durum çaprazlama yapılırken göz önünde bulundurulmalıdır.

Yukardan aşağıya eşleştirme yönteminde ise; uygunluk değerine göre sıralanan ebeveynler ardışık bir şekilde ikililer halinde sırasıyla eşleştirilir.

Rulet tekeri yönteminde; hesaplanan uygunluk değeri en fazla olan kromozomların eşleştirme olasılığı daha yüksek olacak şekilde katsayı atanır.

Burada dikkat edilmesi gereken husus eşleştirme için tekrar aynı kromozomun seçilmesi durumunun önüne geçilmelidir. Rulet tekeri yönteminin programlanması yukarıdan aşağıya eşleştirme yöntemine göre daha zordur.
Turnuva yönteminde tutulan ebeveynlerden küçük bir altküme seçilir ve bu seçilen kümedeki uygunluk değeri en yüksek olan kromozom çiftleşme için seçilir. Daha sonra aynı işlem tekrarlanarak ikinci ebeveyn seçilir. Doğadaki çiftleşme rekabeti yöntemine en çok benzeyen yöntem turnuva yöntemidir [146].

3.5.7 Mutasyon Süreci

Mutasyon süreci GA'nın bir noktada sabitlenmesini ve çok hızlı bir şekilde yakınsamasını önlemektedir. Bunun sonucunda sürekli değişik çözümlerin denenmesi sağlanmaktadır. Mutasyon GA'nın uygunluk değerlerini aramasının ikinci yoludur. Kromozomdaki bir noktaya mutasyon uygulanması sonucunda ikilik sistemle yazılan dizide 1 yerine 0 ya da tam tersi 0 yerine 1 yazılır. Sürekli değişkenler olması durumunda ise diziden rastgele seçilen bir çift nokta yer değiştirilerek birbirlerinin yerine yazılır. Örneğin bir ebeveyn kromozomu $A(2596718403)$ olsun bu durumda; Rastgele sayı üreten bir operatör ile iki sayı belirlenir. Belirlenen sayılar rast1:5, rast2:3 olsun. Kromozomda 5 ve 3 sayıları yer değiştirilir, mutasyona uğramış olan kromozom $A'(2396718405)$ şeklinde oluşturulur [21]. Mutasyon sayısı popülasyondan rastgele olarak seçilir. Mutasyon sayısının artırılması mevcut alanın dışında arama yapma özgürlüğünü artırır. Mutasyonların en iyi çözümler üzerinde değişiklik yapılmasına izin verilmez. En iyi çözümler değiştirilmeden sonraki nesillere aktarılması işlemi, seçicilik (elitizm) olarak adlandırılmaktadır. Bu durumda mutasyona uğrayacak kromozom sayısı belirlenirken, popülasyonun en iyi çözümünün dışında kalan bireylere mutasyon uygulanır [146]. Farklı problem türlerine uygulanmak üzere farklı mutasyon operatörleri geliştirilmiştir. Mutasyon operatörlerinden başlıcaları; Çevirme (Flipping), Ters Çevirme (Reversing) ve Karşılıklı Değişim (Interchanging) olarak sıralanabilir [148].

3.5.8 Genetik Algoritmaların Parametreleri

GA'nın performansını belirleyen ana hususlar GA'nın parametrelerini oluşturur. Bunlar Seçilen Popülasyonun Büyüklüğü, Ebeveynlerin Çaprazlama Oranı, Mutasyon Oranı ve Ebeveyn Seçim Oranıdır. GA'nın parametrelerinin belirlenmesi problem çözümünde önemli bir aşamadır. Çözülmesi planlanan problem göz önünde bulundurularak parametrelerde değişiklik yapmak mümkündür. Geliştirilen GA'nın istenen düzeyde yakınsama yapmaması durumunda GA'nın parametrelerinde değişiklikler yapılmalıdır. Bazen de ikili sistemden sürekli sisteme geçiş, veya tam tersi daha iyi yakınsama için bir çözüm olabilir [146].

Seçilen Popülasyonun Büyüklüğü: Seçilen popülasyonun büyüklüğü GA'nın uygulayacağı yöntemlerin seçiminde belirleyicidir. Örneğin büyük bir popülasyon ile çalışılması durumunda, ebeveynlerin çaprazlanması için sıralama yöntemi çok fazla zaman alacağı için turnuva yöntemi tercih edilmektedir. Optimal popülasyon büyüklüğü ile ilgili çalışmalar bulunmaktadır. GA küçük popülasyon büyüklüğü ve göreceli olarak daha yüksek mutasyon oranları ile daha iyi çalışmaktadır. Popülasyon büyüklüğü arttıkça mutasyon oranı azaltılmaktadır [146].

Ebeveynlerin Çaprazlama Oranı: Çaprazlama oranı yavru kromozomların oluşturulmasında kullanılacak ebeveyn sayısını belirlemektedir. Çaprazlama oranı sıfır ile bir arasında bir sayıdır. Çaprazlama oranının yüksek olması durumunda iyi genlerin bir kromozomda toplanması engellenmiş olur. Düşük bir çaprazlama oranı olması durumunda ise yeterli sayıda yavru oluşmayacaktır [146]. Çaprazlama operatörünün amacı iyi özellikleri sonraki nesillere aktarmaktır. Çok karmaşık bir çaprazlama sonucunda istenenin aksine sonuçlar ortaya çıkması da ihtimal dahilindedir.

Mutasyon Oranı: Popülasyonda mutasyona uğrayacak parça sayısı mutasyon oranı ile belirlenmektedir. Mutasyon oranı farklı çözümlerin bulunmasını sağlayacak kadar büyükse daha çok yol deneneceği için çözüme katkı sağlayacaktır. Bunun yanında daha küçük mutasyon oranları ise mevcut bölgeden daha iyi yararlanılmasını sağlamaktadır. Genel olarak sürekli

parametreler kullanılan GA'lar yüksek mutasyon oranlarına sahiptir [146]. Popülasyon büyüklüğü ile mutasyon oranı birbiri ile ters orantılı olarak yakınsamaya etki etmektedirler.

Ebeveyn Seçim Oranı: Belirlenen ebeveyn kromozomlardan gelecek nesillere aktarılması için seçilen kısmını belirtir. Sonraki nesillere kalacak kromozom sayısı belirlenirken kesin bir kural yoktur. Ancak seçilecek kromozom sayısının yarı yarıya olması algoritmanın daha sağlıklı çalışması için önerilmektedir [146].

Yapılan literatür araştırmasında GA'nın TSP çözümünde yaygın olarak kullanıldığı görülmektedir [149]. GA'nın gelişime açık bir evrimsel algoritma olduğu ve geliştirilmesine yönelik çalışmalar yapıldığı bu durumda elde edilen Geliştirilmiş Genetik Algoritma (Modified Genetic Algorithm-mGA) olarak adlandırıldığı görülmüştür [150–156]. TSP'nin uygulama alanları olan; araç rota planlama [157, 158], insansız sualtı araçları için rota planlama [159, 160], robot için rota planlama [152], İHA için rota planlama [161–163] problemlerinin çözümünde kullanıldığı görülmüştür. GA'nın hızını artırmaya yönelik çalışmalar yapıldığı gözlemlenmiştir [164, 165]. GA'nın diğer evrimsel algoritmalar ile birlikte hibrit olarak kullanıma uygun olduğu, bunun sonucunda hibrit kullanımla ilgili çalışmaların sık olarak karşılaşılan bir yöntem olduğu görülmüştür [159, 166–175].

Evrimsel Algoritmaların temel çalışma prensibinin; uygun bir zaman aralığında en iyi sonuca yakın bir çözüm geliştirmek olduğu belirtilebilir [153]. İncelenen çalışmaların sonucunda; TSP'nin çözümünde kullanılan yöntemler arasında GA'nın oldukça popüler bir uygulama alanı olduğu görülmüştür. Elham D. ve Arash M. tarafından TSP çözümü üzerine yapılan çalışmada GA diğer çözüm yöntemleri ile karşılaştırılmıştır. Geliştirilen GA'nın PSO ve ACO algoritmalarından daha uygun çözümler ürettiği belirtilmiştir [176]. İHA rota planlama probleminin ilk aşamasında karşılaştığımız TSP'yi çözmek için en uygun yöntemin GA olduğu değerlendirilmiştir. Geliştirilen yöntemin ayrıntıları 5'inci Bölüm'de açıklanmıştır.

BÖLÜM 4

ÖNERİLEN SİSTEM

Bu çalışma bünyesinde insansız bir hava aracının uçuşunun gerçekçi ortamda simüle edilebilmesi ve gerçek uçuş rotasının farklı yumuşatma teknikleri ile hesaplanabilmesi için bir sistem önerilmiştir. Önerilen sistem temelde 2 kısımdan oluşmaktadır. İlk kısımda bir evrimsel optimizasyon algoritması yaklaşımı ile İHA'nın dolaşacağı yol hesaplanmakta, daha sonrada bu yol matematiksel yöntemler kullanılarak farklı yumuşatma algoritmaları ile yuvarlanmaktadır. Bu kısımda, bahsedilen 2 farklı sistemin geliştirme detayları verilecektir.

4.1 Genetik Algoritma ile Yol Hesaplama

Evrimsel Algoritmalar çok farklı problemlerin çözümünde kullanılan bir iteratif yaklaşım olup NP-Hard olarak adlandırılan ve parametre sayısının artması ile sistemin karmaşıklığı ve çözümünün üssel olarak arttığı tipten problemlerin çözümü için uygun olan bir yaklaşımdır. (3'üncü Bölüm içerisinde konu hakkında detaylı bilgi verilmiştir).

Bu algoritmalar kullanılarak özellikle çok geniş arama uzayına sahip problemlerin çözümü amaçlamaktadır. Çözüm derken hedeflenen nokta, en iyi çözümün bulunmasından ziyade kabul edilebilir çözümün bulunmasıdır. Burada kabul edilebilirlik, kişiden kişiye veya problemden probleme değişebilecek bir noktadır. Örneğin bazı problemlerde hesaplama süresine göre karar verilirken,

bazı problemlerde iterasyon sayısına göre veya bazı problemlerde sistemin belirli bir iterasyon dahilinde daha fazla sonuç üretip üretememesine göre değişmektedir.Önerilen sistem içerisinde öncelikle Gezgin Satıcı Problemi yapısına uygun olarak bir İHA rota planlaması modellenmektedir. Bir İnsansız Hava Aracının görevini yerine getirmesi için üzerinden geçmesi gereken bazı kontrol noktaları bulunmaktadır. Bu noktaların tam üzerinden geçmek bir amaç olarak görülebileceği gibi, belirli mesafe uzağından da geçme bir amaç olarak değerlendirilebilmektedir. İHA operatörleriyle/uzmanlarıyla yapılan değerlendirmeler sırasında, yerde gizli olan belirli sensörlerden veri toplanması amacıyla veya belirli bölgelerde kamera yardımı ile gözetleme yapılabilmesi açısından İHA'nın rastgele olarak değil de belirli bir plan dahilinde, gözlenecek alanı maksimum kapsayacak şekilde dolaştırılması amaçlanmaktadır.

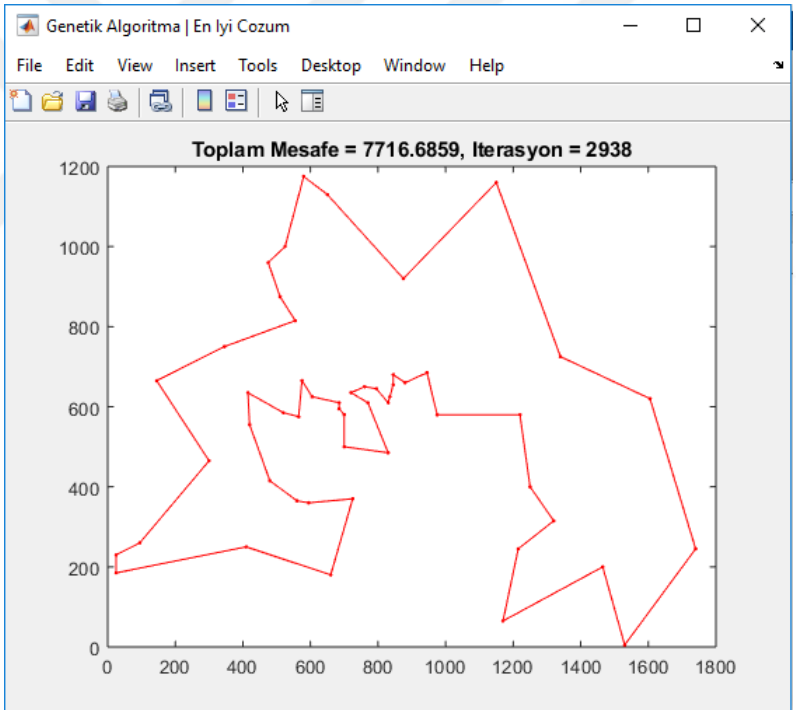
Bu nedenle geliştirilen sistem içerisinde öncelikle kontrol edilecek noktalar sisteme tanımlanmalıdır. Bu işlem temelde iki farklı şekilde yapılabilir. 1'inci yol kontrol noktalarının fare yardımı ile bir harita üzerine işaretlenmesi, 2'nci yol ise bu noktaların koordinatlarının elle girilerek sabit bir text tabanlı dosya üzerinde saklanmasıdır. Bu sistem içerisinde farklı testlerin daha dinamik yapılabilmesi açısından kontrol noktalarının bir metin tabanlı dosya içerisinde saklanması ve dolaşı yolunun ona göre oluşturulması amaçlanmıştır.

```
NAME: lin318
TYPE: TSP
COMMENT: 318-city problem (Lin/Kernighan)
DIMENSION: 318
EDGE_WEIGHT_TYPE: EUC_2D
NODE_COORD_SECTION
1 63 71
2 94 71
3 142 370
4 173 1276
5 205 1213
6 213 69
7 244 69
8 276 630
9 283 732
10 362 69
11 394 69
12 449 370
13 480 1276
14 512 1213
15 528 157
16 583 630
```

Şekil 4.1: TSP Verisi Örneği

Metin tabanlı haritanın saklanması açısından farklı formatta dosyalar hazırlanabilir. Önerilen sistem içerisinde literatür ile uyumlu bir yapı oluşturma adına TSPLIB kütüphanesi yapısı tercih edilmiştir (https://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/) . Bu sayede önerilen sistem farklı sayıda ve şekildeki kontrol noktası dağılımı içinde kullanılabilecektir. Örnek bir Veri dosyası formatı Şekil 4.1'de sunulmaktadır.

Genetik Algoritma Yardımı ile rota bulma problemi MATLAB 2019a programı ile çözülmüştür. MATLAB dili sahip olduğu Grafiksel Kullanıcı Arabirim olanaklarının yüksek olması, kolay yazılım geliştirme imkanı sağlaması sebebi ile tercih edilmiştir. Geliştirilen yazılım Şekil 4.2'deki gibi bir grafik arabirim üzerinden kullanılmaktadır.



Şekil 4.2: MATLAB Grafik Kullanıcı Arayüzü

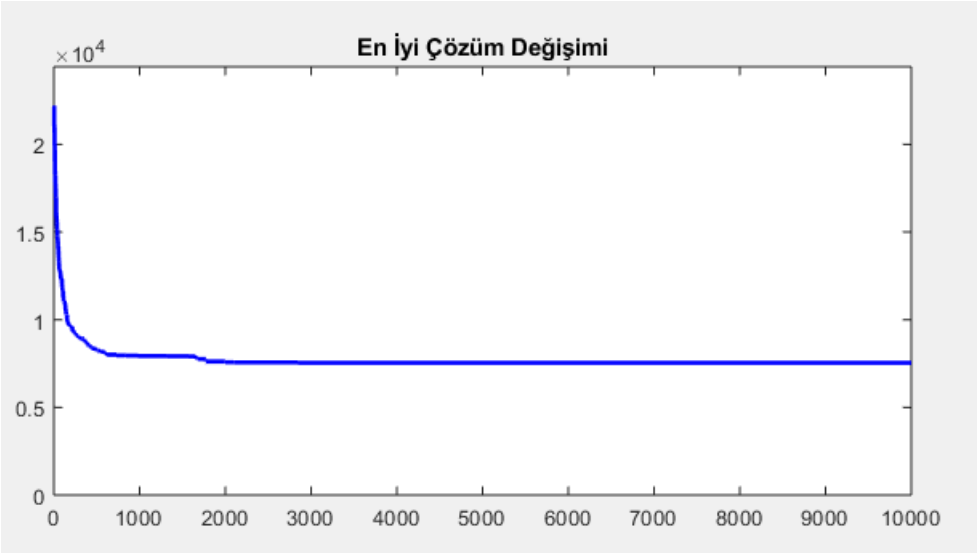
Örnek ekran içerisinde "Berlin52.tsp" dağılımına uygun bir rota planlaması yapılmıştır. Aynı rota Tez içerisindeki değişik testlerde de kullanılmaktadır. Yapılan bu planlama sayesinde dolaşılacak kontrol noktaları bir dizi içerisinde saklanmaktadır. İlgili dizi başka bir programa kolay veri transfer edilebilmesi ve görsellik iyileştirmelerinin yapılması açısından öncelikle bir metin tabanlı

dosyaya çevrilmektedir. Sonrada yumuşatma işlemlerini yapacağımız 2'inci MATLAB programı içerisinde kullanılmaktadır. Örneğin yukarıdaki çalışma için seçilen kontrol noktaları sıralaması şu şekildedir.

[26 27 28 12 25 4 6 48 24 37 39 40 38 15 5 46 44 34 35 36 49 32 1 22 18 31 23 20 50 16 29 30 2 7 42 21 17 3 45 19 41 8 9 10 43 33 51 11 52 14 13 47]

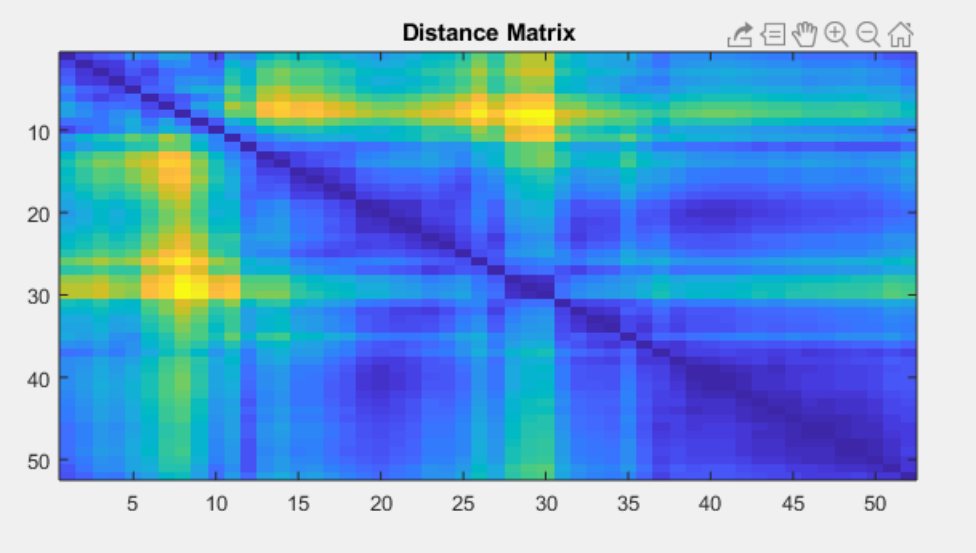
Bu sıralamadan da görüleceği üzere ilk önce 26 numaralı kontrol noktası ziyaret edilecek, sonra 27 ve 28 numaralı kontrol noktalarına sırasıyla uğranacaktır. En son 47 numaralı nokta ziyaret edildikten sonra ise döngü yapısının kurulması açısından yeniden 26 numaralı noktaya dönülecektir. Bu döngüsel çözüm yaklaşımı gerçek dünyada karşılaşılan İHA uçuş planlarına da birebir uymaktadır. Bir İHA belirli bir noktadan uçuşa başladıktan sonra hedef kontrol noktalarını dolaştıktan sonra yeniden harekete başladığı noktaya dönerek iniş yapmak durumundadır. Bu sayede döngüsel yapı oluşmaktadır. Döngü yaklaşımından dolayı ilk başlangıç noktasının hangi kontrol noktasından başladığının bir özelliği bulunmamaktadır. Örneğin, uçuşa başlanan nokta 36 numaralı kontrol noktasında ise de döngüsel rota uzunluğu yine aynı olacaktır.

Önerilen sistem içerisinde rota eniyilemesi iterasyonlarla sağlanmaktadır. İterasyona bağlı olarak bulunan yol uzunluklarının grafiği Şekil 4.3'te gösterilmektedir.



Şekil 4.3: Genetik Algoritma ile İteratif İyileştirme

Burada iyileşmeyi aslında kullandığımız Uzaklık Matrisi (Distance Matrix) ile sağlanmaktadır. Önerilen sistem için mevcut Uzaklık Matrisi değerleri de Şekil 4.4'te sunulmuştur.



Şekil 4.4: Uzaklık Matrisi Değerleri

4.2 Kodlama

Genetik Algoritma ile yapılan optimizasyon/eniyileme işlemlerinde aslında en kritik nokta problemin uygun şekilde kodlanmasını içermektedir. Burada geliştirilen kod (kromozom) bir çözüm önerisini göstermektedir ve her kromozom değişkenlerin diziliminden oluşmaktadır. Bu değişkenler "gen" olarak adlandırılır ve çözülmesi istenen problemin karmaşıklığına göre bu genlerin (değişkenlerin) sayısı değişmektedir.

Problem uzayımızda İnsansız Hava Aracının belirli sayıda kontrol noktasının üzerinden geçmesi amaçlanmaktadır. Eğer gezilecek nokta sayısı 10 ise değişkenlerimizin sayısı da 10 olarak belirlenmiştir. Bu değişkenlerde dolaşılacak noktaların indisini ifade etmektedir. Permütasyon kodlama olarak adlandırılan bu modelin örnek dizilimi şu şekilde gösterilebilir.

5-7-9-2-1-6-0-4-8-3

Burada ilk değişkenin/genin değeri 5, son değişkenin/genin değeri ise 3 olarak belirlenmiştir. Buradaki değişken ayısının 10 tane olarak belirlenmesi permütasyon kodlamanın daha iyi anlaşılması içindir. Gerçek bir problem örneğinde bu değişken sayısı on binlerle ifade edilebilmektedir.

Geliştirdiğimiz sistemde örnek noktalar bir metin tabanlı dosyadan alınmaktadır ve bu dosya içerisinde ilgili noktanın indis değeri ve iki boyutlu düzlemdeki koordinatları yer almaktadır. Dosyadan okuma sırasında bunlar MATLAB içerisinde matrislere atanmakta ve gereken hesaplamalar bu matrisler üzerinden yapılmaktadır. Ancak aynı zamanda bu değerlerin rastgele üretilmesi ve bu şekilde kullanılması da sistemin ilk geliştirimi sırasında kullanılan bir modeldir. MATLAB içerisindeki "rand" fonksiyonu uygun parametrelerle çalıştırılabilmektedir. Örneğin 70 nokta değeri (0..100) arasında aşağıdaki kod satırı ile sağlanabilmektedir.

defaultConfig.xy = 100*rand(70,2);

4.3 Popülasyon Sayısı ve Başlangıç Popülasyonu

Popülasyonda yer alan kromozom sayısı belirlenmesi gereken önemli parametrelerdendir. Eğer kromozom sayısı artarsa çözüm çeşitliliği artar ve daha iyi çözüme ulaşma olasılığı artar, ancak işlem süresi de misliyle artacaktır. Eğer bu sayıyı çok düşük tutarsak yerel optimuma takılma sorunu ile karşı karşıya kalınır. Bu da kromozomların yeni çözüm üretememesini ve en iyi çözüme fazla yaklaşılamamasını ifade etmektedir. Gelişen teknolojiler ve bilgisayarların performanslarının artması neticesinde popülasyon sayısının artırılması tercih edilmektedir. Özellikle bazı paralel hesaplama ortamlarının kullanılması da bu işlem süresini azaltabilmektedir. MATLAB programlama dili de bu paralel hesaplama ortamını desteklemektedir. Eğer Bilgisayarda Graphical Processing Units (GPU) desteği varsa bu ortamın gücünden faydalanma imkanı vardır. Ancak tez çalışması kapsamında bu paralel hesaplama işlemi amaçlanmamıştır. Popülasyondaki kromozom sayısı 100 olarak belirlenmiş ve bu sayı ile kısa zamanda iyi çözümlere ulaşıldığı görülmüştür. Bu

değer atama işlemi parametrik olarak yapılmakta ve "defaultConfig.popSize = 100;" komutu bunun için kullanılmaktadır. Özellikle işlem gücünün yüksek olduğu bilgisayarlarda bu değerin yüksek atanabilmesi gerekmektedir.

Başlangıç popülasyonunun oluşturulması, Genetik Algoritma'da çözüme yakınsama süresi açısından önemlidir. Oluşturulan popülasyon ne kadar güzel çözüm üretiyor ise algoritmanın en iyi çözüme yakınsaması o kadar hızlanmaktadır. Literatür incelendiğinde ilk popülasyonun oluşturulmasında En Yakın Komsu (EYK) algoritması veya benzerlerini kullanarak farklı popülasyon oluşturma tekniklerinin olduğu görülmektedir. EYK modelinde önce rastgele bir nokta belirlenir ve daha sonra buna en yakın olan nokta dolaşı listesine eklenir. Sonra bu listeye en yakın alan 3'üncü nokta eklenir ve bu süreç iteratif olarak tüm noktalar eklenınceye kadar devam eder. Bu işlem MATLAB de aşağıdaki şekilde kodlanmıştır.

```
1 pop(1,:) = (1:n);
2 for k = 2:popSize
3     pop(k,:) = randperm(n);
4 end
```

Her ne kadar EYK algoritması ilk popülasyonu kaliteli şekilde oluştursa da yapılan incelemelerde fazla tercih edilmediği görülmektedir. Bunun sebebi ise Genetik Algoritma'nın asıl gücünü iterasyondan almasından kaynaklanmaktadır. Yeni ve daha iyi çözümlere ileriki iterasyonlarda ulaşılması amaçlanmaktadır. İterasyon sayısının artması hesaplama süresini otomatik olarak artırsa da, EYK algoritmasının çalışması ciddi zaman ihtiyacı doğurduğu için aslında zamansal açıdan kayıp olmamaktadır. Yapılan çalışma kapsamında rastgele popülasyon üretilmesi tercih edilmiştir. Rastgele popülasyon üretilmesinde temelde bir rastgele numara üretilir. Bu numara kromozomun ilk geni olarak atanır. Daha sonra küme içerisinden çıkarılır ve kalan değerler arasından rastgele bir değer daha seçilir. Kromozoma eklenir ve küme içerisinden çıkarılır. Bu süreç tüm değerlerden bir kromozom oluşturulana kadar iteratif olarak devam eder.

4.4 Uygunluk Fonksiyonu

Uygunluk fonksiyonu temelde oluşturulan kromozomun ne kadar iyi bir çözüm olduğunu ölçen bir fonksiyondur. Fonksiyona parametre olarak kromozom yani çözüm önerisi gönderilir. İlgili kromozomun gen değerlerine bakarak çözüm kalitesinin ölçülmesi amaçlanmaktadır. Mevcut problem ortamı için uygunluk fonksiyonu olarak kromozomdaki genlerin sıralı olarak aralarındaki uzaklıkların toplamının alınması amaçlanmıştır. Bu nedenle önce ilk kontrol noktası ile 2'nci kontrol noktası arasındaki mesafe değeri alınır. Daha sonra bu değere 2'nci kontrol noktası ile 3'üncü kontrol noktası arasındaki mesafe eklenir ve bu işlem iteratif olarak devam eder. Tüm noktalar bittikten sonrada en son kontrol noktası ile ilk kontrol noktası arasındaki mesafe de eklenerek fonksiyon tamamlanır. Kontrol noktalarının koordinat değerleri metin tabanlı dosyadan alındığı için biliniyor demektir. Geriye bu koordinatlar arasındaki Öklid uzaklığının ölçülmesi kalmaktadır. Genetik Algoritma'nın çalışma süreci boyunca bu hesaplama birçok kez ve birbirini tekrar edecek şekilde yapılacağı için kontrol noktaları arasında mesafeler en başta hesaplanarak bir Distance Matrix-Uzaklık Matrisi içerisinde tutulur ve doğrudan bir tablodan değer alımı şeklinde yapılır.

4.5 Ebeveyn Seçimi

Genetik Algoritma'da ebeveyn seçimindeki öncelikli amaç uygunluk fonksiyonu iyi olan kromozomun genlerini bir sonraki nesile/popülasyona aktarma olasılığının yüksek olmasıdır. Bu anlamda farklı algoritmalar geliştirilmiştir. Bir önceki kısımda belirtildiği üzere olasılık hesabına en uygunu olarak rulet tekeri ebeveyn seçimi görülmektedir. Ancak bu modelin kendi içerisindeki dezavantajı işlem yoğunluğu olan bir algoritma olmasıdır. Özellikle her çözüm üretiminde 2 kez bu algoritmanın çalışmasından dolayı geliştirilen sistem içerisinde tercih edilmemiştir. Onun yerine Turnuva seçimi tercih edilmiş ve bu sayede rastgele seçilen 2 adet kromozom içerisinden en iyisinin tercih edilmesi amaçlanmıştır.
Her ne kadar rulet tekeri tekniği kadar adil bir dağılıma sahip olmasa da,

ebeveyn seçimindeki ana amaç olan "iyi kromozomların seçilme olasılığının daha iyi olması" ana kriterini sağladığı ve az sayıda hesaplama gücü ihtiyacından dolayı tercih edilmiştir.

4.6 Çaprazlama İşlemi

Silinen kromozom sayısı kadar yeni kromozom (yavru) tutulan ebeveynlerden üretilir. GA'da ebeveynlerden yavru üretilmesi süreci hayvanların çiftleştirilmesine benzemektedir. Elde edilecek olan yavrular (yeni nesiller) ebeveynlerin özelliklerine göre belirlenmektedir. Çaprazlama işlemi evrimsel olarak çözüm üretmek için Genetik Algoritma yaklaşımında sıkça kullanılan bir yöntemdir. Çaprazlama işlemi için 2 adet ebeveyn seçilmesi ve daha sonrada bu ebeveynlerin belirli genlerini (çözümün parçalarını) yeni nesile aktarmaları beklenmektedir. Bunun için kullanılan farklı algoritmalar yukarıda izah edilmiştir.

Gerçeklemesinin kolay olması açısından 1 noktadan çaprazlama yapılması en çok tercih edilen çaprazlama modelidir. Evrimsel algoritmalarda amaç çözümlerdeki iyileştirmelerin iterasyona bağlı olarak yapılması hususundadır. O nedenle iterasyon içerisinde fazla zaman kullanacak operasyonlardan temelde kaçınılmaktadır. Tek noktalı çaprazlama işleminde kromozomdaki gen sayısı kadar rastgele bir değer üretilerek bu değerden önceki genler birinci ebeveynden, sonraki kromozomlar ise ikinci ebeveynden alınmaktadır.

Genetik Algoritma ile üretilen yeni çözüm önerileri isteğe bağlı olarak farklı mutasyon operatöründen/operatörlerinden geçirilerek çözüm kümesine (popülasyona) katılırlar. Her ne kadar farklı çaprazlama operatörleri var olsa da, proje içerisinde evrimsel iyileştirmeye odaklandığı için diğer operatörlerin geliştirilmesi amaçlanmamıştır.

4.7 Mutasyon İşlemi

Mutasyon süreci GA'nın bir noktada sabitlenmesini ve çok hızlı bir şekilde yakınsamasını önlemektedir. Çaprazlama operatörünün kullanılması sebebi ile

popülasyon içerisinde popülasyonun tamamı veya belirli kısmı dar bir çerçevede sıkışabilir. Bu durumda alternatif çözümler bulunabilmesi için mevcut noktalardan tamamen ayrı başka noktalara sıçrayarak yeni değerleri denemek gerekmektedir. Bu işlem mutasyon operatörü ile sağlanmaktadır. Geliştirilen sistem içerisinde 3 farklı mutasyon operatörü kullanılmaktadır. Bunlar "Flip, Slide ve Swap" operatörleridir. Bu operatörlerin kullanım için seçim algoritması aşağıda gösterilmektedir.

```
1 for k = 1:4
2     tmpPop(k,:) = bestOf4Route;
3     switch k
4       case 2
5          tmpPop(k,I:J) = tmpPop(k,J:-1:I);
6       case 3
7          tmpPop(k,[I J]) = tmpPop(k,[J I]);
8       case 4
9          tmpPop(k,I:J) = tmpPop(k,[I+1:J I]);
10      otherwise
11      end
12  end
```

Swap operatöründe 2 tane rastgele değer üretilmektedir. Bu değerler 0 ile ziyaret edilecek nokta sayısı arasındadır. Bu indislerdeki değerlerin yer değiştirilmesi ile mutasyon sağlanır. Slide operatörü kaydırma anlamı taşımaktadır. Yine 2 tane rastgele değer üretilerek bu değerler arasındaki noktalar 1 sağa kaydırılır. Flip operatöründe ise üretilen indis değerleri arasındaki değerler tam olarak ters çevrilmektedir. Bu operatörün Swap operatöründen farkı ise daha az sayıda bağı kırmasıdır.

Geliştirilen sistem genişletilebilir olarak tasarlanmıştır. İstenmesi durumunda buradaki mutasyon işlemlerin türleri rahatlıkla değiştirilebilir ve yeni operatörlerde eklenebilir. Farklı operatörlerin kullanılması ile popülasyon içerisinde çeşitliliğin artırılması amaçlanmaktadır.

4.8 Sonlanma Kriteri

Genetik Algoritma iteratif bir çözüm üretmektedir. Her yeni iterasyonda yeni bir nesil/popülasyon oluşturmakta ve bu oluşturduğu çözümlerde daha iyiye gitmeyi amaçlamaktadır. Ancak bu iterasyonun ne zaman biteceği programcı tarafından verilmesi gereken bir karardır. Her hangi bir limit konulmazsa sonsuz döngü durumunda çözüm üretilmemesi durumu ile de karşılaşılabilir. Literatürdeki uygulamalarda farklı sonlanma kriterlerinin kullanıldığı görülmektedir. Bunlardan en çok tercih edileninin iterasyon sayısına bağlı olduğu görülmektedir. Bu nedenle geliştirilen sistemde de bu değer parametrik olarak tanımlanmıştır. Mevcut sistemde bu değer "defaultConfig.numIter = 1e4;" kodu ile 10,000 olarak belirlenmiştir.

Değerin çok büyük olması durumunda gereksiz hesaplama yapılması ve zaman kaybedilmesi durumu ile karşılaşılacaktır. Değerin küçük atanması durumunda ise iyi çözümlere erişme olanağı ortadan kalkacaktır. Bu nedenle parametrik olarak bu değerin atanması tercih edilmiştir. Özellikle daha kompleks problemler için döngü sayısının artırılması uygun olacaktır.

4.9 Oluşturulan Çözümün Çizilmesi ve Yumuşatılması

Tez kapsamında geliştirilen yazılım 2 kısımdan oluşmaktadır. Birinci kısımda Genetik Algoritma yardımı ile parametrik olarak bir dosyadan alınan kontrol noktası değerlerine göre minimum dolaşı yolu üretilmekte, ikinci aşamada ise, bu oluşturulan dolaşı yolunun farklı yumuşatma algoritmaları sayesinde yumuşatılması planlanmaktadır.

Üst tarafta anlatılan genetik algoritma yapısı ile üretilen dolaşı yolu, koordinatları ile bir text dosya içerisinde saklanmaktadır. "transfer.txt" olarak adlandırılan bu text dosyasının formatı Şekil 4.5'te gösterilmektedir. Bu şekilden de görüleceği üzere dolaşı yolu sıralı olarak verilmektedir. Dolaşı kendi içerisinde çemberi tamamlayacağı için, yani başlangıç noktasına İHA'nın yeniden dönmesi gerektiğinden, başlangıç noktasının hangi kontrol noktasından

başladığının bir önemi yoktur.

vor.m | inceleme.m | tsp_ga.m | output.m | transfer.txt

1	58	2097.00	981.00
2	67	1917.00	687.00
3	28	1795.00	962.00
4	93	1787.00	1009.00
5	1	1380.00	939.00
6	47	1393.00	1368.00
7	63	1115.00	1052.00
8	6	984.00	965.00
9	49	938.00	955.00
10	90	742.00	1025.00
11	75	839.00	620.00
12	92	1187.00	706.00
13	8	1286.00	525.00
14	42	1323.00	280.00
15	89	1429.00	134.00
16	31	1256.00	61.00
17	80	1178.00	100.00
18	56	953.00	268.00
19	97	776.00	392.00
20	4	457.00	334.00
21	65	327.00	265.00
22	26	178.00	24.00

Şekil 4.5: Transfer Edilen Dosya Örneği

Programın yumuşatma kısmı alınan bu metin tabanlı dosyayı, ilk değer kontrol nokta numarası, ikinci değer ilgili noktanın X koordinatı, 3'üncü değer ise ilgili noktanın Y koordinatı olacak şekilde yorumlayarak dolaşılacak yolu çizer. Bu çizim sırasında tercih edilen yumuşatma algoritmasına göre gerekli yumuşatma işlemlerini de yapar. Bu yumuşatma işlemlerinin detayları bir sonraki kısımda izah edilmiştir.

BÖLÜM 5

YUMUŞATMA YÖNTEMLERİ

5.1 Bezier Eğrisi

Bezier Eğrileri bir mühendis olan Pierre Bezier tarafından araçların kaportalarını tasarlamak için 1962 yılında geliştirilmiştir [177]. Polinomlar veya Kübik Spline'lar gibi diğer eğri türlerinin aksine, Bezier Eğrisi, eğriyi tanımlamak için kullanılan kontrol noktalarının dışbükey örtüsünün, noktaları kapsayan en küçük dış bükey kümenin, tamamen içinde yer alır. Kontrol noktalarının değiştirilmesi ile Bezier Eğrisi çevrilebilir ve döndürülebilir [178]. Yöntemde belli sayıda kontrol noktası kullanılarak eğri parçasına yaklaşım uygulanır. Eğriler ilk ve son kontrol noktalarının dışındaki noktalardan geçmemektedir. n-inci mertebeden bir Bezier Eğrisi n+1 kontrol noktası ile üretilir. Bezier Eğrileri, kendilerini üreten kontrol noktalarından oluşan konveks kontrol çokgeninin içinde kaldığından, konveks kombinasyonlar olarak adlandırılan yapı ile üretilmektedirler [179].

Bezier Eğrisi, matematiksel olarak, Bernstein polinomları kullanılarak ifade edilir. Bu durumda, n'inci dereceden baz fonksiyon, kontrol noktaları i ile parametize edilmek üzere, Denklem 5.1'de belirtilen formül ile gösterilir [180].

$$B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i} \tag{5.1}$$

Burada, $0 \leq t \leq 1$ olacak şekilde belirlenmiştir. $\binom{n}{i}$ ise Denklem 5.2 ile verilen kombinasyon formülü olup binom açılımından gelen katsayıların hesaplanması

için kullanılmaktadır.

$$\binom{n}{i} = \frac{n\,!}{(n-i)\,!i\,!} \tag{5.2}$$

Bezier Eğrisi, C(t), cebirsel olarak, P_i i'nci kontrol noktası ve $B_{i,n}$ ilgili Bernstein baz fonksiyonu olmak üzere, Denklem 5.3'te belirtilen şekilde formüle edilir.

$$C(t) = \sum_{i=0}^{n} P_i B_{i,n}(t) \tag{5.3}$$

Yeteri kadar yüksek derece Bezier Eğrileri kullanılarak çeşitli şekiller gösterilebilir. Ancak polinom tarzında ifade edilen Bezier Eğrilerinin derecesi artırıldıkça karmaşıklığı da artacaktır ve bu nedenle işlem süresi de uzayacaktır. Bezier Eğrilerinin derecesinin artırılması sonucu karmaşıklığın artmasının yanında oluşacak olan yüksek dereceli eğrilerin hesaplanması sırasında yuvarlama hatalarının sonuca etkisi daha fazla olacaktır. Yüksek dereceli Bezier Eğrilerinde, eğrinin oluşturulması için verilen kontrol noktaları ile eğri arasındaki ilişki daha az olacaktır. Bununla birlikte bir kontrol noktasında yapılacak değişiklik eğrinin tamamını etkileyecektir.

İHA için rota planlama probleminin çözümünde Bezier Eğrilerinin kullanılması durumunda yukarıda belirtilen olumsuz etkiler neticesinde beklenen çözümü sağlamayacaktır. Çok sayıda kontrol noktasının bulunduğu böyle bir durumda tek parçadan oluşan yüksek derece bir Bezier Eğrisi kullanmak yerine, daha düşük dereceli birden fazla Bezier Eğrisinin birlikte kullanılmasıyla karmaşık şekillerin oluşturulması daha çok tercih edilmektedir. Özellikle ikinci ve üçüncü derece Bezier Eğrileri kullanarak eğri üretmek, kontrol noktası sayısına göre gerektiğinde hibrit olarak birlikte kullanmak tercih edilmektedir [181].

Denklem 5.1, Denklem 5.2 ve Denklem 5.3 kullanarak oluşturulan Bezier Eğrilerinin formüllerini sırasıyla ifade edecek olursak; Lineer Bezier Eğrisi Denklem 5.4'te, Kuadratik Bezier Eğrisi Denklem 5.5'te, Kübik Bezier Eğrisi Denklem 5.6'da, Kuartik Bezier Eğrisi ise Denklem 5.7'de gösterildiği gibi formülize edilmektedir.

$$B_{1,1}(t) = (1-t)P_0 + tP_1 \tag{5.4}$$

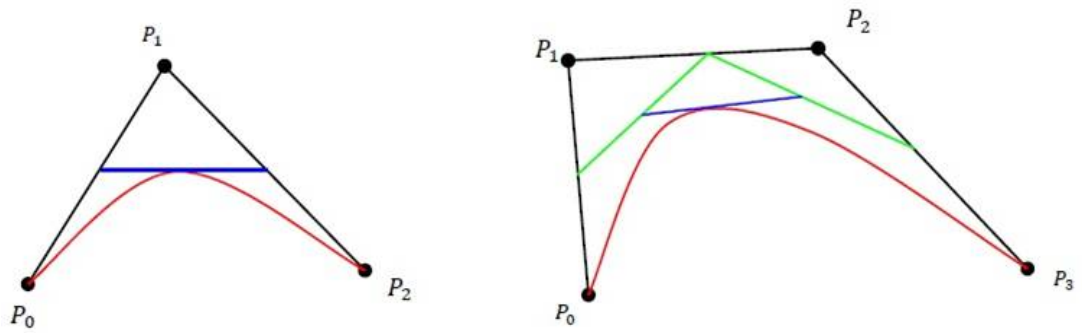
$$B_{2,2}(t) = (1-t)^2P_0 + 2t(1-t)P_1 + t^2P_2 \tag{5.5}$$

$$B_{3,3}(t) = (1-t)^3P_0 + 3t(1-t)^2P_1 + 3t^2(1-t)P_2 + t^3P_4 \tag{5.6}$$

$$B_{4,4}(t) = (1-t)^4P_0 + 4t(1-t)^3P_1 + 6t^2(1-t)^2P_2 + 4t^3(1-t)P_4 + t^4P_5 \tag{5.7}$$

Kübik Bezier Eğrilerinin denkleminde P_i, i=0, 1, 2, 3, noktalarının katsayılarına dikkat edildiğinde sırasıyla $((1-t)+t)^3$ ifadesinin binom açılımı sonucunda elde edilen terimler olduğu görülecektir. Bu özellik N+1 kontrol noktası ile verilen tüm Bezier Eğrileri için geçerlidir. Yani N+1 kontrol noktaları P_i, i=0, 1, ... N, olmak üzere, noktaların katsayıları $((1-t)+t)^N$ ifadesinin binom açılımından elde edilen terimlerdir. Denklem 5.6 ve Denklem 5.7 ayrıntılı olarak incelendiğinde bunun doğruluğu görülecektir.

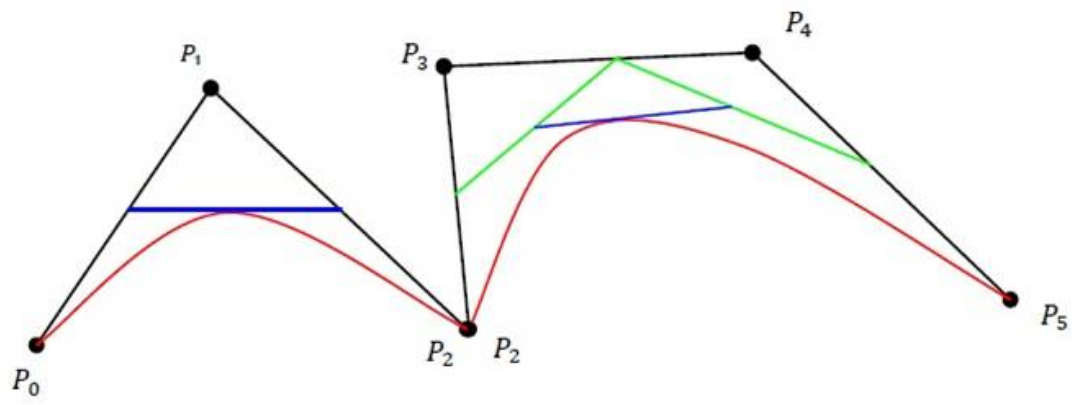
Şekil 5.1'de t $\in$ [0,1] aralığında değiştirilmek suretiyle oluşturulan ve t=0,5 anındaki teğetleri ile verilen Kuadratik ve Kübik Bezier eğri örneklerinin grafikleri bulunmaktadır [24]. Eğriler verilen kontrol noktalarından oluşan konveks çokgenin içinde kalmaktadır.



Şekil 5.1: Kuadratik ve Kübik Bezier Eğrileri

Yukarıdaki şekillerde verilen iki Bezier Eğrisinin birleştirilmesi durumunda oluşan Şekil 5.2 incelendiğinde İHA'nın manevra kısıtlamasına uymadığı görülmektedir. Eğrilerin birleştirilmesi sırasında ilk eğrinin bitiş noktasının sonrasında gelen eğrinin başlangıç noktası olarak belirlenmesi rotanın sürekli

olmasını sağlamaktadır.



Şekil 5.2: İki Bezier Eğrisinin Birleştirilmesi

Ancak süreklilik rota planlamasında tek başına yeterli değildir. Kullanmayı planladığımız Bezier Eğrileri, birleşim noktasında teğetlerinin aynı olması koşulunu da sağlayan C^1 sürekliliğine ve daha düzgün bir birleşime sahip olan Bezier Eğrileridir. Doğal yollarla C^1 sürekliliğine sahip eğrilerin oluşması ihtimali çok düşük olduğundan, suni yöntemler kullanarak, araya nokta eklemek suretiyle, C^1 sürekliliğine sahip eğrinin oluşması sağlanır.
Burada C^i sürekliliğin mertebesini ifade etmekte olup, süreklilik çeşitleri Tablo5.1'de gösterilmiştir.

Tablo 5.1: Süreklilik Çeşitleri

C^{-1}	Eğri süreksizlik içermektedir.
C^0	Eğri süreklidir. Ancak türevin tanımsız olduğu noktalar içermektedir.
C^1	Eğri üzerindeki her noktada birinci türevler tanımlıdır.
C^2	Eğri üzerindeki her noktada ikinci türevler tanımlıdır.
C^n	Eğri üzerindeki her noktada ilk n mertebe türevler tanımlıdır.

Bezier Eğrilerinde Sanal Nokta Ekleme Yöntemi

Bir eğrinin kırılma olmadan pürüzsüz olması türevin tanımlı ve sürekli olması ile elde edilir. Bir noktada türev o noktadaki teğet doğrusunun eğimini

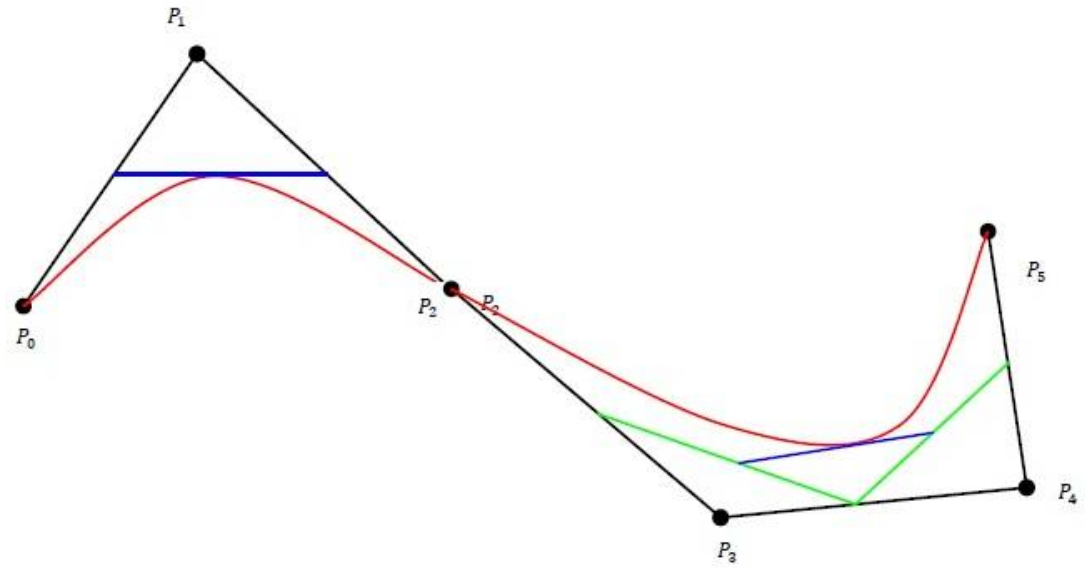
vermektedir. Bir noktada türev tanımlanırken sağdan ve soldan türevler hesaplanır. Sağdan ve soldan türevlerin var olması ve birbirine eşit olması durumunda o noktada türevin tanımlı olduğunu gösterir. Sağdan ve soldan türevlerin tanımlı olmaması veya bir birine eşit olmaması durumu ise o noktada türevin olmadığını gösterir. Bir noktada sağdan türev ve soldan türev Denklem 5.8 ile verilen formül ile hesaplanır. Bezier Eğrileri polinomlardan oluştuğu için tanım kümelerinde türevleri vardır. Bu nedenle bizi ilgilendiren asıl konu iki eğrinin birleştiği noktada var olan bu türevlerin birbirlerine eşit olup olmadığı hususudur.

$$\begin{aligned} f'(x_0{}^+) &= \lim_{x\to x_0{}^+} = \tfrac{f(x)-f(x_0)}{x-x_0} \quad \text{sağdan türev} \\ f'(x_0{}^-) &= \lim_{x\to x_0{}^-} = \tfrac{f(x)-f(x_0)}{x-x_0} \quad \text{soldan türev} \end{aligned} \tag{5.8}$$

İki eğri parçasının birleşiminin gerçekleştiği kontrol noktasında sağdan ve soldan teğetlerin eğimlerinin aynı olması durumunda, sağdan ve soldan elde edilen iki teğet doğrusu aynı noktadan geçeceği için, Denklem 5.9 verilen bir noktası ve eğimi bilinen doğru denklemi oluşturularak kullanılan doğru denklemlerinin bir birine eşit olduğu görülecektir. Burada, $P_i\,(x_i, y_i)$ eğrilerin birleştiği nokta m_i ise bu noktadaki türevdir. Bu durumda oluşacak eğri Şekil 5.3'te sunulmuştur. Şekilde P_1, P_2 ve P_3 noktalarının doğrusal olduğu görülmektedir.

$$y - y_i = m_i\,(x - x_i) \tag{5.9}$$

Şekil 5.3 incelendiğinde C^1 sürekliliği yani birinci türevlerin sürekli olması için iki Bezier Eğrisinin birleştirilmesi esnasında birinci eğrinin bitiş noktasındaki teğet ile ikinci eğrinin başlangıç noktasındaki teğetin aynı doğru olması gerek ve yeter koşuldur. Bu koşul ikinci eğrinin başlangıç noktasından sonra kendisinden önceki iki kontrol noktası ile doğrusal olacak bir sanal nokta eklenmek sureti ile sağlanır.
Birinci eğriyi oluşturan kontrol noktaları: P_i: $i = 0, 1, \ldots, i$
İkinci eğriyi oluşturan kontrol noktaları: P_j: $j = i, i+1, \ldots, k$
İkinci eğrinin birinci eğri ile pürüzsüz bir şekilde birleşmesi için eğriye eklemek istediğimiz sanal nokta ise P_{i*} ile gösterilsin. Bu arada ikinci eğrinin nokta sayısı artık bir artırılacağı için derecesi de artacaktır. Bunu daha önceden planlayıp,



Şekil 5.3: İki Bezier Eğrisinin Düzgün Birleştirilmesi

kontrol noktası sayısı buna göre belirlenmelidir.

Oransal Uzaklıkta Sanal Nokta Ekleme: P_{i-1}, P_i ve P_{i*} noktaları aynı doğru üzerinde olacağından ve sıra bilindiğinden $P_{i-1}(x_{i-1}, y_{i-1})$ ve $P_i(x_i, y_i)$ noktaları arasındaki uzaklık ile $P_i(x_i, y_i)$ ile $P_{i*}(x_{i*}, y_{i*})$ noktaları arasındaki uzaklık bir k katsayısına bağlı olarak ifade edilebilir. Bu ifade düzenlendiğinde Denklem 5.10 elde edilir.

$$P_i - P_{i-1} = h(P_{i*} - P_i) \quad ve \quad P_{i*} = P_i + \frac{P_i - P_{i-1}}{h} \tag{5.10}$$

Sabit Uzaklıkta Sanal Nokta Ekleme: Denklem 5.10 incelendiğinde P_{i*} sanal noktasının P_{i-1}, P_i kontrol noktaları kullanılarak elde edilen $P_i - P_{i-1}$ vektörünün belli bir oranı kullanılarak üretildiği görülecektir. Bu durumda $P_i - P_{i-1}$ vektörünün tüm eğriler boyunca değişiklik göstereceği açıktır. $P_i - P_{i-1}$ vektörünün normuna bölünmesi ile elde edilecek birim uzunlukta bir vektör ise tüm eğri boyunca eklenecek sanal noktaların aynı uzaklıkta olmasını sağlayacaktır. Sabit uzunlukta sanal nokta eklenmesi için oluşturulan eşitlik Denklem 5.11 ile sunulmuştur.

$$P_{i*} = P_i + h^\star \frac{P_i - P_{i-1}}{\|P_i - P_{i-1}\|} \tag{5.11}$$

Denklem 5.10'da h değeri arttıkça eklenecek sanal noktanın uzaklığının azaldığı, Denklem 5.11'de ise $h^\star$ değerinin artması ile eklenecek sanal noktanın uzaklığının arttığı görülmektedir. Elde edilen denklemlerden yararlanılarak her bir eğri segmenti için bir önceki ile birleştiği noktadan hemen sonrasına bir sanal nokta eklenir. Eklediğimiz yeni kontrol noktaları sayesinde eğrilerin birleşim noktalarında bulunan teğetleri aynı olacağından C^1 sürekliliği sağlanır, dolayısıyla pürüzsüz bir eğri elde edilir.

5.1.1 Lineer Bezier Eğrisi

Verilen n tane kontrol noktasından geçmesi istenen kapalı doğru Denklem 5.4 ile verilen Lineer Bezier Eğrisi kullanılarak elde edilebilir. Burada yapacağımız hesaplamanın standart olması için incelenen örneklerde TSP hesaplaması Lineer Bezier Eğrisi kullanılarak gerçekleştirilmiştir. Lineer Bezier Eğrisi matris kullanılarak Denklem 5.12 ile belirtilen şekilde gösterilebilir.

$$B_{1,1}(t) = \begin{bmatrix} t & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \end{bmatrix} \tag{5.12}$$

5.1.2 Kuadratik Bezier Eğrisi

İkinci derece Bezier Eğrileri, Kuadratik (Quadratic) Bezier Eğrileri olarak adlandırılmaktadır. İkinci derece bir Bezier Eğrisini oluşturmak için Denklem 5.5 incelendiğinde 3 tane kontrol noktasının gerektiği görülecektir. Kuadratik Bezier Eğrisi kullanılarak oluşturulan eğrilerin birleştirilmesi esnasında, önce oluşturulan eğri parçasının bitiş noktası bir sonraki eğrinin başlangıç noktası olarak seçilmiştir. İkinci derece Bezier Eğrileri kullanılan 3 kontrol noktasından ortadaki kontrol noktası hariç uçlardaki kontrol noktalarının ikisinden de geçmektedir. Elde edilen eğri parçacığı az sayıda kontrol noktası kullanılarak elde edildiği için noktalarla eğri arasındaki ilişki daha yüksektir. Arada bir kontrol noktasının değiştirilmesi eğrinin sadece ilgili kontrol noktasını içeren kısmının değişmesine neden olacaktır ve büyük çaplı bir değişiklik olmayacaktır.

Denklem 5.5 ile elde edilen formülün düzenlenmesi ile Denklem 5.13 elde edilir.

$$B_{2,2}(t) = (t^2 - 2t + 1)P_0 + (-2t^2 + 2t)P_1 + t^2 P_2 \tag{5.13}$$

Denklem 5.13 ile oluşturulan Kuadratik Bezier Eğrisi matris kullanarak Denklem 5.14'te belirtildiği şekilde ifade edilebilir. Bu gösterim bize MATLAB hesaplamaları sırasında kolaylık sağlayacaktır.

$$B_{2,2}(t) = \begin{bmatrix} t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 1 & -2 & 1 \\ -2 & 2 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \end{bmatrix} \tag{5.14}$$

Kuadratik Bezier Eğrileri Denklem 5.10 ve Denklem 5.11 ile verilen sanal nokta ekleme formülleri kullanılarak C^1 sürekli olması sağlanmaktadır. C^1 sürekli olan Kuadratik Bezier Eğrileri Düzgün Kuadratik Bezier Eğrileri olarak adlandırılmıştır. Sanal noktalar eğriyi oluşturacak kontrol noktalarının arasında kalacağından Kuadratik Eğrinin ziyaret etmediği noktalar sanal noktalar olacaktır. Bu durumda sadece ilk eğri parçasında ziyaret edilmeyen bir nokta kalmaktadır. İlk eğri parçasında yapılan düzeltme işlemi ve ekstra sanal nokta ekleme ile bu kontrol noktasının da ziyaret edilmesi sağlanmıştır.

5.1.3 Kübik Bezier Eğrisi

Üçüncü derece Bezier Eğrileri, Kübik (Cubic) Bezier Eğrileri olarak adlandırılmaktadır. Üçüncü derece Bezier Eğrileri üretmek için derecenin bir fazlası olan 4 tane kontrol noktası kullanılmaktadır. Kübik Bezier Eğrileri, oluşturulurken önce üretilen eğri parçasının bitiş noktası sonraki eğrinin başlangıç noktası olarak seçilmiştir ve bu şekilde eğri sürekli hale getirilmiştir. Üçüncü derece Bezier Eğrileri, her eğri parçasında arada kalan iki kontrol noktasının üzerinden geçmeyecektir. Eğriyi oluşturan kontrol noktası sayısı arttığı için eğrinin kontrol noktaları ile arasındaki ilişki bir miktar azalmış olur. Ancak eğrinin toplam uzunluğu göz önünde bulundurulduğunda daha kısa bir eğri oluşturulmuş olur. Bunun nedeni ise kontrol noktalarından daha az etkilenen eğri daha az sapmaya uğrayacaktır. Oluşturulan Bezier Eğrisinin daha az sapmaya uğraması sonucunda, eğriliği daha fazla yumuşamış olur. Denklem

5.6 ile oluşturulan formülün düzenlenmesi ile Denklem 5.15 elde edilir.

$$B_{3,3}(t) = (t^3 - 3t^2 + 3t - 1)P_0 + (3t - 6t^2 + 3t^3)P_1 + (3t^2 - 3t^3)P_2 + t^3 P_3 \tag{5.15}$$

Denklem 5.15'te belirtilen Kübik Bezier Eğrisi matrisler kullanılarak Denklem 5.16'de belirtilen şekilde düzenlenir.

$$B_{3,3}(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 1 & 3 & -3 & 1 \\ -3 & -6 & 3 & 0 \\ 3 & 3 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix} \tag{5.16}$$

Kübik Bezier Eğrilerinin uç uca eklenmesi ile oluşan eğrinin C^1 sürekliliğini sağlaması için Denklem 5.10 ve Denklem 5.11'de belirtilen formüller kullanılarak eğriye sanal noktalar eklenir. Bunun sonucunda oluşan eğri Düzgün Kübik Bezier Eğrisi olarak adlandırılmıştır. Eklenen sanal noktalar eğri parçasının ziyaret etmediği noktalardan biri olacağı için eğrinin daha fazla kontrol noktasının üzerinden geçmesi sağlanır. Ancak bunun sonucunda eğrinin uzunluğunun artması kaçınılmazdır. İlk eğri parçasına sanal nokta eklenmediği için uzunluğunda herhangi bir değişiklik olmaz.

5.1.4 Kuartik Bezier Eğrisi

Dördüncü derece Bezier Eğrileri, Kuartik (Kuartic) Bezier Eğrisi olarak adlandırılmaktadır. Dördüncü derece Bezier Eğrileri 5 adet kontrol noktası kullanılarak üretilmektedir. Kuartik Bezier Eğrileri kullanılarak oluşturulan eğrinin birleştirilmesi sırasında, önce oluşturulan eğrinin bitiş noktası bir sonraki eğri parçasının başlangıç noktası olarak belirlenmiş ve bu şekilde eğrinin sürekli olması sağlanmıştır. Dördüncü derece Bezier Eğrileri, her eğri parçasında başlangıç ve sondaki kontrol noktalarının üzerinden geçerken arada kalan kontrol noktalarına yakınsamaktadır. Eğrinin derecesinin yükselmesi sonucunda daha fazla kontrol noktası kullanıldığı için eğri ile kontrol noktaları arasındaki

ilişki bir miktar daha azalmış olur. Bunun sonucunda arada kalan kontrol noktalarına daha az yakınsayan eğrinin uzunluğu azalır. Burada dikkat edilmesi gereken husus eğrinin üzerinden geçmediği kontrol noktaları ile eğri arasında oluşan mesafenin belirlenen üst sınırın altında kalması gerekmektedir. Kontrol noktalarından daha az etkilenen Kuartik Bezier Eğrisi, daha yumuşak bir şekil alır. Denklem 5.7'de ifade edilen formülün düzenlenmesi ile Denklem 5.17 elde edilir.

$$\begin{aligned} B_{4,4}(t) &= (t^4 - 4t^3 + 6t^2 - 4t + 1)P_0 + (4t - 12t^2 + 12t^3 - 4t^4)P_1 \\ &+(6t^2 - 12t^3 + 6t^4)P_2 + (4t^3 - 4t^4)P_3 + t^4 P_4 \end{aligned} \tag{5.17}$$

Denklem 5.17'de belirtilen Kuartik Bezier Eğrisi matrisler kullanılarak Denklem 5.18'de belirtilen şekilde düzenlenir.

$$B_{4,4}(t) = \begin{bmatrix} t^4 & t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 1 & 4 & 6 & -4 & 1 \\ -4 & 12 & -12 & 4 & 0 \\ 6 & -12 & 6 & 0 & 0 \\ -4 & 4 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix} \tag{5.18}$$

Kuartik Bezier Eğrilerinin oluşturduğu eğride C^1 sürekliliği Denklem 5.10 ve Denklem 5.11 ile verilen formüller kullanılarak elde edilen sanal noktaların eğriye eklenmesi ile sağlanır. Bunun sonucunda oluşan eğri Düzgün Kuartik Bezier Eğrisi olarak adlandırılmıştır. Sanal noktaların eklenmesi ile eğrinin uzunluğunda bir miktar artış ortaya çıkmaktadır. Ancak C^1 sürekliliği olmayan eğri rota planlamasında kullanılamayacağından bu artışın olması kaçınılmazdır. Elde edilen denklemler 6'ncı Bölüm'de örnek problemlere uygulanmış ve bu işlemin sonucunda ortaya çıkan veriler tablolar altında toplanmıştır. Eklenen sanal noktaların eğrilerin uzunluklarına etkileri azaltılmaya çalışılmış ve sanal noktanın yerini belirleme konusunda ikinci bir yöntem geliştirilmiştir. Geliştirilen bu yönteme göre kendisinden önceki iki kontrol noktasına belirlenen oran ile eklenen sanal noktaların sabit bir büyüklükle eklenmesi sağlanmıştır. Bu durumda kendisinden önceki kontrol noktaları arasındaki mesafe arttıkça eğrinin uzunluğuna etkisi artan kontrol noktaları yerine, eğriye sabit bir

uzunlukta etki eden sanal noktaları elde edilmiştir. Sabit uzunluk kullanılarak eklenen sanal nokta kendisinden önce kullanılan iki kontrol noktası tarafından üretilen vektör, ve bu vektörün normuna bölünmesi ile oluşturulan birim vektör kullanılarak üretilmiştir.

5.1.5 Rasyonel Bezier Eğrileri (Hiperbolik Bezier Eğrileri)

Polinomlar kullanılarak düzgün biçimde gösterilemeyen önemli eğriler vardır. Tüm konikler iki fonksiyonun oranı olan rasyonel fonksiyonlar kullanılarak daha iyi gösterilebilirler. Bezier Eğrilerinin Bernstein polinomları kullanılarak oluşturulduğunu göz önünde bulunduralım. Bu kapsamda Rasyonel Bezier Eğrileri Denklem 5.19'da belirtilen formül ile tanımlanır [182].

$$R(t) = \frac{\sum_{i=0}^{n}(P_i w_i B_i^n(t))}{\sum_{i=0}^{n}(w_i B_i^n(t))} \tag{5.19}$$

Burada w_i değerleri kontrol noktalarının eğrinin oluşturulması esnasında eğriye olan etkisini belirleyen ağırlık katsayılarını ifade etmektedir. Denklem 5.19'da bulunan tüm ağırlık katsayıları için $w_i = 1$ seçilirse tekrar Bezier Eğrisi elde edilir.

w_i değerlerinin kontrol noktasına etkisi incelendiğinde;

$w_i > 1$ ise eğri P_i noktasına daha çok yaklaşır,

$w_i < 1$ ise eğri P_i noktasına normalde olduğundan daha az yaklaşır,

Rasyonel Bezier Eğrileri uçak endüstrisinde önemli bir araç haline gelen konikleri göstermek için kullanışlı bir yöntemdir.

Oluşturulan Bezier Eğrileri başlangıçta ve bitişte bulunan kontrol noktalarının üzerinden geçmektedir. Bu nedenle arada kalan ve eğrinin üzerinden geçmediği kontrol noktasının katsayısı eğride oluşturmak isteyeceğimiz değişikliği belirlemektedir. Rasyonel Kuadratik Bezier Eğrisi üç adet kontrol noktası kullanılarak üretilmektedir. w_0, w_1 ve w_2 katsayılar P_0, P_1 ve P_2 kontrol

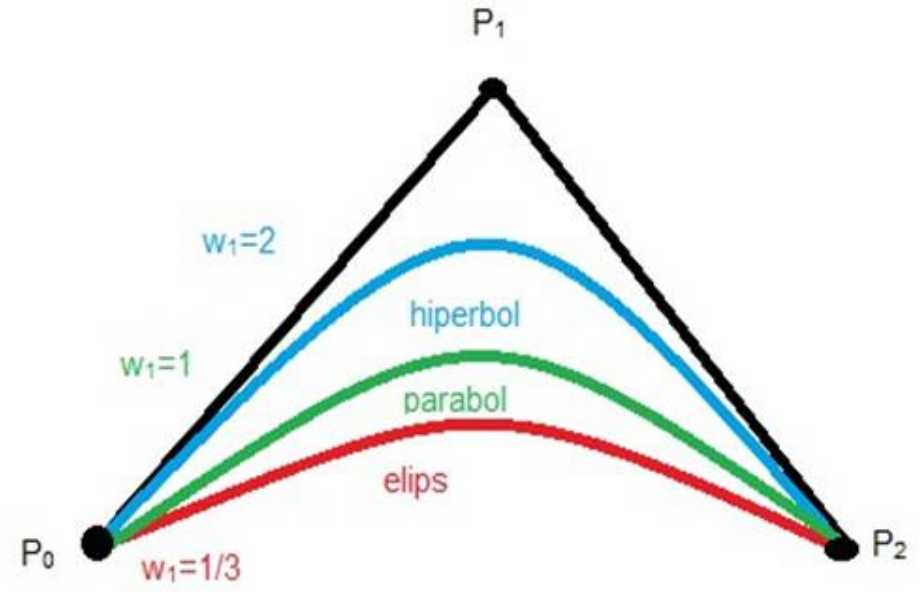
noktaları olmak üzere eğrinin formülü Denklem 5.20'de ifade edilmiştir;

$$c(t) = \frac{P_0 w_0 B_0^2(t) + P_1 w_1 B_1^2(t) + P_2 w_2 B_2^2(t)}{w_0 B_0^2(t) + w_1 B_1^2(t) + w_2 B_2^2(t)} \tag{5.20}$$

$w_0 = w_2 = 1$ seçilirse ve $s = \frac{1}{1+w_1}$ olarak tanımlanırsa;

$$\begin{cases} s = \frac{1}{2} & \text{ise parabolik bir yay oluşur,} \\ s < \frac{1}{2} & \text{ise eliptik bir yay oluşur,} \\ s > \frac{1}{2} & \text{ise hiperbolik bir yay oluşur.} \end{cases}$$

Rasyonel Kuadratik Bezier Eğrilerinde katsayıdaki değişimin oluşturduğu etki Şekil 5.4 ile gösterilmiştir.



Şekil 5.4: Rasyonel Kuadratik Bezier Eğrileri

Rasyonel Bezier Eğrilerine sanal kontrol noktaları eklenerek C^1 sürekliliğine sahip eğriler elde edilir. Belirlenen ihtiyaç doğrultusunda Rasyonel Bezier Eğrileri kullanılabilir. Örneğin, arada kalan bir kontrol noktasına daha yakın geçmesini istediğimiz durumda Hiperbolik Bezier Eğrilerini kullanabiliriz. Ancak

bunun belli bir üst sınır dahilinde olması gerekmektedir. Çok yüksek değerlerde yakınlaştırma yapılması durumunda planlanan yolun eğriliği bozulacaktır.
Rasyonel Bezier Eğrileri kullanılarak rota planlaması 6'ncı Bölüm'de örnek problemler üzerinde incelenmiştir.

5.2 B-Spline Eğrisi

B-Spline (Basis Spline) baz fonksiyonları üzerindeki ilk önemli çalışmalar 70 yıl kadar önce Schoenberg tarafından gerçekleştirilmiş ve bunu Cox ve de Boor tarafından temel algoritmanın geliştirilmesi takip etmiştir [183, 184]. Bilgisayar Destekli Geometrik Tasarım (Computer Aided Geometric Design-CAGD) alanı içindeki B-Spline'ların bu alandaki birçok öncü; Riesenfeld, Boehm, Schumder ve daha sonra gelen birçok araştırmacı tarafından göz alıcı sunum yöntemleri ve uygulanabilir olmaları kanıtlanmıştır [185].
B-Spline Eğrileri de Bezier Eğrileri gibi kontrol noktalarının hepsinden geçmeyen eğrilerdir. Değişik derecelerde kullanılabilirler. Veri noktaları kullanılarak ve birleştirme yapılarak Bezier Eğrileri ile eğri oluşturulurken türevlerin sürekliliğinin her zaman kontrol edilmesi ve sağlanması gerekmektedir. Ayrıca Bezier Eğrilerinde bir kontrol noktasının yeri değiştiğinde tüm eğri bundan etkilenmektedir. B-spline Eğrileri Bezier Eğrilerinin sahip olduğu bu dezavantajların üstesinden gelebilmek için yerel etki alanlarına sahip baz fonksiyonlarına sahiptir. Bu fonksiyonlar kendilerine ait kısımlar dışında sıfıra eşittir. Bu nedenle eğri kendisine komşu birkaç kontrol noktasına göre şekil alır. B-spline Eğrilerinin derecesi ile kontrol noktalarının sayısı birbirlerinden bağımsızdır [186].
k-ıncı mertebeden, verilen n+1 kontrol noktası $P_0, P_1, \ldots, P_n$ için tanımlanan C(u) B-spline Eğrisinin matematiksel ifadesi Denklem 5.21 ile aşağıda belirtilmiştir:

$$C(u) = \sum_{i=0}^{n} N_{i,k}(u) P_i \tag{5.21}$$

Burada u'nun tanım aralığı $0 \leq u \leq n - k + 2$ olacak şekilde belirlenir. n-k+2 sayısı aynı zaman da eğrinin kaç parçadan oluşacağını da belirtir.

k-ıncı mertebeden bir B-Spline kırılma noktalarında en fazla C^{k-2} süreklilik ile k-1 inci dereceden polinomların çeşitli parçalarının bir araya gelmesiyle oluşturulur. k-ıncı mertebeden bir B-Spline Eğrisi, k tane komşu noktanın oluşturduğu dışbükey örtünün içinde kalır. Bu durumda eğri, kendisini oluşturan kontrol noktalarından, başlangıç ve bitiş noktalarına uğramayacağından açık bir eğri oluşur. Bunu önlemek için eğriler oluşturulurken başlangıç ve bitiş noktaları birkaç tekrar olacak şekilde yazılmıştır. Kırılma noktalarının azalmayan bir kümesi $t_0 \leq t_1 \leq \cdots \leq t_{n+k}$ ile baz fonksiyonların parametrelenmesini belirleyen düğüm vektörü Denklem 5.22 ile tanımlanır.

$$T = (t_0, t_1, \ldots, t_{n+k}) \tag{5.22}$$

t_i düğümleri $0 \leq i \leq n+k$ olmak üzere t_i düğümleri Denklem 5.23'te belirtilen formül ile belirlenir;

$$t_i = \begin{cases} 0 & \text{eğer } i < k \quad ise \\ i-k+1 & \text{eğer } k \leq i \leq n \quad ise \\ n-k+2 & \text{eğer } i > n \quad ise \end{cases} \tag{5.23}$$

Verilen bir düğüm vektörü T, birleşmiş B-Spline baz fonksiyonları, $N_{i,k}.(u)$, $k = 1$ için Denklem 5.24'te belirtilen şekilde tanımlanır;

$$N_{i,1}(u) = \begin{cases} 1 & t_i \leq u \leq t_{i+1} \text{ ise} \\ 0 & \text{değilse} \end{cases} \tag{5.24}$$

Düğüm vektörü $k > 1$ ve $i = 1, 2, 3, \ldots n$ için ise Denklem 5.25'te belirtilen şekilde tanımlanır;

$$N_{i,k}(u) = \frac{u - t_i}{t_{i+k-1} - t_i} N_{i,k-1}(u) + \frac{t_{i+k} - u}{t_{i+k} - t_{i+1}} N_{i+1,k-1}(u) \tag{5.25}$$

Bu denklemin tanımsız olmaması için $\frac{0}{0} = 0$ olarak kabul edilmiştir.

B-Spline Eğrilerini oluşturan denklemlerin özellikleri [185]:

Pozitiflik: $t_i \leq u \leq t_{i+1}$ için $N_{i,k}(u) > 0$

Bölgesel Olma: Eğer $u \notin [t_i, t_{i+1}]$ ise $N_{i,k}(u) = 0$

Birimin Parçalanışı (Genlik): $\sum_{i=0}^{n} N_{i,k}(u) = 1$

Süreklilik: $\sum_{i=0}^{n} N_{i,k}(u)$ eğrisi her bir düğüm noktasında C^{k-2} sürekliliğine sahiptir.

B-Spline Eğrisinin kontrol noktalarına aynı zamanda Boor noktaları da denir. Baz fonksiyon $N_{i,k}(u)$ ve üzerinde tanımlandığı düğüm vektörü;

$T = (t_0, t_1, \ldots, t_{k-1}, t_k, t_{k+1}, \ldots, t_{n-1}, t_n, t_{n+1}, \ldots t_{n+k})$,

n+k+1 elemanı ile yani kontrol noktalarının sayısı olan $n+1$ ile eğrinin mertebesi olan k sayılarının toplamı olduğu, Denklem 5.22 ve 5.23 ile verilmişti. Her düğüm aralığı $t_i < u < t_{i+1}$ iki ardışık bağlantı $C(t_i)$ ve $C(t_{i+1})$ arasında bir polinomal eğri üzerine çizilmiştir. Düğüm vektörünü normalleştirme, $[0, 1]$ aralığını örtecek şekilde, bu aralıktaki değişen sayıların yüksek yoğunluğu nedeniyle değişen noktadaki aritmetik hesaplamanın nümerik hassasiyetini artırır

.

5.2.1 Kuadratik B-Spline Eğrisi

İkinci derece B-Spline Eğrileri Kuadratik B-Spline Eğrileri olarak adlandırılır. B-Spline Eğrilerinde mertebe ve derece kavramları birbirine karıştırılmamalıdır. Bunun sonucunda ikinci derece bir denklemden oluşan Kuadratik B-Spline Eğrilerini üretmek için Boor Algoritması denklemlerinde k=3 seçilir. n+1 kontrol noktası için Denklem 5.21 ve Denklem 5.25'te değerler yerine yazıldığında Denklem 5.26'da gösterilen formül elde edilir [187].

$$C(u) = \tfrac{1}{2}(i+1-u)^2 P_i + \tfrac{1}{2}\left[(u-i+1)(i+1-u) + (i+2-u)(u-i)\right]^2 P_{i+1} + \tfrac{1}{2}(u-i)^2 P_{i+2} \tag{5.26}$$

Burada $1 \leq i \leq n-k+2$ ve $i \leq u \leq 1$ dir.

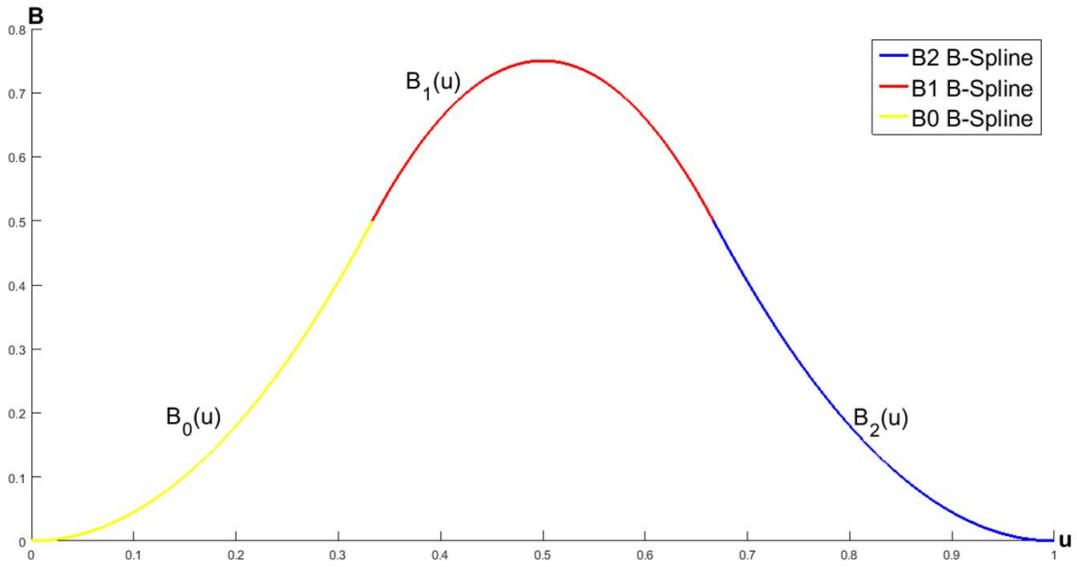
Elde edilen bu denklemde $u' = u - i$ dönüşümü yapılırsa, $0 \leq u' \leq 1$ olacağından standart hale gelir ve bunun sonucunda aşağıda belirtilen Denklem 5.27 elde edilir.

$$C\left(u'\right) = \frac{1}{2}\left[\left(1-u'\right)^2 P_i + \left(-2u'^2 + 2u' + 1\right) P_{i+1} + u'^2 P_{i+2}\right] \tag{5.27}$$

Artık tüm eğri parçaları u' nin $[0,1]$ aralığında değişiminden elde edilecektir. Denklemde tekrar u' yerine u yazılarak Denklem 5.28 elde edilir.

$$C\left(u\right) = \frac{1}{2}\left[\left(1-u\right)^2 P_i + \left(-2u^2 + 2u + 1\right) P_{i+1} + u^2 P_{i+2}\right] \tag{5.28}$$

Kuadratik B-Spline Eğrilerinin hesaplanmasının ikinci bir yolu ise; B-Spline Eğrilerinin özelliklerinin kullanılmasıdır. Kuadratik B-Spline Eğrileri, k=3 seçilerek üretilir. C^1 sürekliliğine sahip 4 kontrol noktası ile üretilen Şekil 5.5'te gösterildiği gibi 3 eğri parçası oluşacaktır. Birim parametre $u \in [0,1]$ olarak belirlenmiştir. Kuadratik B-Spline Eğrileri ikinci derece polinomlardan oluştuğundan her bir eğri parçası Denklem 5.29'da verilen denklemler kullanılarak ifade edilir.



Şekil 5.5: Kuadratik B-Spline Baz Fonksiyonları

$$\begin{aligned} B_0\left(u\right) &= a_0 + b_0 u + c_0 u^2 \\ B_1\left(u\right) &= a_1 + b_1 u + c_1 u^2 \\ B_2\left(u\right) &= a_2 + b_2 u + c_2 u^2 \end{aligned} \tag{5.29}$$

B-Spline Eğrilerinin özellikleri kullanılarak Tablo 5.2 oluşturulur.

Tablo 5.2'de verilen değerlerin yerine yazılması ile elde edilen eşitlikler Denklem

Tablo 5.2: Kuadratik B-Spline Baz Denklemleri

SN	C^0 **Özelliği**	SN	C^1 **Özelliği**
1	$B_0(0) = 0$	5	$B_0{}'(0) = 0$
2	$B_0(1) = B_1(0)$	6	$B_0{}'(1) = B_1'(0)$
3	$B_1(1) = B_2(0)$	7	$B_1{}'(1) = B_2'(0)$
4	$B_2(1) = 0$	8	$B_2{}'(1) = 0$

5.30 ile sunulmuştur. Elde edilen sekiz eşitliğe genlik özelliği $1 = B_0(0) + B_1(0) + B_2(0)$ eklenmiştir.

$$
\begin{aligned}
&1) \quad a_0 = 0 \\
&2) \quad b_0 + c_0 - a_1 = 0 \\
&3) \quad a_1 + b_1 + c_1 - a_2 = 0 \\
&4) \quad a_2 + b_2 + c_2 = 0 \\
&5) \quad b_0 = 0 \\
&6) \quad 2c_0 - b_1 = 0 \\
&7) \quad b_1 + 2c_1 - b_2 = 0 \\
&8) \quad b_2 + 2c_2 = 0 \\
&9) \quad a_0 + a_1 + a_2 = 1
\end{aligned}
\tag{5.30}
$$

Elde edilen denklem sistemi matrisler kullanılarak Denklem 5.31 ile ifade edilir.

$$
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 2 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 2 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 \\
1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
a_0 \\ b_0 \\ c_0 \\ a_1 \\ b_1 \\ c_1 \\ a_2 \\ b_2 \\ c_2
\end{bmatrix}
=
\begin{bmatrix}
0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1
\end{bmatrix}
\tag{5.31}
$$

Denklem sistemi MATLAB programıyla çözüldüğünde elde edilen sonuç Denklem 5.32 ile verilmiştir.

$$\begin{bmatrix} a_0 \\ b_0 \\ c_0 \\ a_1 \\ b_1 \\ c_1 \\ a_2 \\ b_2 \\ c_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{2} \\ \frac{1}{2} \\ 1 \\ -1 \\ \frac{1}{2} \\ -1 \\ \frac{1}{2} \end{bmatrix} \tag{5.32}$$

Elde edilen katsayılar Denklem 5.29'da yerine yazıldığında Kuadratik B-Spline Eğrisi formülleri Denklem 5.33'te gösterilen şekilde elde edilir.

$$\begin{aligned} B_0(u) &= \tfrac{1}{2}u^2 \\ B_1(u) &= \tfrac{1}{2}[1 + 2u - 2u^2] \\ B_2(u) &= \tfrac{1}{2}[1 - 2u + u^2] \end{aligned} \tag{5.33}$$

Elde edilen denklemler yerine yazıldığında Şekil 5.5'te gösterilen ilk eğri parçasının son denklem ile üretildiği görülmüştür. Bunun sonucunda eğrilerin sıralamasının tersi alınmıştır. Başlangıçta eğrilerin sırası rastgele olarak yazıldığı için bu durum normaldir. Bulunan formülün Denklem 5.28 ile aynı olduğu görülmektedir.

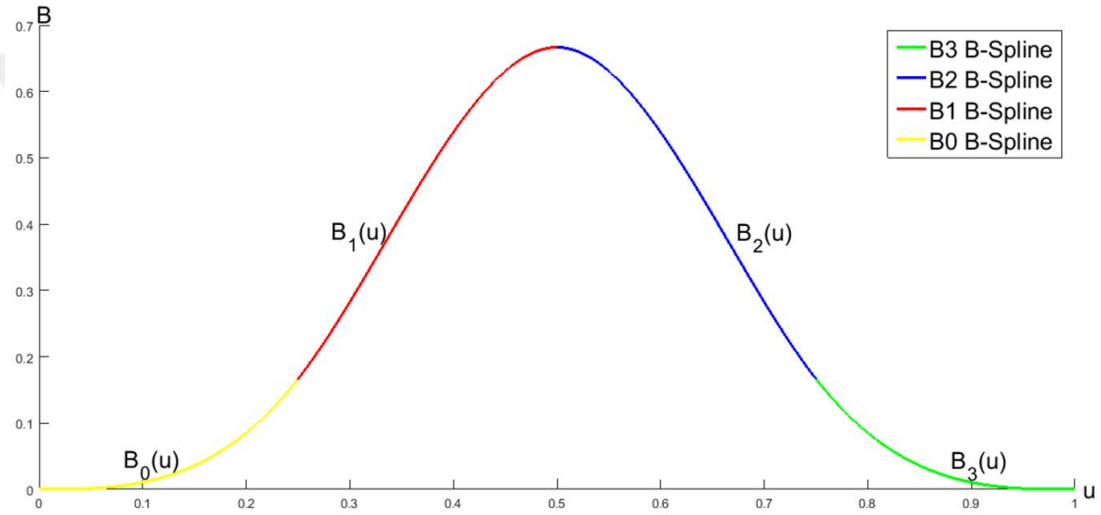
Denklem 5.33 ile belirtilen Kuartik B-Spline Eğrisi matrisler kullanılarak Denklem 5.34'te belirtilen şekilde düzenlenir.

$$C(u) = \frac{1}{2}\begin{bmatrix} u^2 & u & 1 \end{bmatrix}\begin{bmatrix} 1 & -2 & 1 \\ -2 & 2 & 0 \\ 1 & 1 & 0 \end{bmatrix}\begin{bmatrix} P_{i-1} \\ P_i \\ P_{i+1} \end{bmatrix} \tag{5.34}$$

$i \in [1, n-1]$olacak şekilde eğrinin her parçası u değerinin $[0, 1]$ aralığında değişmesiyle elde edilir.

5.2.2 Kübik B-Spline Eğrisi

Üçüncü derece B-Spline Eğrileri Kübik B-Spline Eğrileri olarak adlandırılır. Üçüncü derece bir B-Spline Eğrisi üretmek için k=4 seçilir bunun sonucunda C^2 sürekliliğine sahip 5 kontrol noktası ile üretilen Şekil 5.6 ile gösterildiği gibi 4 eğri parçası oluşacaktır. Burada parametre olarak birim parametre kullanılacaktır, $u \in [0, 1]$. Kübik B-Spline Eğrileri üçüncü derece polinomlar kullanılarak üretildiğini tanımından biliyoruz. Buradan yola çıkarak oluşan her bir eğri parçası Denklem 5.35'te verilen denklemler ile ifade edilir.



Şekil 5.6: Kübik B-Spline Baz Fonksiyonları

$$\begin{aligned} B_0(u) &= a_0 + b_0 u + c_0 u^2 + d_0 u^3 \\ B_1(u) &= a_1 + b_1 u + c_1 u^2 + d_1 u^3 \\ B_2(u) &= a_2 + b_2 u + c_2 u^2 + d_2 u^3 \\ B_3(u) &= a_3 + b_3 u + c_3 u^2 + d_3 u^3 \end{aligned} \tag{5.35}$$

Denklemlerin katsayılarını hesaplamak için B-Spline Eğrilerinin özellikleri kullanılacaktır. Kübik B-Spline Eğrisinin mertebesi k=4 olduğu için eğri, C^0, C^1 ve C^2 sürekliliklerine sahiptir. Bunun sonucu olarak bir eğrinin bittiği noktada diğeri başlamalıdır. Bu durumun daha iyi anlaşılması için Tablo 5.3 düzenlenmiştir.

16 bilinmeyen bulunan denklem sistemimizin çözümü için tabloda verilen 15 denkleme ek olarak genlik özelliği kullanılarak oluşturulan Denklem 5.36 eklenir.

Tablo 5.3: Kübik B-Spline Baz Denklemleri

C^0 **Özelliği**	C^1 **Özelliği**	C^2 **Özelliği**
$B_0(0) = 0$	$B_0{}'(0) = 0$	$B_0{}''(0) = 0$
$B_0(1) = B_1(0)$	$B_0{}'(1) = B_1'(0)$	$B_0{}''(1) = B_1{}''(0)$
$B_1(1) = B_2(0)$	$B_1{}'(1) = B_2'(0)$	$B_1{}''(1) = B_2{}''(0)$
$B_2(1) = B_3(0)$	$B_2{}'(1) = B_3'(0)$	$B_2{}''(1) = B_3{}''(0)$
$B_3(1) = 0$	$B_3{}'(1) = 0$	$B_3{}''(1) = 0$

$$1 = B_0(0) + B_1(0) + B_2(0) + B_3(0) \tag{5.36}$$

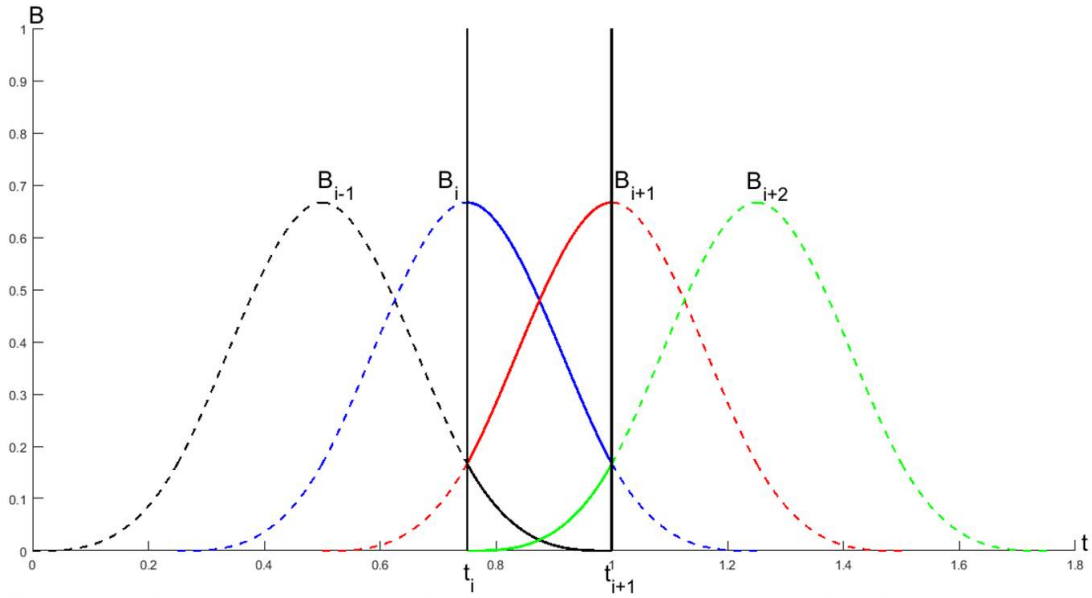
Oluşturulan denklem sistemi MATLAB programı ile çözülmüştür. Elde edilen 16 katsayı Denklem 5.35'te yerine yazıldığında oluşan Kübik B-Spline Eğrisi formülleri Denklem 5.37'de ifade edilmiştir.

$$\begin{aligned} B_0(u) &= \tfrac{1}{6}u^3 \\ B_1(u) &= \tfrac{1}{6}[1 + 3u + 3u^2 - 3u^3] \\ B_2(u) &= \tfrac{1}{6}[4 - 6u^2 + 3u^3] \\ B_3(u) &= \tfrac{1}{6}[1 - 3u + 3u^2 - 3^3] \end{aligned} \tag{5.37}$$

Burada oluşturulan denklemler yerine yazıldığında ilk eğrinin son denklem ile üretildiği, son eğrinin ise ilk denklem ile üretildiği görülmüştür. Bunun sonucunda denklemlerin sırası tersine çevrilmiştir. Daha kullanışlı olması için denklemler matris formunda yazıldığında Denklem 5.38 oluşturulur.

$$C(u) = \frac{1}{6}\begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}\begin{bmatrix} P_{i-1} \\ P_i \\ P_{i+1} \\ P_{i+2} \end{bmatrix} \tag{5.38}$$

Eğrinin i-inci bölmesi o bölgede bulunan baz fonksiyonların oluşturdukları Şekil 5.7 ile gösterilen dört baz fonksiyon eğrisinin lineer toplamı şeklinde ifade edilir.



Şekil 5.7: Kübik B-Spline Baz Fonksiyonları

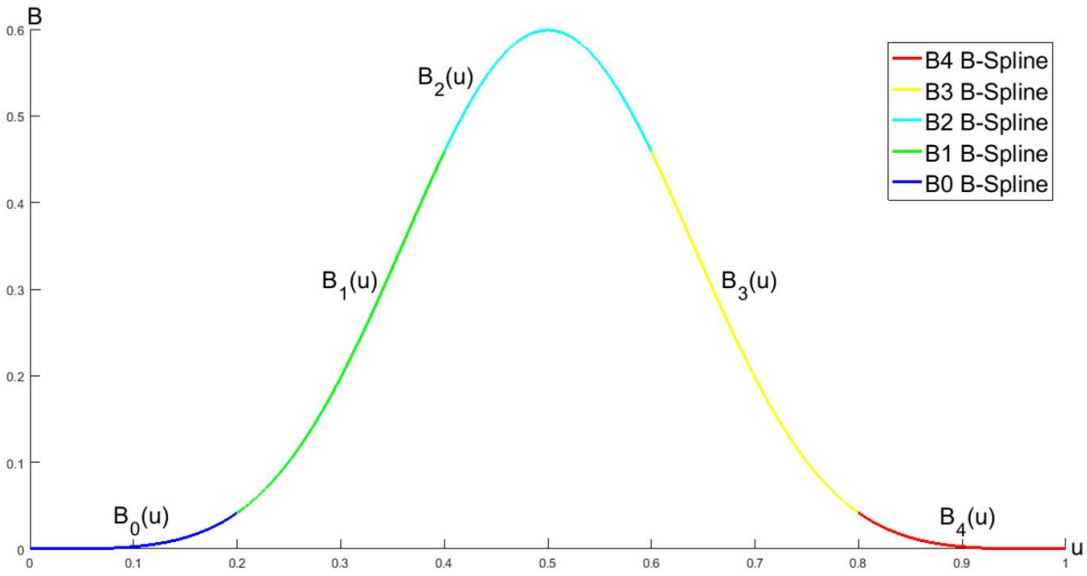
5.2.3 Kuartik B-Spline Eğrisi

Dördüncü derece B-Spline Eğrileri Kuartik B-Spline Eğrileri olarak adlandırılır. Dördüncü derece bir B-Spline Eğrisi üretmek için mertebe k=5 seçilir. Bunun sonucunda C^3 sürekliliğine sahip 6 kontrol noktası ile üretilen Şekil 5.8'de gösterildiği gibi 5 eğri parçası oluşacaktır. Birim parametre $u \in [0,1]$ olarak belirlenmiştir. Kuartik B-Spline Eğrileri dördüncü derece polinomlardan oluştuğundan her bir eğri parçası Denklem 5.39'da verilen denklemler kullanılarak ifade edilir.

$$\begin{aligned} B_0(u) &= a_0 + b_0u + c_0u^2 + d_0u^3 + e_0u^4 \\ B_1(u) &= a_1 + b_1u + c_1u^2 + d_1u^3 + e_1u^4 \\ B_2(u) &= a_2 + b_2u + c_2u^2 + d_2u^3 + e_2u^4 \\ B_3(u) &= a_3 + b_3u + c_3u^2 + d_3u^3 + e_3u^4 \\ B_4(u) &= a_4 + b_4u + c_4u^2 + d_4u^3 + e_4u^4 \end{aligned} \tag{5.39}$$

Kuartik B-Spline Eğrisinin mertebesi k=5 olduğu için eğri, C^0, C^1, C^2 ve C^3 sürekliliklerine sahiptir. Her eğri parçasının bitiş noktası kendisinden sonraki eğrinin başlangıç noktasına eşittir. Bu özellikler kullanılarak Tablo 5.4'te verilen denklemler üretilmiştir.

Denklemlerin katsayıları 25 bilinmeyenden oluşmaktadır. Tabloda verilen 24



Şekil 5.8: Kuartik B-Spline Baz Fonksiyonları

Tablo 5.4: Kuartik B-Spline Baz Denklemleri

C^0 **Özelliği**	C^1 **Özelliği**	C^2 **Özelliği**	C^3 **Özelliği**
$B_0(0) = 0$	$B_0'(0) = 0$	$B_0''(0) = 0$	$B_0'''(0) = 0$
$B_0(1) = B_1(0)$	$B_0'(1) = B_1'(0)$	$B_0''(1) = B_1''(0)$	$B_0'''(1) = B_1'''(0)$
$B_1(1) = B_2(0)$	$B_1'(1) = B_2'(0)$	$B_1''(1) = B_2''(0)$	$B_1'''(1) = B_2'''(0)$
$B_2(1) = B_3(0)$	$B_2'(1) = B_3'(0)$	$B_2''(1) = B_3''(0)$	$B_2'''(1) = B_3'''(0)$
$B_3(1) = B_4(0)$	$B_3'(1) = B_4'(0)$	$B_3''(1) = B_4''(0)$	$B_3'''(1) = B_4'''(0)$
$B_4(1) = 0$	$B_4'(1) = 0$	$B_4''(1) = 0$	$B_4'''(1) = 0$

denkleme ek olarak genlik özelliği kullanarak oluşturulan Denklem 5.40 kullanılır.

$$1 = B_0(0) + B_1(0) + B_2(0) + B_3(0) + B_4(0) \tag{5.40}$$

Elde edilen denklem sistemi MATLAB programı ile çözülmüştür. Elde edilen 25 katsayı Denklem 5.39'da yerine yazıldığında oluşan Kuartik B-Spline Eğrisi formülleri Denklem 5.41'de ifade edilmiştir.

$$
\begin{aligned}
B_0(u) &= \tfrac{1}{24}u^4 \\
B_1(u) &= \tfrac{1}{24}[1 + 4u + 6u^2 + 4u^3 - 4u^4] \\
B_2(u) &= \tfrac{1}{24}[11 + 12u - 6u^2 - 12u^3 + 6u^4] \\
B_3(u) &= \tfrac{1}{24}[11 - 12u - 6u^2 + 12u^3 - 4u^4] \\
B_4(u) &= \tfrac{1}{24}[1 - 4u + 6u^2 - 4u^3 + u^4]
\end{aligned} \tag{5.41}
$$

Elde edilen denklemler incelendiğinde ilk eğrinin son denklem ile üretildiği son eğrinin ilk denklem ile üretildiği görülmüştür. Denklemlerin sırası eğrilerin sırasına uygun olması açısından tersine sıralanmıştır. Son olarak denklemlerin MATLAB ortamında kullanılmasında kolaylık sağlayacağı değerlendirilen matris formuna dönüştürülerek Denklem 5.42'de ifade edilmiştir.

$$
C(u) = \frac{1}{24}\begin{bmatrix} u^4 & u^3 & u^2 & u & 1 \end{bmatrix}
\begin{bmatrix}
1 & -4 & 6 & -4 & 1 \\
-4 & 12 & -12 & 4 & 0 \\
6 & -6 & -6 & 6 & 0 \\
-4 & -12 & 12 & 4 & 0 \\
1 & 11 & 11 & 1 & 0
\end{bmatrix}
\begin{bmatrix} P_{i-1} \\ P_i \\ P_{i+1} \\ P_{i+2} \\ P_{i+3} \end{bmatrix} \tag{5.42}
$$

Denklem 5.38 ve Denklem 5.42'de bulunan matrisler göz önünde bulundurularak bir genelleme yapılacak olursa; mertebesi k (derecesi k-1) olan B-Spline eğrisi matris formunda k parametresine bağlı olarak Denklem 5.43 ile ifade edilir [183].

$$
C_u = U_k M_k P_k(0) \tag{5.43}
$$

Burada U_k, 1×k ve P_k, k×1 boyutlu matrisleri basitçe oluşturulabilir. M_k k×k boyutlu matrisi ise Denklem 5.44'te verilen formül ile hesaplanır [188].

$$
[M_k] = [M_{i+1,j+1}] = \left[\frac{1}{(k-1)!} C_{k-1,i} \sum_{m=j}^{k-1} \left(k - (m+1)^i\right) (-1)^{m-j} C_{k,m-j} \right] \tag{5.44}
$$

Burada; $C_{i,j} = \binom{i}{j}$ binom katsayısıdır.

Bu bölümde elde edilen B-Spline Eğrileri kullanılarak 6'ncı Bölüm'de örnek problemler üzerinde rota planlaması yapılmış ve elde edilen sonuçlar incelenmiştir.

5.3 Dubins Yolu

Düzlemde verilen iki noktayı eğrilik kısıtı ile birbirine bağlayan en kısa yol Dubins Yolu (Dubins Path) olarak ifade edilmektedir. 1957 yılında Lester Eli Dubins tarafından Analitik Geometri ve Diferansiyel Geometri kullanılarak geliştirilen bir yöntemdir [189]. Verilen problem için çözümün sonlu bir eğri kümesi arasından bulunabileceği kanıtlanmıştır. Eğri kümesi Dubins eğrileri olarak adlandırılan altı elemandan oluşur. Dubins eğrileri kümesi aşağıda belirtilmiştir [190]:

DY={RSR, RSL, LSL, LSR, RLR, LRL }

Bu kümede belirtilen eğriler için; S (Straight) düz bir çizgi parçasını temsil eder, L (Left) sola dairesel bir yayı ve R (Right) sağa dairesel bir yayı ifade eder. L ve R yaylarının yarıçapları r_{min} olacak şekildedir.

Bu yöntemin en kısa yolu ifade ettiği daha sonra araştırmacılar tarafından Pontryagin'in maksimum prensibi kullanılarak ispatlanmıştır [191]. Bu yöntem Euler-Lagrange denkleminin özel bir halidir. Yöntemde en kısa yol, maksimum eğim ve düz çizgilerin dairesel yaylara eklenmesi ile elde edilir. Dubins Yolu'nun non-holonomik araçlar olan, tekerlekli robotlar, uçaklar ve sualtı araçlara yönelik yol planlaması için robotik ve kontrol teorisi alanlarında kullanımı yaygındır [192–194].

Minimum dönüş yarıçapı r, yönü θ ve başlangıç noktası $(x_s, y_s) \in R^2$ noktanın düzlemdeki koordinatları olmak üzere, hareket Denklem 5.45 ile belirtilen formüle göre gerçekleşmektedir [190].

$$\begin{cases} \dot{x} = v_0 Cos\theta \\ \dot{y} = v_0 Sin\theta \\ \dot{\theta} = \frac{v_0}{r} u, \qquad u \in [-1, 1] \end{cases} \tag{5.45}$$

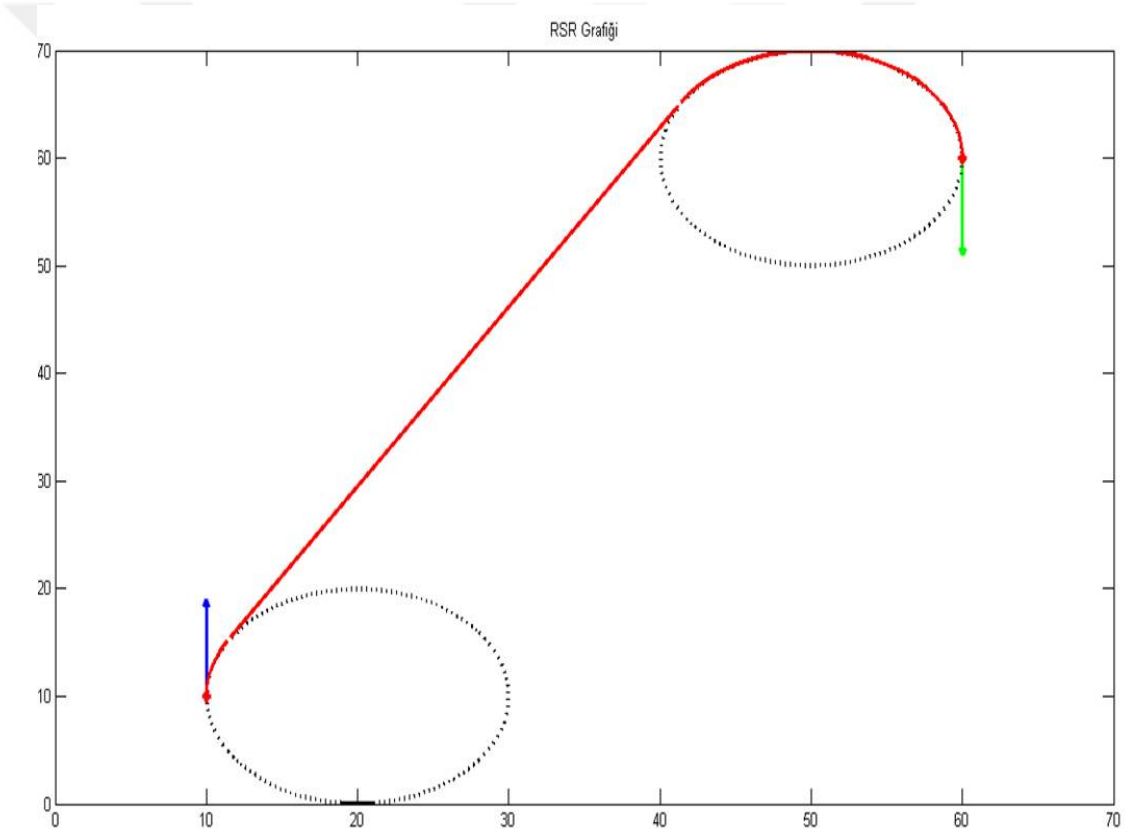
Bu denklemde v_0 sabit hızı ifade etmektedir, u ise normalleştirme parametresidir.

Dubins Yolu'nu oluşturan eğriler bir takım geometrik hesaplamalar ile elde edilmektedir. Dubins Yolu'nun elde edilmesi ile ilgili ayrıntılı hesaplama işlemleri için Andy Giese tarafından oluşturulan Dubins Yollarının adım adım

hesaplanmasına yönelik çalışmadan yararlanılmıştır [195].

5.3.1 CSC Yoları

CSC (Circle-Straight-Circle) yolları iki kontrol noktasının birbirine bir eğri ile bağlanması esnasında bir dairesel yay, bir doğru parçası ve ikinci bir dairesel yay kullanan yollardır. DY kümesi ile belirttiğimiz altı Dubins Yolu'ndan dördü bu duruma uymaktadır. Yani RSR, RSL, LSR ve LSR yolları. Bir RSR yolu örneği Şekil 5.9 ile gösterilmiştir.

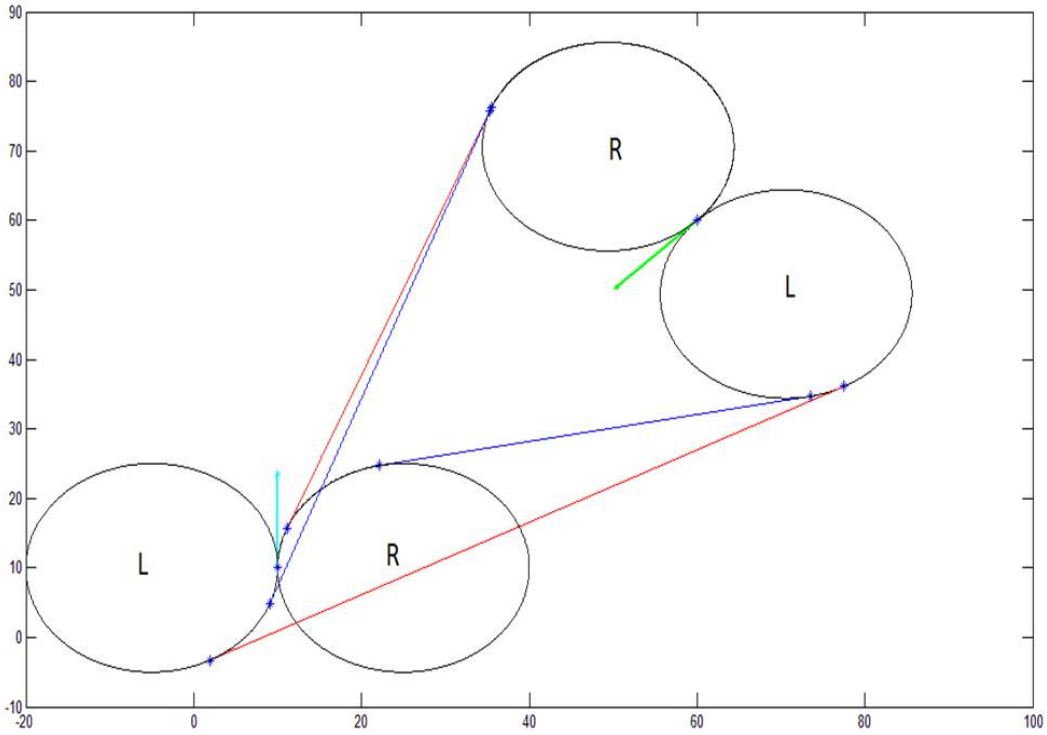


Şekil 5.9: Bir RSR Yolu Örneği

Dubin Yolunun oluşturulması için, ilk olarak başlangıç ve hedef olarak belirlediğimiz kontrol noktalarının konum ve yönleri belirlenmiştir. Başlangıç ve hedef noktalarından İHA'nın hareket etmekte olduğu yönü belirtecek şekilde oklar çizilmiştir. Dubins Yolu'nu oluşturacak olan İHA'nın bulunduğu noktanın

ve hedef noktasının sağına ve soluna bu noktalara teğet olacak şekilde r_{min} yarıçaplı çemberler çizilmiştir. Başlangıç konfigürasyonundaki çemberlerden hedef konfigürasyonundaki çemberlere teğet çizgileri çizilir.

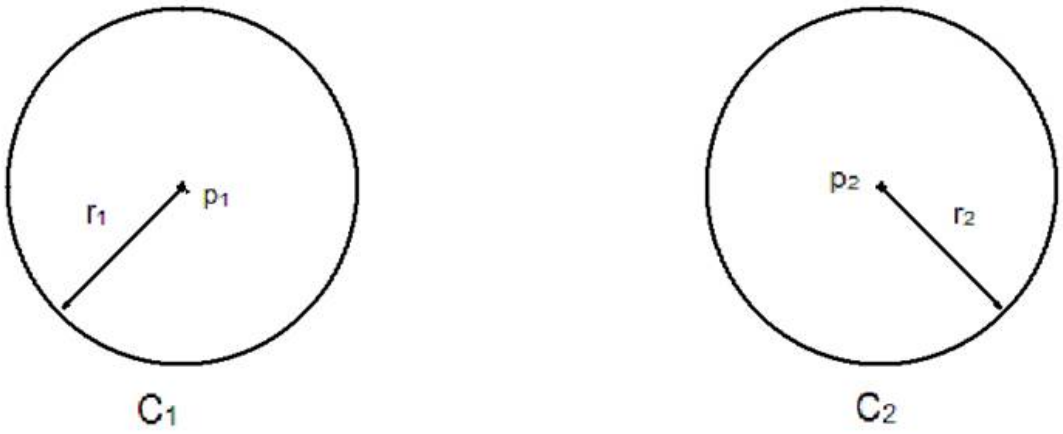
Çember çiftleri olan (RR), (LL), (RL), (LR) için, dört muhtemel teğet çizgisi olmalıdır, dikkat edilmesi gereken her çift için geçerli olan sadece bir çizgi vardır. Bu olası teğetler Şekil 5.10'da gösterilmiştir. RR daireleri için İHA'nın başlangıç çemberinden uzanan yalnızca bir çizgi hedef noktanın çemberine de teğet olur. Burada İHA'nın ileri yönde hareket ettiği ve bu hareket sırasında çemberlerin üzerinde izleyeceği rotalara göre hesaplama yapılmaktadır. İki çemberin yönleri de hesaba katıldığında birbirlerine ikisi dış ve ikisi iç olmak üzere toplam dört teğeti bulunur. Bu nedenle bir CSC yolu için izlenecek tek bir teğet çizgisi vardır. Bu teğet çizgi, yolun S kısmını oluşturur. Çizginin çemberle teğet olduğu noktalar İHA'nın yolunu tamamlamak için geçmek zorunda olduğu noktalardır. Bu nedenle, CSC yollarını hesaplamanın temel noktası bu teğetleri doğru hesaplamaktır.



Şekil 5.10: Çemberler Arasında Oluşabilecek Teğet Doğruları

5.3.2 Teğet Doğrusu Hesaplama

CLC yollarında kullanılan başlangıç ve hedef noktasına komşu çemberlerin birleştirilmesinde kullanılan teğet doğrularının hesaplanmasında kullanılan geometrik işlemler bu bölümde incelenecektir. Öncelikle geometrik yöntemler kullanılarak teğet doğrusu elde etmek için gereken hesaplamalar gerçekleştirilmiştir. Daha sonra ise daha etkili bir yöntem olan vektör tabanlı bir biçimde teğet doğrusu hesaplamalarının nasıl yapılacağı belirtilecektir.
Verilen iki çember C_1 ve C_2 bunların yarıçapları da sırasıyla r_1 ve r_2 olsun. Şekil 5.11'de çemberler gösterilmiştir. C_1'in merkezi $P_1 = (x_1, y_1)$ ve benzer şekilde C_2'nin merkezi $P_2 = (x_2, y_2)$ olarak tanımlanır.



Şekil 5.11: Teğetlerin Hesaplanması İçin Kullanılacak Çemberler.

Yöntem-1: Teğetlerin Geometrik Olarak Hesaplanması

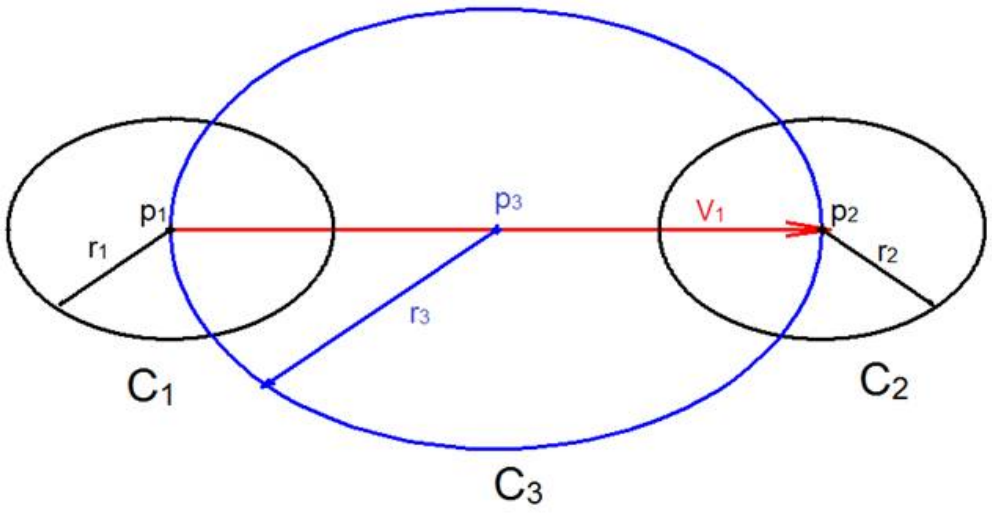
İç Teğetler

İki çemberin içten birbirine teğet olduğu durumda teğet doğrusunun hesaplanması için aşağıdaki adımlar takip edilir:

1.Adım P_1 den P_2 ye $\vec{V_1}$ vektörü çizilir. $\vec{V_1} = (x_2 - x_1, y_2 - y_1)$.
Bu vektörün büyüklüğü D ile ifade edilecek olursa: $D = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ ile hesaplanır.

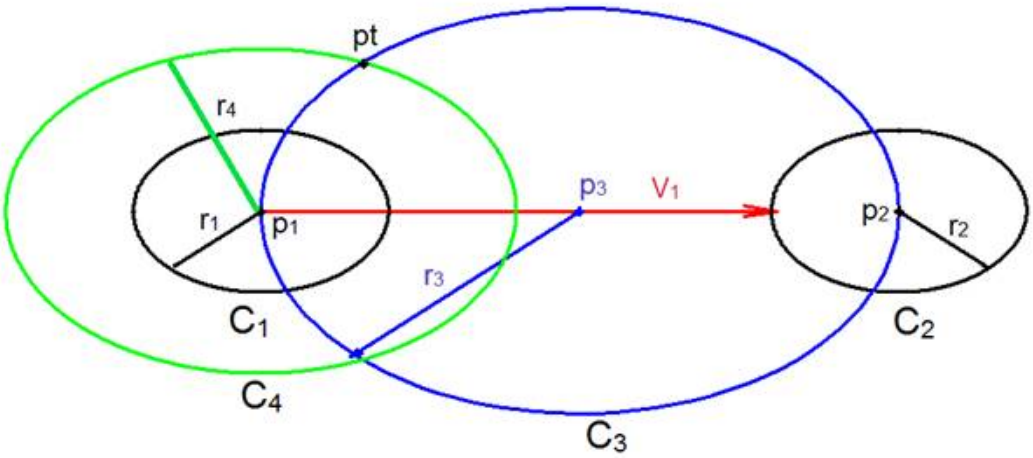
2.Adım $\vec{V_1}$'in orta noktasını merkez olarak kabul eden ve $r_3 = \frac{D}{2}$ yarıçaplı

bir çember Şekil 5.12'de gösterilen şekilde oluşturulur.
Yani $P_3 = (\frac{(x_1+x_2)}{2}, \frac{(y_1+y_2)}{2})$ olur.



Şekil 5.12: İç Teğetlerin Hesaplanması 1. ve 2.Adım.

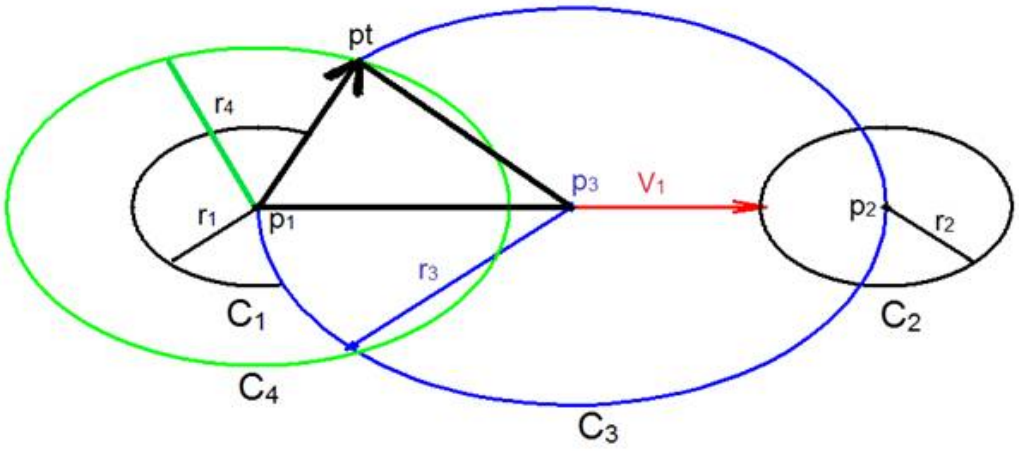
3.Adım C_1 ile aynı merkezli $r_4 = r_1 + r_2$ yarıçaplı bir C_4 çemberi Şekil 5.13'te belirtildiği gibi oluşturulur.



Şekil 5.13: İç Teğetlerin Hesaplanması 3.Adım.

4.Adım P_1'den C_3 ve C_4 arasındaki kesişimde "üstte" olan nokta $P_t = (x_t, y_t)$ olarak belirlenir ve bu noktaya C_1'in merkezinden Şekil 5.14'te gösterildiği gibi bir $\vec{V_2}$ vektörü oluşturulur. Bu vektörü diğer vektörleri kullanarak hesaplayabilirsek birinci teğet noktasını elde edebiliriz, çünkü $\vec{V_2}$ vektörü ile P_t'nin bulunduğu konumun elde edilmesi sağlanır.

5.Adım P_t'nin bulunduğu konum hesaplanırken ilk olarak Şekil 5.14'te gösterilen P_1, P_3 ve P_t noktalarından bir üçgen çizilir. $\overline{P_1P_3}$ ve $\overline{P_3P_t}$'nin büyüklüğü $r_3 = \frac{D}{2}$ dir. $\overline{P_1P_t}$ bölümünün büyüklüğü $r_4 = r_1 + r_2$ dir.
Burada hesaplamak istediğimiz : $\gamma = \widehat{P_tP_1P_3}$ açısıdır.



Şekil 5.14: İç Teğetlerin Hesaplanması 4. ve 5. Adım.

Bu açıyı hesaplamak için Denklem 5.46 ile belirtilen kosinüs teoremi kullanılır.

$$\begin{aligned} a^2 &= b^2 + c^2 - 2bcCosA \\ b^2 &= a^2 + c^2 - 2acCosB \\ c^2 &= b^2 + a^2 - 2baCosC \end{aligned} \tag{5.46}$$

Denklem 5.46'da bulunan değişkenler yerlerine yazıldığında elde edilen eşitlik Denklem 5.47'de gösterilmiştir.

$$(r_3)^2 = (r_3)^2 + (r_4)^2 - 2\,(r_3)\,(r_4)\,Cos\,(\gamma) \tag{5.47}$$

Denklem 5.47'de gerekli sadeleştirme işlemleri yapıldığında; γ açısı r_3 ve r_4

cinsinden Denklem 5.48'de belirtilen şekilde hesaplanır.

$$Cos\left(\gamma\right) = \frac{r_4}{2r_3} \tag{5.48}$$

γ açısı $\vec{V_1}$ vektörünün $\vec{V_2} = (P_t - P_1)$ vektörü ile aynı yörüngeden geçmesi için gereken açıdır. $\vec{V_1}$ vektörünün x-ekseni ile pozitif yönde yaptığı θ açısı, Denklem 5.49 kullanılarak hesaplanır.

$$\theta = \gamma + atan2\left(\vec{V_1}\right) \tag{5.49}$$

Burada kullandığımız atan2 fonksiyonu bilgisayar işlemleri için tanımlanmış özel bir fonksiyondur. Hesaplama yaparken arctan fonksiyonu $\left(\frac{-\pi}{2}, \frac{\pi}{2}\right)$ aralığını kullandığından bu aralığın dışında kalan açıları belirlemek için x ve y değerlerinin bulunduğu bölgeleri de hesaba katarak işlem yapmaktadır. Denklem 5.50'de atan2 fonksiyonunun çalışma prensibi gösterilmiştir.

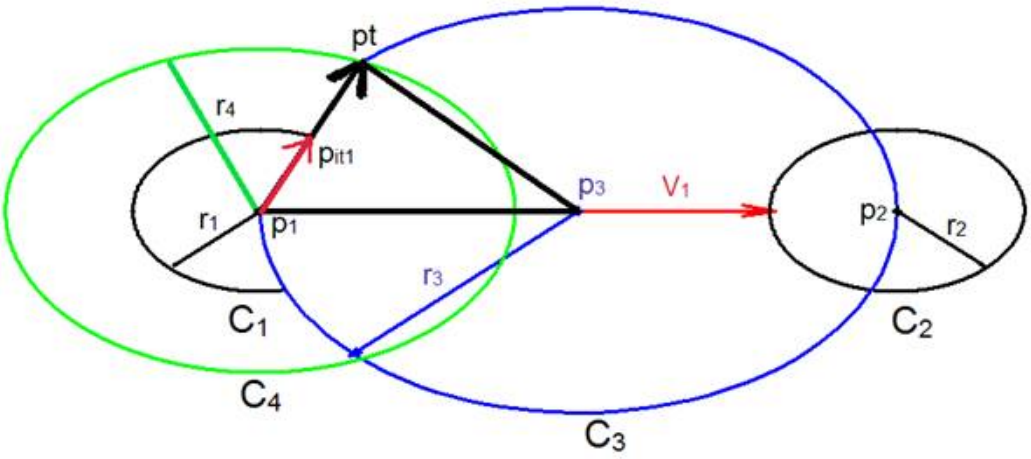
$$atan2\left(x,y\right) = \begin{cases} arctan\left(\frac{y}{x}\right) & \text{eğer } x > 0 \\ arctan\left(\frac{y}{x}\right) + \pi & \text{eğer } x < 0, y \geq 0 \\ arctan\left(\frac{y}{x}\right) - \pi & \text{eğer } x < 0, y < 0 \\ +\frac{\pi}{2} & \text{eğer } x = 0, y > 0 \\ -\frac{\pi}{2} & \text{eğer} x = 0, y < 0 \\ \text{tanımsız} & \text{eğer } x = 0, y = 0 \end{cases} \tag{5.50}$$

P_t noktası P_1 noktasından $\vec{V_2}$ değerini r_4 mesafesi boyunca geçirilerek elde edilir. Denklem 5.51'de gösterilen şekilde P_t noktası elde edilir.

$$\begin{aligned} x_t &= x_1 + (r_1 + r_2)\, Cos\left(\theta\right) \\ y_t &= y_1 + (r_1 + r_2)\, Sin\left(\theta\right) \end{aligned} \tag{5.51}$$

6.Adım C_1 üzerindeki ilk iç teğet noktası P_{it1}'yi bulmak için P_t'yi elde etme şekline benzer bir yöntem izlenir. Şekil 5.15'te gösterildiği gibi $\vec{V_2}$ boyunca P_1'den yola çıkılır. Ancak sadece r_1 mesafesinde ilerlenir. Çünkü P_t bilindiği için $\vec{V_2} = P_t - P_1$ hesaplanabilir. $\vec{V_2}$ normalize edilip (büyüklüğünün bir birim olacak şekilde vektörün normuna bölünmesi işlemi), P_1 den P_{it1} e bir $\vec{V_3}$ vektörü elde etmek

için r_1 ile çarpılması gerekmektedir. Bu durumda $\vec{V_3}$ vektörü Denklem 5.52 ile gösterilen formül ile hesaplanır.



Şekil 5.15: İç Teğetlerin Hesaplanması 6.Adım.

$$\vec{V_3} = \frac{\vec{V_2}}{\left\|\vec{V_2}\right\|} r_1 \tag{5.52}$$

Elde edilen veriler doğrultusunda P_{it1} noktası Denklem 5.53'te belirtilen formül ile hesaplanır.

$$P_{it1} = P_1 + \vec{V_3} \tag{5.53}$$

Noktaya vektörün eklenmesinden kastedilen noktanın ötelenmesi işlemidir, yani noktanın bileşenlerine vektörün bileşenlerinin eklenmesidir. Bunun sonucunda istediğimiz nokta elde edilir.

7.Adım P_t noktası daha önce hesaplandığı için P_t'den P_2'ye $\vec{V_4}$ vektörü Denklem 5.54 ile hesaplanır. Bu durumda Şekil 5.16'da gösterildiği gibi elde edilen V_4 vektörü C_1 ve C_2 arasındaki iç teğet doğrusuna paraleldir.

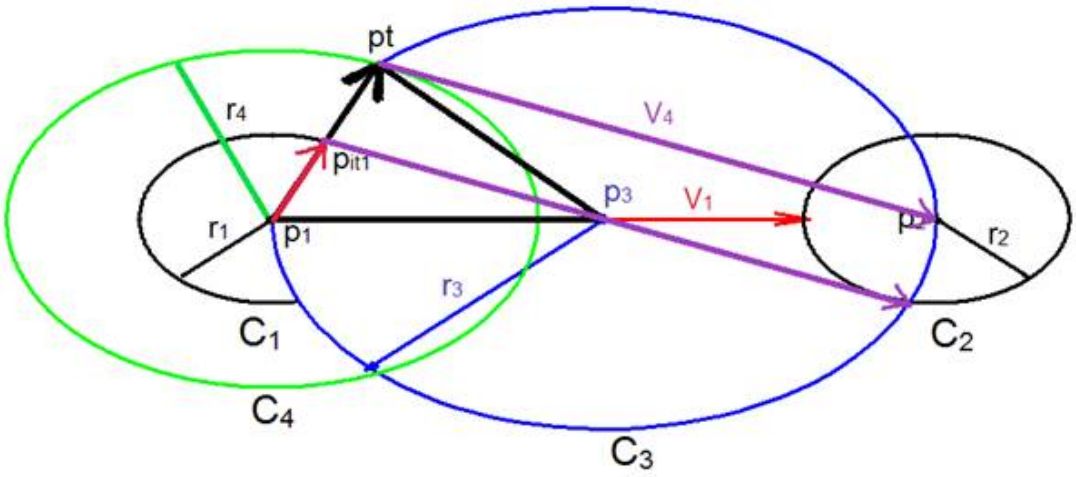
$$\vec{V_4} = P_2 - P_t \tag{5.54}$$

C_2 üzerindeki bir iç teğet noktasını bulmak için $\vec{V_4}$ vektörünün büyüklüğünden ve yönünden yararlanılır. Denklem 5.53 ile hesaplanan P_{it1} noktasına $\vec{V_4}$ vektörü

eklenerek C_2 üzerindeki iç teğet noktası P_{it2} Denklem 5.55'teki formül ile hesaplanır.

$$P_{it2} = P_{it1} + \vec{V_4} \tag{5.55}$$

C_1 ve C_2 arasındaki iç teğet noktaları P_{it1} ve P_{it2}'nin hesaplanması ile benzer



Şekil 5.16: İç Teğetlerin Hesaplanması 7.Adım.

şekilde C_3 ve C_4 arasındaki "alt" kesişim noktası kullanılarak diğer iç teğet noktaları olan P_{it3} ve P_{it4} hesaplanır.

P_{it3} hesaplanırken "alt" tarafta olacağından α açısı Denklem 5.56'da belirtilen formül ile hesaplanır.

$$\alpha = atan2\left(\vec{V_1}\right) - \gamma \tag{5.56}$$

Bu durumda P_{t2} noktası Denklem 5.57'de verilen eşitlik ile elde edilir.

$$\begin{aligned} x_{t2} &= x_1 + (r_1 + r_2)\, Cos\,(\alpha) \\ y_{t2} &= y_1 + (r_1 + r_2)\, Sin\,(\alpha) \end{aligned} \tag{5.57}$$

$\vec{V_5} = P_{t2} - P_1$ vektörü tanımlanır ve normalize edilir. P_1 noktasına r_1 büyüklüğünde normalize edilmiş $\vec{V_5}$ vektörünün eklenmesi ile P_{it3} noktası Denklem 5.58 ile verilen eşitlik ile hesaplanır.

$$P_{it3} = P_1 + \frac{\vec{V_5}}{\left\|\vec{V_5}\right\|} r_1 \tag{5.58}$$

$\vec{V_6}$ vektörü P_{t2}'den P_2'ye tanımlanır ve benzer şekilde C_2 üzerindeki ikinci iç teğet noktası P_{it4} Denklem 5.59'da ifade edilen formül ile hesaplanır.

$$\begin{aligned} \vec{V_6} &= P_2 - P_{t2} \\ P_{it4} &= P_{it2} + \vec{V_6} \end{aligned} \tag{5.59}$$

Dış Teğetler

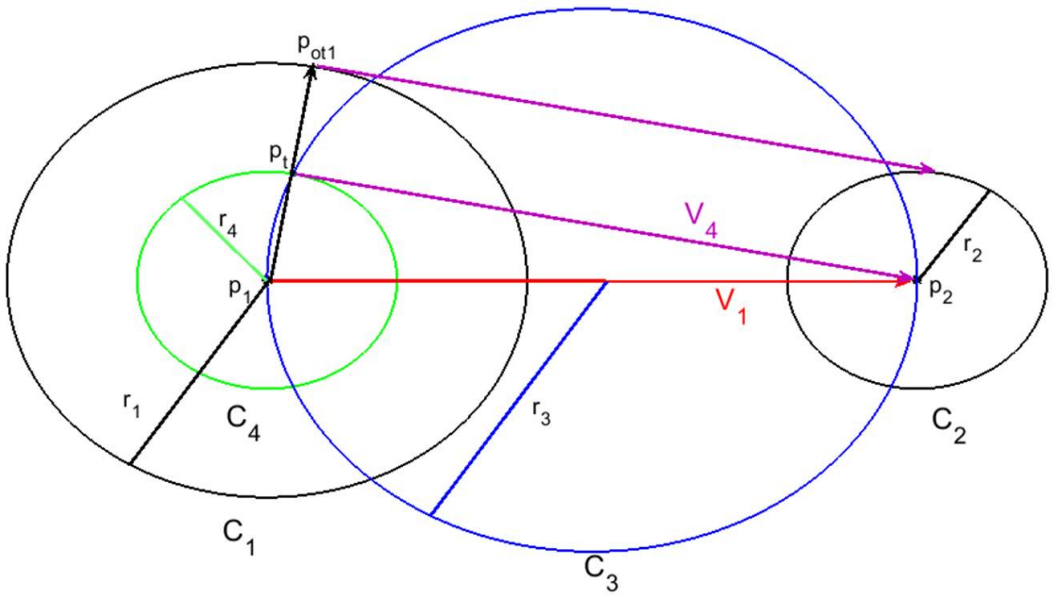
Dış teğet noktalarının oluşturulması da iç teğet noktalarının oluşturulması ile benzer şekildedir. Aynı şekilde iki çember C_1 ve C_2 verilmiş olsun ve $r_1 \geq r_2$ olduğunu varsayalım. P_1 merkezli C_4 çemberi yarıçapı $r_4 = r_1 - r_2$ olacak şekilde çizilir. İç teğet noktası bulmak için gerçekleştirilen adımlar izlenerek $\vec{V_1}$'in orta noktası üzerinde tam olarak aynı C_3 çemberi oluşturulur. P_t'yi elde etmek için C_3 ve C_4 arasındaki kesişim bulunur. P_1'den C_3 ve C_4 arasındaki kesişimde "üstte" olan P_t noktasına bir $\vec{V_2}$ vektörü oluşturulur. $\vec{V_2}$ üzerinde r_1'in büyüklüğü kadar ilerlendiğinde P_{ot1} noktası elde edilir. Normalleştirme öncesinde $\vec{V_2}$'nin büyüklüğü $r_4 > r_1$ değil $r_4 < r_1$ olduğunu hatırlamak gerekir. C_2 üzerindeki bu noktaya karşılık gelen dış teğet noktası P_{ot2} Denklem 5.60 kullanılarak hesaplanır. Dış teğet doğrusu oluşturmak için uygulanan adımlar Şekil 5.17 üzerinde gösterilmiştir.

$$P_{ot2} = P_{ot1} + \vec{V_4} \tag{5.60}$$

İç teğet noktalarının oluşturulması ile dış teğet noktalarının hesaplanması arasındaki tek fark C_4 çemberinin oluşturulma şeklidir. Bunun dışındaki tüm adımlar aynı şekilde uygulanır.

Yöntem-2: Teğetlerin Vektör Tabanlı Olarak Hesaplanması

Bu bölümde anlatılacak yöntemin geçerliliğinden emin olmak için önceki bölümde anlatılan 7 adımın bilinmesi gerekmektedir. Merkezleri sırasıyla $P_1 = (x_1, y_1)$ ve $P_2 = (x_2, y_2)$ olan ve yarıçapları da sırasıyla r_1 ve r_2 olan C_1 ve C_2 çemberleri verilmiş olsun. Bu durumda vektör tabanlı yaklaşımda



Şekil 5.17: Dış Teğetlerin Hesaplanması

uygulanacak adımlar:

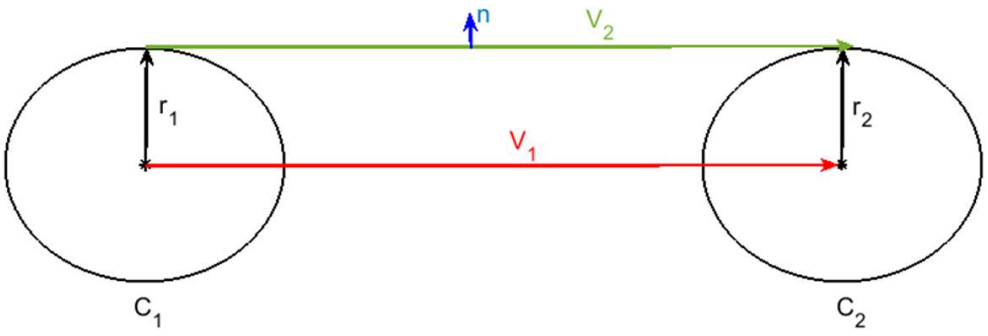
1.Adım P_1 noktasıdan P_2 noktasına $\vec{V_1}$ vektörü çizilir. Bu vektör önceden tanımlandığı gibi D büyüklüğüne sahiptir.

2.Adım Çemberlerin dış teğet noktalarından $\vec{V_2}$ vektörü çizilir. Bu noktalar P_{ot1} ve P_{ot2} noktaları olarak adlandırılır. $\vec{V_2}$ vektörü Denklem 5.61'de belirtilen şekilde tanımlanır.

$$\vec{V_2} = P_{ot2} - P_{ot1} \tag{5.61}$$

3.Adım $\vec{V_2}$ vektörüne dik bir birim vektör çizilir. Bu birim vektör $\hat{n}$ (normal vektör) olarak adlandırılsın. Oluşturulan vektörler Şekil 5.18 ile gösterilmiştir. Burada C_1 ve C_2 çemberlerinin yarıçapları eşit alınmıştır. Ancak yöntem yarıçapların eşit olmadığı durumlarda da geçerlidir.

4.Adım Elde edilen bu üç vektörün birbiri ile ilişkileri incelendiğinde;
$\vec{V_2}$ vektörü ile $\hat{n}$ vektörü birbirlerine dik oldukları için skaler çarpımları sıfırdır. $\hat{n}$ birim vektör olduğu için $\hat{n}.\hat{n} = 1$ dir.



Şekil 5.18: Dış Teğetlerin Vektör ile Hesaplanması

$\vec{V_1}$ vektörü uç uca ekleme yöntemi ile $\vec{V_2}$ vektörüne paralel hale getirilir. Bunun için $\hat{n}$ vektörü kullanılarak gerçekleştirilen, $\vec{V_1} - (r_2 - r_1)\hat{n}$ işlemi sonucu elde edilen vektör $\vec{V_2}$ vektörüne paraleldir.

İşlem sonucunda elde edilen vektör aynı zamanda $\hat{n}$ vektörüne dik olacağından skaler çarpımı sıfır olur. Denklem 5.62'de verilen işlem adımları ile eşitlik en sade halini alır.

$$\begin{aligned} \hat{n}\left(\vec{V_1} - (r_2 - r_1)\hat{n}\right) &= 0 \\ \hat{n}\vec{V_1} - (r_2 - r_1) &= 0 \\ \hat{n}\vec{V_1} &= r_2 - r_1 \end{aligned} \tag{5.62}$$

Elde edilen eşitliğin her iki tarafı $\vec{V_1}$ vektörünün büyüklüğü D'ye bölünerek $\vec{V_1}$ vektörü normalleştirilir. Normalleştirilen $\vec{V_1}$ vektörü $\hat{V_1}$ ile gösterilecektir. Bu durumda elde edilen eşitlik Denklem 5.63 ile ifade edilir.

$$\hat{V_1}\hat{n} = \frac{r_2 - r_1}{D} \tag{5.63}$$

$\vec{A}$ ile $\vec{B}$ iki vektör ve aralarındaki açı θ olsun. Bu durumda skaler çarpımları Denklem 5.64 ile tanımlanır.

$$\vec{A}.\vec{B} = \left\|\vec{A}\right\| . \left\|\vec{B}\right\| .Cos(\theta) \tag{5.64}$$

Denklem 5.63'te $\hat{V_1}$ ve $\hat{n}$ birim vektörler olduğu için büyüklükleri 1 birimdir. Bu durumda $c = \frac{r_2 - r_1}{D}$ olarak tanımlanırsa $Cos(\theta) = c$ olduğu görülür. Trigonometrik fonksiyonların özelliği $Cos(\theta)^2 + Sin(\theta)^2 = 1$ kullanılarak

$Sin(\theta) = \sqrt{1-c^2}$ olarak bulunur.

Koordinat sisteminde verilen bir noktayı döndürmek için kullanılan matris Denklem 5.65'te verilen eşitlikte gösterilmiştir. Dönme işlemi sonrasında, (x, y) noktası θ açısı kadar döndürülerek (x', y') noktasına dönüşür.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} Cos\,(\theta) & -Sin\,(\theta) \\ Sin\,(\theta) & Cos\,(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$
$$x' = xCos\,(\theta) - ySin\,(\theta) \tag{5.65}$$
$$y' = xSin\,(\theta) + yCos\,(\theta)$$

$\hat{V_1}$ ve $\hat{n}$ vektörleri arasındaki açının sinüs ve kosinüs değerleri bilinmektedir. Bu durumda $\hat{V_1}$ vektörü arccos(c) açısı kadar döndürüldüğünde $\hat{n}$ vektörüne eşit olur. $\hat{n} = (n_x, n_y)$ olarak tanımlanırsa Denklem 5.66 ile $\hat{n}$ vektörü hesaplanır.

$$n_x = \hat{V}_{1x}c - \hat{V}_{1y}\sqrt{1-c^2}$$
$$n_y = \hat{V}_{1x}\sqrt{1-c^2} + \hat{V}_{1y}c \tag{5.66}$$

Gerçekleştirilen işlemler sonucunda $\hat{n}$ ve $\vec{V_2}$ vektörleri elde edilmiştir. C_1 çemberinin merkezine $r_1\hat{n}$ eklendiğinde birinci dış teğet noktası P_{ot1}, bu noktaya ise $\vec{V_2}$ vektörü eklendiğinde ikinci dış teğet noktası P_{ot2} bulunur.

Diğer teğet noktaları da vektör tabanlı yöntem kullanılarak benzer şekilde hesaplanır.

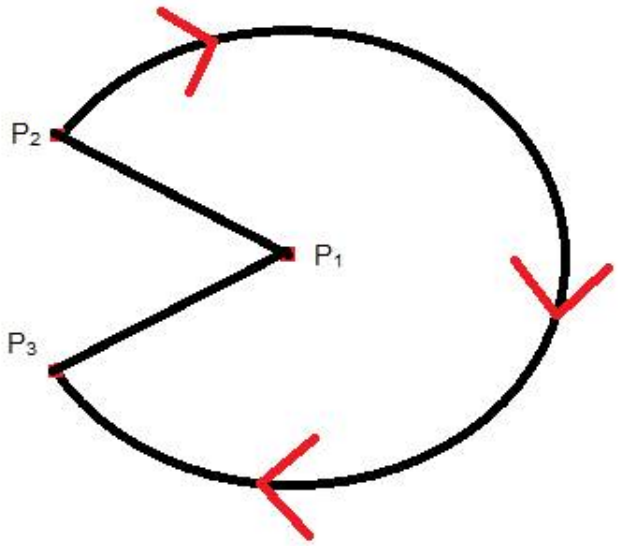
5.3.3 CSC Eğrilerinin Hesaplanması

Çemberlerin teğet noktalarının hesaplanması ile bir CSC yolundaki 'S' kısmı bulunmuş olur. Bu durumda bir CSC yolunun hesaplanması; İHA'nın başlangıç noktasından ilk teğet noktasına kadar minimum yarıçap ile dönmesi ardından ikinci teğet noktasına düz bir yol izlemesi daha sonra ise hedef noktaya kadar minimum yarıçap ile dönmesi ile sağlanır. Dikkat edilmesi gereken husus dönüşün gerçekleştirileceği çemberlerin yönüdür. Çemberin üzerindeki iki noktanın arasında iki farklı yönde iki yay vardır. İHA'nın dönüş işlemi sırasında çemberin yönünü takip etmesi gerekir. Burada sağa dönüş negatif, sola dönüş ise pozitif olarak ifade edilmektedir. Bir İHA sağa dönmesi için oluşturulan çember

üzerinde sola dönemez. Çember üzerindeki yay uzunluklarından hangi uzunluğu tercih edeceğimiz sonucun doğru hesaplanması açısından önem arz etmektedir. Trigonometrik fonksiyonlar burada bize her zaman en kısa yay uzunluğunu verecektir. Ancak bazı durumlarda uzun olan yay tercih edilmektedir. Bu durumda ortaya çıkan sorunun çözümü bir sonraki bölümde ele alınmıştır.

Yay Uzunluklarını Hesaplama

İHA'nın üzerinde dönüş gerçekleştireceği r_1 yarıçaplı P_1 merkezli bir çember ve bu çember üzerinde iki nokta P_2 ve P_3 verilmiş olsun. Çemberin yönünü d ile belirtelim, bu sola veya sağa olabilir. Çemberin yay uzunlukları yarıçap ve çember üzerindeki noktalar arasında kalan açı θ kullanılarak radyan cinsinden hesaplanır. Noktalar arasındaki açı θ kosinüs teoremi kullanılarak hesaplanır. Bu durumda yay uzunluğu $L = r_1\theta$ olarak bulunur. Bu durumda kısa yayın uzunluğu elde edilir. Ancak Şekil 5.19'da gösterildiği gibi kısa yay her zaman geçerli olmayacaktır.



Şekil 5.19: Yönlü Çemberde, Büyük Yay Uzunluğunun Geçerli Olduğu Durum.

Çemberin merkezinden hareketimizi yönlendirecek olan noktalara vektörler, $\vec{V_1} = P_2 - P_1$ ve $\vec{V_2} = P_3 - P_1$ olarak tanımlanır. Geçiş yapılmak istenen yay; P_2

noktasından P_3 noktasına d yönü olarak belirlensin. atan2() fonksiyonu kullanılarak vektörler arasında kalan küçük açı Denlem 5.67 ile gösterilen şekilde hesaplanır. atan2() fonksiyonu ayrıca vektörlerin yönleri hakkında da bilgi vermektedir.

$$\theta = atan2\left(\vec{V_2}\right) - atan2\left(\vec{V_1}\right) \tag{5.67}$$

Burada $\vec{V_1}$ vektörünün $\vec{V_2}$ vektörüne dönmesi, dönüş yönüne yöre, pozitif veya negatif değer alacaktır. Pozitif bir dönüş "sola" ve negatif dönüş ise "sağa" dönüştür. Bu açının işareti d'ye karşı kontrol edİlir. Açının işareti dönülmek istenen yöne uymuyorsa, 2π ekleyerek veya çıkararak düzeltilebilir. Şekil 5.19'da verilen durumda pozitif yönde bir dönüş gerçekleştiği görülmektedir. Çember ise "sağa" yani negatif yönde dönüş gerçekleştirmektedir. Bu durumda negatif bir açı elde etmek için θ açısından 2π çıkartılır. Bu işlem için aşağıda sunulan Algoritma kullanılır.

Algorithm 3 YayUzunluğu Algoritması

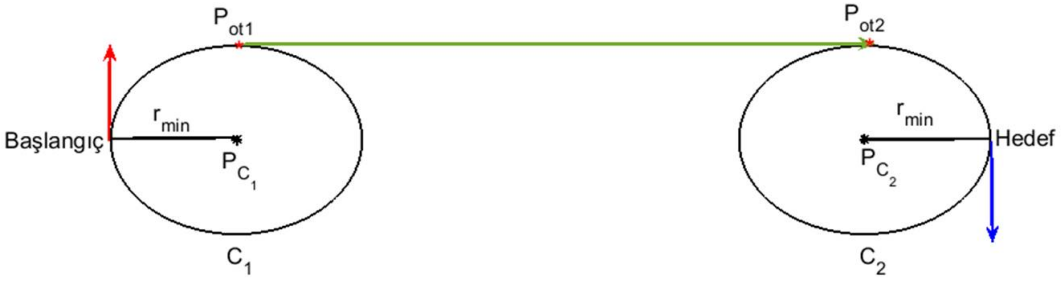
```
1: procedure YAYUZUNLUĞU(V1, V2)
2:     θ = atan2(V2) − atan2(V1)
3:     if θ < 0 and d="sol" then
4:         θ = θ + 2π
5:     else if θ > 0 and d="sağ" then
6:         θ = θ − 2π
7:     end if
8:     return |θ × r1|
```

İHA'nın gerçekleştireceği dönüşler bu algoritma sayesinde hesaplanır.
Elde edilen formüller örnek olarak bir RSR yolunun hesaplanmasında kullanılacaktır.

Bir RSR Yolunun Hesaplanması

Başlangıç ve hedef konumları ve yönleri sırasıyla $b(x_1, y_1, \theta_1)$ ve $h(x_2, y_2, \theta_2)$ olarak verilsin. Önce minimum yarıçap ile sağa dönen ardından düz bir yol

giden ve ardından tekrar minimum yarıçap ile sağa dönen bir yol izlenecektir. Minimum dönüş yarıçapının r_{min} olarak verildiği durumda başlangıç ve hedef noktalarının sağına r_{min} yarıçaplı C_1 ve C_2 çemberleri oluşturulur. Şekil 5.20'de bir RSR yolunun hesaplamasında kullanılan değişkenler sunulmuştur.
C_1 çemberinin merkezi Denklem 5.68 ile verilen formül kullanılarak hesaplanır.



Şekil 5.20: Bir RSR Yolu Hesaplama.

$$P_{C_1}(x_{P1}, y_{P1}) = \left(x_1 + r_{min}Cos(\theta_1 - \frac{\pi}{2}), y_1 + r_{min}Sin(\theta_1 - \frac{\pi}{2})\right) \tag{5.68}$$

Benzer şekilde C_2 çemberinin merkezi Denklem 5.69 ile verilen formül kullanılarak hesaplanır.

$$P_{C_2}(x_{P2}, y_{P2}) = \left(x_2 + r_{min}Cos(\theta_2 - \frac{\pi}{2}), y_2 + r_{min}Sin(\theta_2 - \frac{\pi}{2})\right) \tag{5.69}$$

Çemberlerin merkezlerini birleştiren vektör $\vec{V_1} = P_{C_2} - P_{C_1}$ oluşturulur. Bu vektörün büyüklüğü D olarak hesaplanır. $\hat{V_1} = \frac{\vec{V_1}}{D}$ birim vektörü oluşturulur. $\hat{V_1}$ birim vektörü Denklem 5.70 ile verilen eşitlik kullanılarak 90° döndürülerek $\hat{n}$ birim vektörü elde edilir.

$$\begin{bmatrix} \hat{n}_x \\ \hat{n}_y \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \hat{V}_{1x} \\ \hat{V}_{1y} \end{bmatrix}$$
$$\hat{n}_x = -\hat{V}_{1y} \tag{5.70}$$
$$\hat{n}_y = \hat{V}_{1x}$$

C_1 çemberinin merkezine $r_{min}.\hat{n}$ eklenerek dış teğet noktası P_{ot1} hesaplanır. P_{ot1} noktasına $\vec{V_1}$ vektörü eklenerek diğer dış teğet noktası P_{ot2} hesaplanır. P_{ot1} ve P_{ot2} noktaları Denklem 5.71 ile verilen eşitlikler ile gösterilmiştir.

$$\begin{aligned} P_{ot1} &= P_{C_1} + r_{min}.\hat{n} \\ P_{ot2} &= P_{ot1} + \vec{V_1} \end{aligned} \tag{5.71}$$

Bir RSR Yolunda kullanılan tüm değişkenler buraya kadar olan kısımda hesaplandı. Başlangıç noktasından dönüş hareketine başlayan İHA P_{ot1} noktasına kadar sağa dönüş gerçekleştirecektir. P_{ot1} noktasından P_{ot2} noktasına kadar düz bir yol izleyecektir. P_{ot2} noktasından hedef noktasına kadar sağa dönüş gerçekleştirecektir. Hedef noktasına geldiği andan itibaren İHA planladığı yönde hareketine devam edecektir. Yay Uzunluğu Algoritması kullanılarak dönüş yayları hesaplanır. Bunun için öncelikle çemberlerin merkezlerinden dönüş hareketinin başlangıç ve bitiş noktalarına Denklem 5.72 ile verilen vektörler hesaplanır.

$$\begin{aligned} \vec{V_2} &= b - P_{C_1} \\ \vec{V_3} &= P_{ot1} - P_{C_1} \\ \vec{V_4} &= P_{ot1} - P_{C_3} \\ \vec{V_5} &= P_{ot2} - P_{C_3} \end{aligned} \tag{5.72}$$

Oluşturulan vektörler arasındaki açılar Denklem 5.73'te gösterilen şekilde hesaplanır.

$$\begin{aligned} \theta_1 &= atan2(\vec{V_3}) - atan2(\vec{V_2}) \\ \theta_2 &= atan2(\vec{V_5}) - atan2(\vec{V_4}) \end{aligned} \tag{5.73}$$

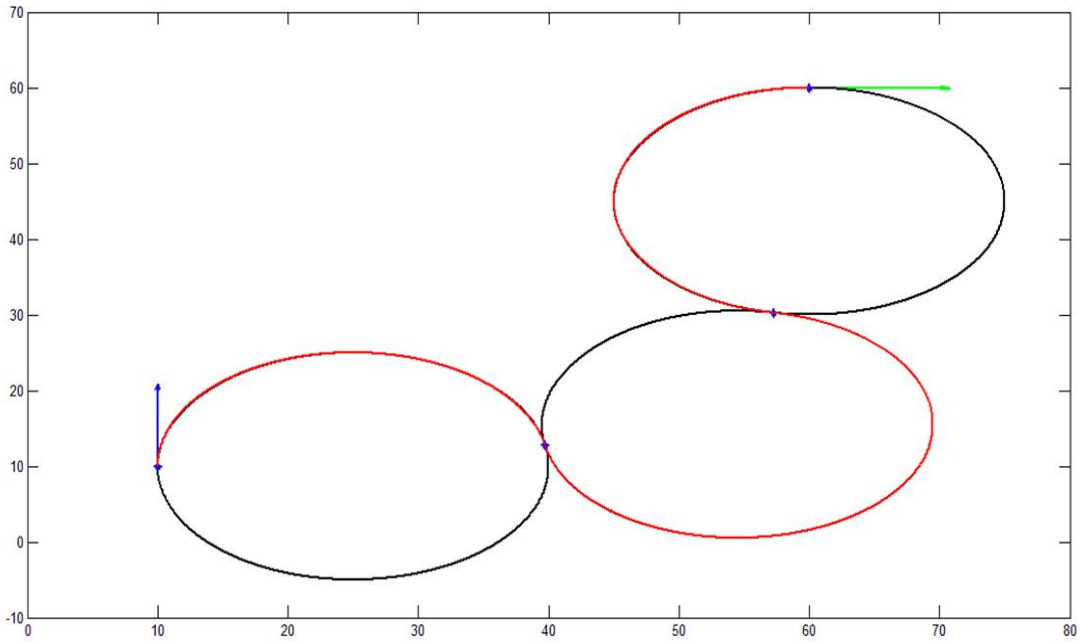
Hesaplanan açıların RSR yoluna uygun olması için yönlerinin kontrol edilmesi ve düzeltme işlemi Denklem 5.74'te verilen eşitlik ile sağlanır.

$$\begin{cases} \text{eğer } \theta_1 > 0 \text{ ise} & \theta_1 = \theta_1 - 2\pi \\ \text{eğer } \theta_2 > 0 \text{ ise} & \theta_2 = \theta_2 - 2\pi \end{cases} \tag{5.74}$$

MATLAB programında elde edilen veriler kullanılarak RSR Yolu oluşturulur. RSL, LSR ve LSL yolları benzer şekilde hesaplanır.

5.3.4 CCC Yolları

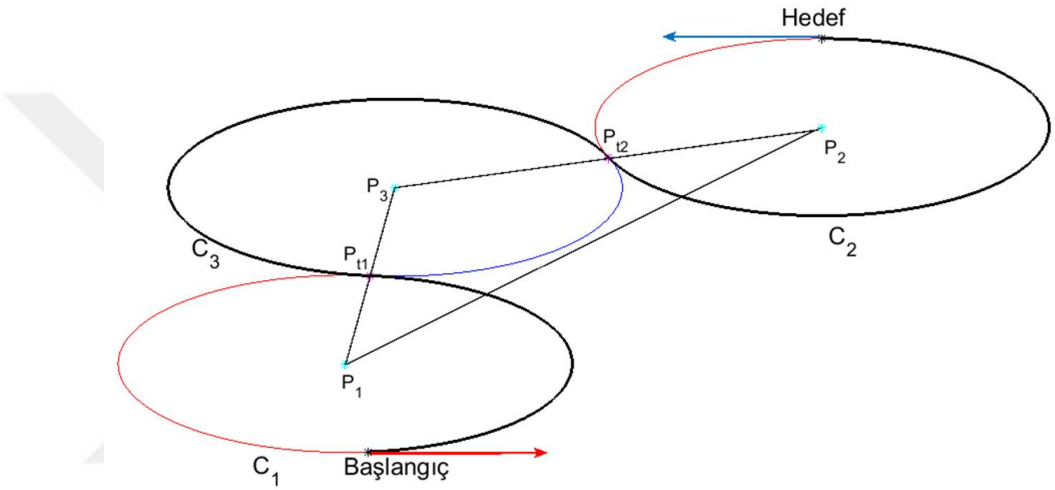
Oluşturulmak istenen rota planının ihtiyaçları doğrultusunda ortaya çıkan CCC (Circle-Circle-Circle) yolları farklı bir yöntem kullanılarak oluşturulurlar. Bu yollar başlangıçta bir yönde dÖnüş, sonrasında zıt yönde bir dönüş ve daha sonrasında ise tekrar başlagıç yönünde bir dönüşten oluşurlar. Şekil 5.21'de bir RLR yolu örneği gösterilmiştir. CCC yolları başlangıç ile hedef noktaların birbirine oldukça yakın olduğu durumlarda gereklidir. Aksi takdirde bu yolun oluşması için bir çemberin yarıçapının r_{min} den daha büyük olması gerekir, bunu yaptığımızda ise CCC yolu problemimiz için uygun bir çözüm olmaktan çıkacaktır. Dubins Yolları kümesinde iki tip CCC yolu vardır: RLR ve LRL. Yolu oluşturmak için ardarda dönmemiz gereken üç çember başlangıç ve hedef noktalarına ve birbirlerine teğettir. Ancak buradaki teğet noktaları, bir önceki bölümde bahsettiğimiz teğet doğruları için kullanılan noktalar ile aynı değildir. Bu nedenle, RLR ve LRL yollarını hesaplamak için bu üç çemberin birbirlerine teğet oldukları noktaların doğru hesaplanması gerekir.



Şekil 5.21: Bir RLR Yolu Örneği.

5.3.5 CCC Eğrilerinin Hesaplanması

Birbirine teğet üç çemberin çizilebilmesi için başlangıç ve hedef nokta arasındaki uzaklığın $4r_{min}$'den küçük olması gerekmektedir [196]. Eğer uzaklık $4r_{min}$'e eşit olursa bu durumda bir CLC eğrisinin kullanılması daha avantajlı olur.
CCC eğrisini hesaplamak için üçüncü çemberin konumu ve ilk iki çember ile teğet olduğu noktaların hesaplanması gerekmektedir. Şekil 5.22 ile sunulan LRL Yolu örnek olarak hesaplanacaktır.



Şekil 5.22: Bir LRL Yolu Hesaplama.

Başlangıç noktasının solunda yer alacak şekilde bir C_1 çemberi ve hedef noktasının solunda yer alacak şekilde bir C_2 çemberi $r = r_{min}$ yarıçapı ile oluşturulur. Bu iki çembere teğet olacak şekilde r yarıçaplı bir C_3 çemberi çizilir. P_1, P_2 ve P_3 sırasıyla C_1, C_2 ve C_3 çemberlerinin merkezi olsun. Amaç P_3 ve teğet noktaları olan P_{t1} ile P_{t2}'yi hesaplamaktır. Çemberlerin merkezleri birleştirilerek $\overset{\triangle}{P_1P_2P_3}$ üçgeni oluşturulur. Çemberler birbirine teğet olduğundan ve ayrıca P_1 ve P_2 noktaları bilindiğinden $\overline{P_1P_3}$ ve $\overline{P_2P_3}$ kenarlarının uzunlukları $2r_{min}$ olur. $\overline{P_1P_2}$ kenarının uzunluğu $D = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ ile hesaplanır. Kosinüs teoremi kullanılarak $\theta = \widehat{P_2P_1P_3}$ açısı Denklem 5.75 ile hesaplanır. C_1 ve C_2 çemberlerinin merkezleri arasında oluşturulan $\vec{V_1} = P_2 - P_1$ vektörü θ açısı kadar döndürülerek C_3 çemberinin merkezi P_3 hesaplanır.

$$\theta = arccos\left(\frac{D}{4r}\right) \tag{5.75}$$

LRL yolu oluşturmak için C_1'in "solunda" RLR için ise "sağında" bir C_3 oluşturulur. Bunun için önce, $\vec{V_1}$ vektörünün x ekseni ile arasındaki açı $atan2\left(\vec{V_1}\right)$ fonksiyonu kullanılarak hesaplanır. Denklem 5.76 ile RLR ve LRL eğrileri oluşturmak için kullanılacak açılar gösterilmiştir.

$$\begin{cases} \text{LRL için} & \theta = atan2\left(\vec{V_1}\right) + \theta \\ \text{RLR için} & \theta = atan2\left(\vec{V_1}\right) - \theta \end{cases} \tag{5.76}$$

Yapılan düzenleme ile θ açısı x ekseni ile pozitif yönde yapılan açıyı belirtmektedir. Bu durumda C_3 çemberinin merkezi P_3 Denklem 5.77 ile hesaplanır.

$$P_3 = (x_1 + 2rCos(\theta), y_1 + 2rSin(\theta)) \tag{5.77}$$

Teğet noktaları P_{t1} ve P_{t2}, P_3 noktasından P_1 ve P_2 noktalarına tanımlanan vektörler yardımıyla hesaplanır. $\vec{V_2} = P_1 - P_3$ ve $\vec{V_3} = P_2 - P_3$ olsun. Bu durumda $\vec{V_2}$ ve $\vec{V_3}$ vektörlerine Denklem 5.78'de belirtilen dönüşümler uygulanır.

$$\vec{V_2} = \frac{\vec{V_2}}{\left\|\vec{V_2}\right\|}r \quad \text{ve} \quad \vec{V_3} = \frac{\vec{V_3}}{\left\|\vec{V_3}\right\|}r \tag{5.78}$$

Bunun sonucunda $P_{t1} = P_3 + \vec{V_2}$ olarak hesaplanır.
Benzer işlemler sonrasında $P_{t2} = P_3 + \vec{V_3}$ olarak bulunur.

LRL Yolunun Hesaplanması

Başlangıç ve hedef konumları ve yönleri sırasıyla $b(x_1, y_1, \theta_1)$ ve $h(x_2, y_2, \theta_2)$ olarak verilsin. Önce minimum yarıçap ile sola dönen ardından minimum yarıçap ile sağa dönen ve ardından tekrar minimum yarıçap ile sola dönen bir yol izlenecektir.
Minimum dönüş yarıçapının r_{min} olarak verildiği durumda başlangıç ve hedef noktalarının soluna r_{min} yarıçaplı C_1 ve C_2 çemberleri oluşturulur. Ardından

aşağıdaki adımlar uygulanır:

1- C_1 çemberinin merkezi Denklem 5.79 ile verilen formül kullanılarak hesaplanır.

$$P_{C_1}(x_{P1}, y_{P1}) = \Big(x_1 + r_{min}Cos(\theta_1 + \frac{\pi}{2}), y_1 + r_{min}Sin(\theta_1 + \frac{\pi}{2})\Big) \tag{5.79}$$

2- Benzer şekilde C_2 çemberinin merkezi Denklem 5.80 ile verilen formül kullanılarak hesaplanır.

$$P_{C_2}(x_{P2}, y_{P2}) = \Big(x_2 + r_{min}Cos(\theta_2 + \frac{\pi}{2}), y_2 + r_{min}Sin(\theta_2 + \frac{\pi}{2})\Big) \tag{5.80}$$

3- Çemberlerin merkezleri arasındaki uzaklık D olmak üzere $\theta = arccos(\frac{D}{4r_{min}})$ olarak hesaplanır.

4- $\vec{V_1} = P_{C_2} - P_{C_1}$ olarak hesaplanır.

5- $\theta = atan2(\vec{V_1}) + \theta$ LRL için düzenlenir.

6- C_3 çemberinin merkezi Denklem 5.81 ile hesaplanır.

$$P_{C_3}(x_{P3}, y_{P3}) = (x_{P1} + 2r_{min}Cos(\theta), y_{P1} + 2r_{min}Sin(\theta)) \tag{5.81}$$

7- $\vec{V_2} = P_{C_1} - P_{C_3}$ ve $\vec{V_3} = P_{C_2} - P_{C_3}$ vektörleri hesaplanır.

8- $\vec{V_2} = \frac{\vec{V_2}}{\|\vec{V_2}\|}r$ ve $\vec{V_3} = \frac{\vec{V_3}}{\|\vec{V_3}\|}r$ dönüşümleri gerçekleştirilir.

9- Çemberlerin teğet noktaları $P_{t1} = P_{C_3} + \vec{V_2}$ ve $P_{t2} = P_{C_3} + \vec{V_3}$ formülleri ile bulunur.

Buraya kadar bir LRL Yolu için gerekli olan tüm eğri parçalarının başlangıç ve bitiş noktaları hesaplandı. Başlangıçta b konumunda olan İHA buradan sola doğru dairesel bir yay ile P_{t1} noktasına kadar gidecektir. Bu noktadan sonra sağa doğru dairesel bir yay ile P_{t2} noktasına kadar gidecektir. Son olarak ise sola doğru dairesel bir yay ile hedef noktası h'ye kadar gidecektir. İHA hedef noktasından itibaren, hedef noktasına geldiğinde bulunması gereken yön ile birlikte yoluna devam edecektir. Bu çember yaylarının oluşturulması sırasında iki nokta arasında bulunan yaylardan hangisinin tercih edileceği Yay Uzunluğu Algoritması ile belirlenir.

10- Dönüş yaylarını hesaplamak için;

$\vec{V_4} = b - P_{C_1}$ ve $\vec{V_5} = P_{t1} - P_{C_1}$

$\vec{V_6} = P_{t1} - P_{C_3}$ ve $\vec{V_7} = P_{t2} - P_{C_3}$

$\vec{V_8} = P_{t2} - P_{C_2}$ ve $\vec{V_9} = h - P_{C_2}$ vektörleri hesaplanır.

11- Vektörler arasındaki açılar:

$\theta_1 = atan2(\vec{V_5}) - atan2(\vec{V_4})$

$\theta_2 = atan2(\vec{V_7}) - atan2(\vec{V_6})$

$\theta_3 = atan2(\vec{V_9}) - atan2(\vec{V_1})$ bulunur.

12- Son olarak bulunan açılar için LRL yoluna uygun olarak Denklem 5.82 ile verilen düzeltme işlemi yapılır.

$$\begin{cases} \text{eğer } \theta_1 < 0 \text{ ise} & \theta_1 = \theta_1 + 2\pi \\ \text{eğer } \theta_2 > 0 \text{ ise} & \theta_2 = \theta_2 - 2\pi \\ \text{eğer } \theta_3 < 0 \text{ ise} & \theta_3 = \theta_3 + 2\pi \end{cases} \tag{5.82}$$

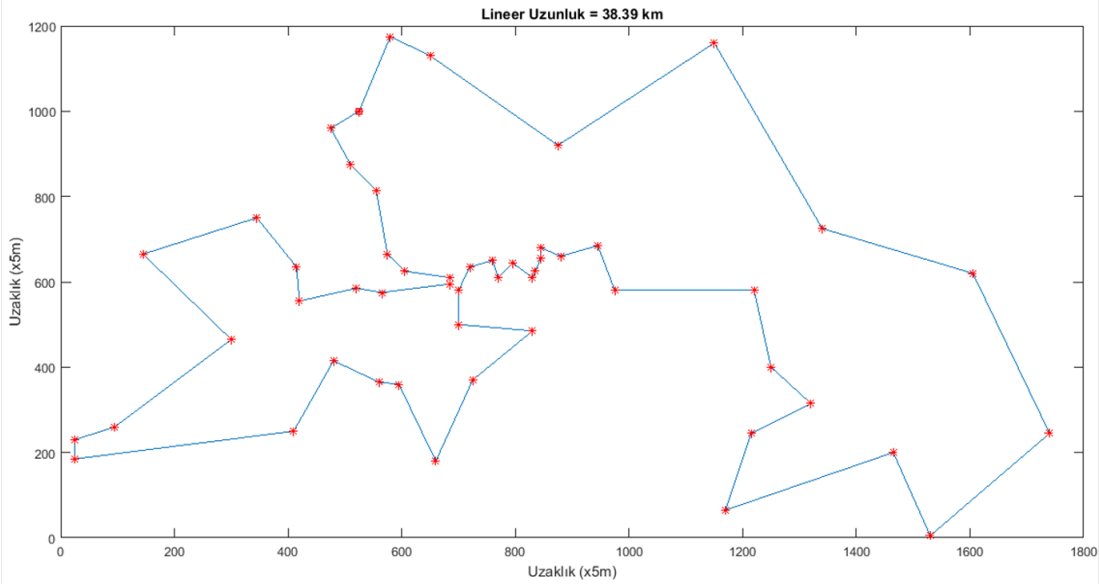
Elde edilen veriler kullanılarak MATLAB programında iki nokta arasında bir LRL Yolu oluşturulur. RLR Yolu'da benzer adımlar izlenerek hesaplanır. Bu bölümde incelenen Dubins Yolları kullanılarak 6'ncı Bölüm'de örnek problemler üzerinde rota planlaması yapılmış ve elde edilen sonuçlar incelenmiştir.

BÖLÜM 6

DENEYSEL SONUÇLAR

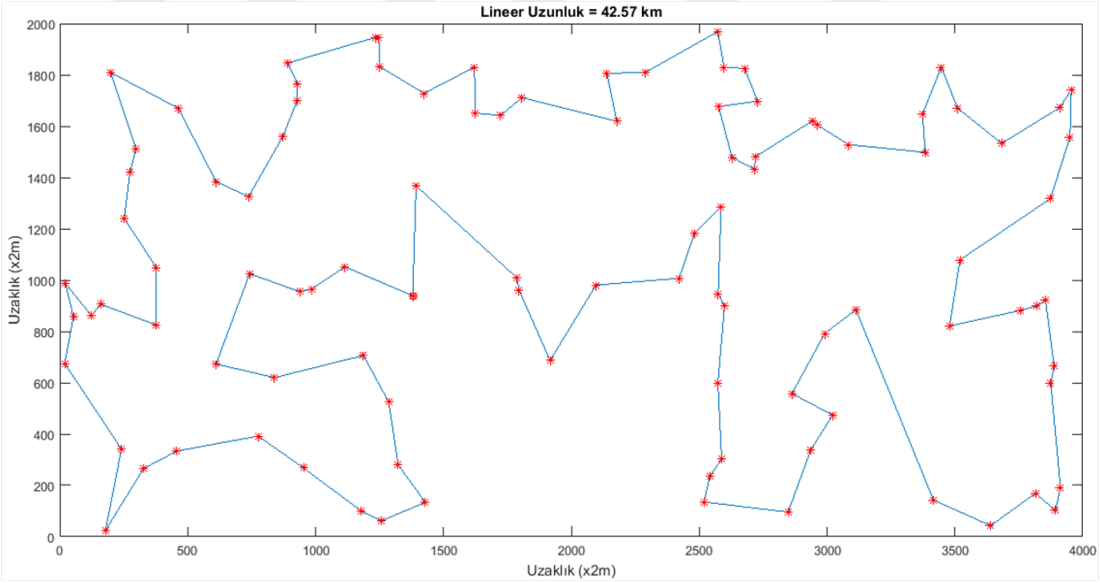
Bu bölümde, 4'üncü Bölüm'de önerilen sistem örnek problemlere uygulanarak elde edilen sonuçlar incelenmiştir. Örnek TSP problemleri ilk olarak GA kullanılarak çözülmüştür. Başlangıç popülasyonu rastgele üretildiği için kodun her çalışması sonucunda birbirlerine yakın ancak farklı sonuçlar alınmaktadır. Problemin ikinci kısmında uygulanan, rota yumuşatma işlemlerinde sonuçlarının karşılaştırılması için çözümlerden biri seçilmiştir. Seçilen çözüm programın on kez çalıştırılması sonunda bulunan en iyi çözümdür. Örnek problemler birim uzaklık üzerinden hesaplanmaktadır. Burada Küçük Sınıf İHA göz önünde bulundurulacağı için bulunan toplam uzaklığın İHA'nın menzili, maksimum 50 km, içinde kalması sağlanmıştır. Berlin52 ve KroA100 problemleri örnek olarak incelenmiştir. Berlin52 probleminin çözümü Şekil 6.1'de sunulmuştur. Berlin52 ye ait birim uzunluk 7677,68 br'dir. Birim uzunluğun 5 m seçilmesi durumunda 38388,41 m olarak hesaplanır. Bulunan sonuçların karşılaştırılması sırasında 10 m'den küçük farkların anlamlı bir sonuç olmayacağı değerlendirildiğinden, uzunluk birimi olarak km kullanılacaktır. Bu durumda Berlin52 probleminin doğrusal olarak uzunluğu 38,39 km'dir. İncelenen örneklerin rota hesaplamaları sırasında birim uzunluk farklı ölçü birimleri ile değiştirilebilir. Bu sayede daha büyük yada daha küçük menzile sahip İHA'lar için rota planlanması durumunda birim uzunluk değiştirilerek ölçeğin büyütülmesi veya küçültülmesi sağlanır.

KroA100 örnek probleminin çözümü Şekil 6.2 ile gösterilmiştir. KroA100'e ait birim uzunluk 21285,44 br'dir. Birim uzunluğunun 2 m seçilmesi durumunda,



Şekil 6.1: Berlin52 TSP Çözümü Doğrusal Grafiği.

toplam mesafe doğrusal olarak 42570,89 m olur. Bu mesafeyi 42,57 km olarak ifade ettiğimizde 50 km'lik menzil mesafesi içinde kaldığı görülmektedir.



Şekil 6.2: KroA100 TSP Çözümü Doğrusal Grafiği.

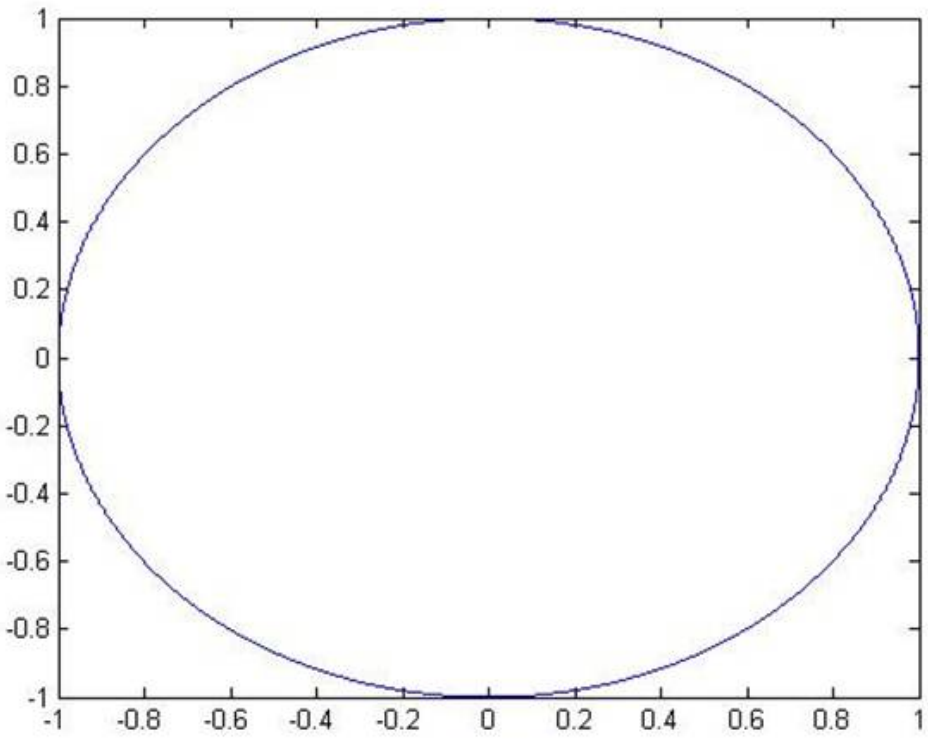
Seçilen TSP örneklerini yumuşatmak için Bezier Eğrileri, B-Spline Eğrileri ve Dubins Yolu yöntemleri örnek problemlere uygulanmıştır. Uygulanan yöntemlerle elde edilen eğrilerin uzunlukları sayısal integral kullanılarak

hesaplanmıştır. Karşılaştırma yapılırken kolaylık sağlaması açısından alınan sonuçlara ait veriler tablolar ile ifade edilmiştir.

6.1 Eğrilerin Uzunluklarının Hesaplanması

TSP ile elde edilen çözümün yumuşatılması sonucunda elde edilen eğrilerin uzunluğunun hesaplanması sırasında kullanılan yöntem, oluşabilecek hatalar göz önünde bulundurulduğunda iyi incelenmesi gereken bir husustur. Yumuşatma işlemi için kullanılan eğriler analitik fonksiyonlar kullanılarak elde edilmediği için belirli integral ile eğri uzunluğu hesaplama yöntemi burada uygulanamaz. Bu durumda eğrilerin uzunluğunun hesaplaması için sayısal türev ve sayısal integral yöntemlerinden yararlanılır. Sayısal integralin hesaplanması aşamasında Trapez (Yamuk) yöntemi tercih edilmiştir. Sayısal hesaplamaların eğri uzunluğu hesaplarken kullanılması durumunda analitik değere yakın sonuçlar elde edilmekle birlikte, bilgisayar hesaplamalarında karşılaşılabilecek yuvarlama ve kesme hataları olabileceği göz önünde bulundurulmuştur. Mutlak Hata ve Bağıl Hata hakkında fikir vermesi açısından eğri uzunluğunu hem sayısal integral kullanarak hesaplayacağımız hem de analitik olarak hesaplayacağımız bir eğri denklemi üzerinde test edilmiştir. Trapez Kuralı uygulanması esnasında oluşacak hata oranını belirlemek için eğri uzunluğunu analitik olarak da kolayca hesaplayacağımız Şekil 6.3 ile verilen birim çemberin çevresi test örneği olarak seçilmiştir.
Merkezi (0,0), yarıçapı r=1 olan çemberin çevre uzunluğu analitik olarak: $2\pi r$ dir

Trapez kuralı ile sayısal integral hesaplaması sırasında $[0, 2\pi]$ aralığı N değerlerine bölünerek sonuç elde edilmiştir. N değerleri arttıkça adımlar küçüldüğü için analitik sonuç ile aradaki farkın azaldığı görülmüştür. N değerleri sırasıyla; 50, 100, 250, 500, 1000, 2500, 5000 ve 10000 seçilerek, Mutlak Hata değerleri hesaplanmış ve hesaplanan Mutlak Hata gerçek değere bölünerek Bağıl Hata oranları bulunmuştur. Ayrıca Bağıl Hata değeri 100 ile çarpılarak



Şekil 6.3: (0,0) Merkezli, r=1 Yarıçaplı Çember Grafiği.

Yüzde Bağıl Hata değerleri hesaplanmıştır. Sonuçlar Tablo 6.1 ile sunulmuştur. N=10000 seçildiğinde adım $2\pi/10000$ olmaktadır bu durumda Bağıl Hata oranı (10^{-4}oranına) onbinde bir oranına indirilmiştir. Bu değerin hesaplamalar sırasında sonucu etkilemeyecek kadar küçük olduğu değerlendirilmiştir. Elde edilen veriler doğrultusunda uzunluk hesaplamaları sırasında N=10000 seçilmiştir.

Denklem 6.1'de çemberin parametrik denklemi verilmiştir.

$$\text{Ç}\,(\theta) = \begin{cases} r\cos\theta \\ r\sin\theta \quad r = 1, \quad 0 \leq \theta \leq 2\pi \end{cases} \tag{6.1}$$

Eğri uzunluğunun formülü Denklem 6.2'de belirtilmiştir.

$$l = \int_0^{2\pi} \sqrt{\left(\frac{dy}{d\theta}\right)^2 + \left(\frac{dx}{d\theta}\right)^2}\, d\theta \tag{6.2}$$

Tablo 6.1: Trapez Kuralının Hata Oranının İncelenmesi

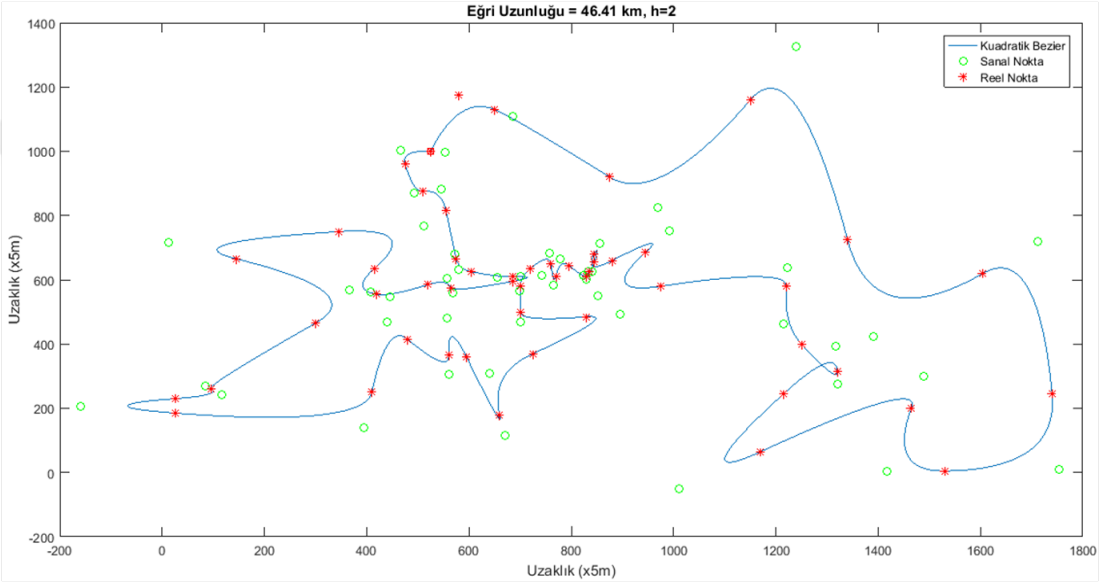
N Değerleri	**50**	**100**	**250**	**500**	**1000**	**2500**	**5000**	**10000**
Sayısal İntegral	6,1507	6,2187	6,2578	6,2706	6,2769	6,2807	6,2819	6,2826
Analitik Sonuç	6,2832	6,2832	6,2832	6,2832	6,2832	6,2832	6,2832	6,2832
Mutlak Hata	0,1324	0,0645	0,0254	0,0126	0,0063	0,0025	0,0013	0,0062848
Bağıl Hata	0,0211	0,0103	0,0040	0,0020	0,0010	0,00040042	0,00020011	0,00010003
Yüzde Bağıl Hata	%2,11	%1,03	%0,4	%0,20	%0,10	%0,04	%0,02	%0,01

6.2 Bezier Eğrileri

Bezier Eğrilerinin oluşturulması sırasında kontrol noktası sayısından etkilenmeyecek şekilde eğri parçalarının birbirine eklenmesi yöntemi tercih edilmiştir. Birbirine eklenen parçalardan oluşan Bezier Eğrileri birleşim esnasında C^1 sürekliliğinin sağlaması için araya eklenen sanal noktalar ile birlikte kullanılmıştır. Sanal noktaların elde edilmesi sırasında kullanılan yöntemler ve eğrilerin uzunluğuna etkileri örnek problemler üzerinde incelenmiştir. Araya sanal nokta eklenmesi sonucunda kontrol noktalarının sırası değişeceğinden, bir sonraki eğriyi oluşturacak kontrol noktalarının her seferinde yeniden tanımlanması gerekir. Sanal kontrol noktalarının geleceği sıra belirlenmiş ve gerçek kontrol noktaları ile birlikte tek bir veri dizisi haline getirilmiştir. Oluşturulan veri dizisinin boyutu hesaplanmıştır. Uygulama sırasında n'inci derece Bezier Eğrisinin n+1 kontrol noktası kullanılarak oluşturulduğundan 5'inci Bölüm'de bahsedilmişti. Bu durumda son eğri parçası için n+1 kontrol noktası kaldığından emin olmak gerekir. Birleştirilmiş veri dizisinin boyutu m olmak üzere, $\frac{m-1}{n}$ değerinin tam sayı olması bunu sağlayacaktır. Başlangıç noktası ihtiyaç duyulan sayıda eklenerek, eğri uzunlunu etkilemeyecek bir şekilde, kapalı eğri oluşması sağlanmıştır. Bu işlemin eğrinin uzunluğuna herhangi etkisi bulunmamaktadır.

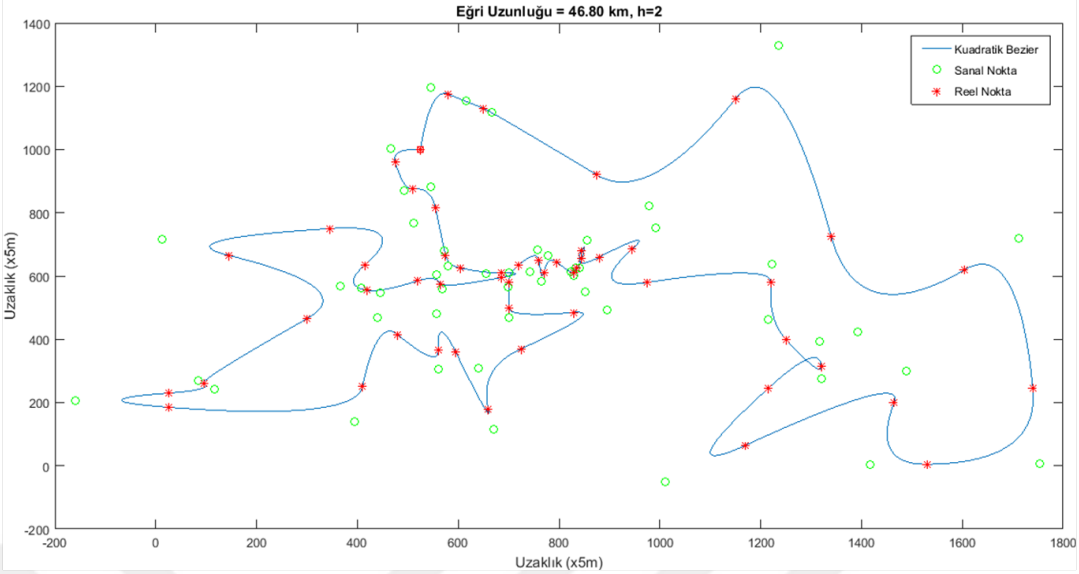
6.2.1 Kuadratik Bezier Eğrileri

Kuadratik Bezier Eğrileri C^1 sürekliliğini sağlayacak şekilde sanal noktalar eklenerek Berlin52 problemine uygulanmıştır. Oluşturulan İHA rotası Şekil 6.4'te sunulmuştur. Eğrinin uzunluğu h=2 seçildiği durumda 46,41 km olarak hesaplanmıştır. Kuadratik Bezier Eğrileri kullanılarak üretilen rota 2'nci kontrol noktası dışındaki tüm kontrol noktalarından geçmektedir. Bu noktanın eğriye olan uzaklığı 0,23 km olarak hesaplanmıştır.



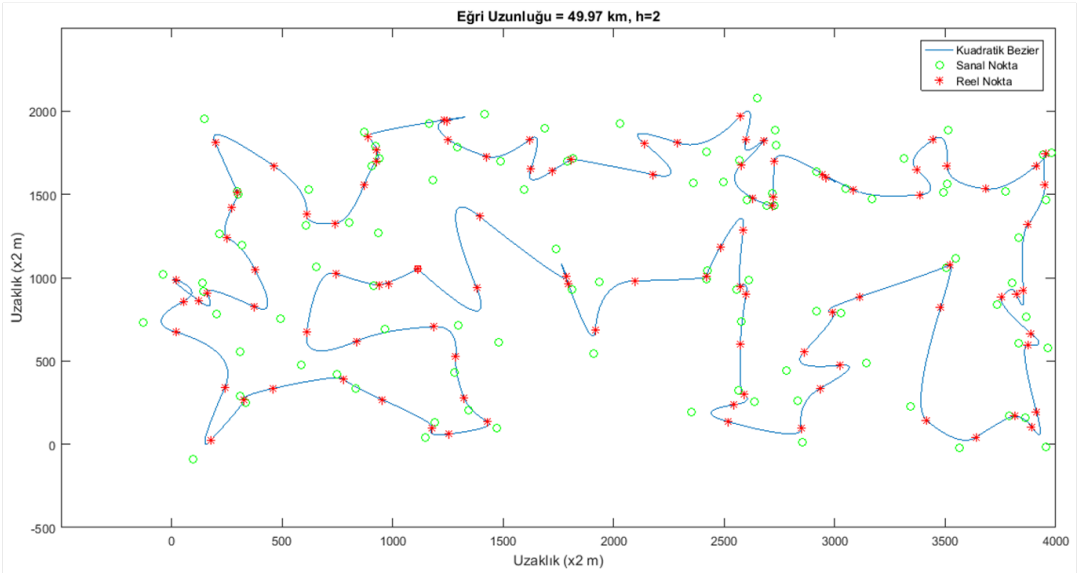
Şekil 6.4: Kuadratik Bezier Eğrisi ile Berlin52 Problemi Çözümü (h=2).

Kontrol noktası sayısından bağımsız olarak Kuadratik Bezier Eğrileri ikinci kontrol noktası hariç tüm kontrol noktalarından geçmektedir. Bu durum bir düzeltme işlemi ile giderilmiş ve C^1 sürekliliği ile tüm kontrol noktalarının üzerinden geçmesi sağlanmıştır. Bu işlem için daha önce kullandığımız sanal nokta ekleme yöntemlerinden farklı olarak iki sanal nokta eklenmiştir. Sanal noktalardan ilki 2'nci ve 3'üncü kontrol noktaları doğrusal olacak şekilde 2'nci kontrol noktası ile 1'inci Kontrol noktası arasına eklenmiştir. İkinci sanal nokta ise, eğrinin uzunluğuna etkisi olmayacak şekilde, 2'nci ve 3'üncü Kontrol noktalarının orta noktası olarak belirlenmiştir. Düzeltme sonrasında oluşturulan eğrinin üzerinden geçmediği kontrol noktası kalmamıştır. Oluşturulan rota Şekil 6.5 ile sunulmuştur. Oluşturulan rota h=2 için 46,80 km uzunluğuna sahiptir.



Şekil 6.5: Düzenlenmiş Kuadratik Bezier Eğrisi ile Berlin52 Çözümü (h=2).

Tüm kontrol noktaları üzerinden geçecek şekilde düzenlenen Kuadratik Bezier Eğrilerinin KroA100 problemine uygulanması sonucu oluşturulan rota Şekil 6.6 ile sunulmuştur. Eğrinin uzunluğu h=2 için 49,97 km dir.



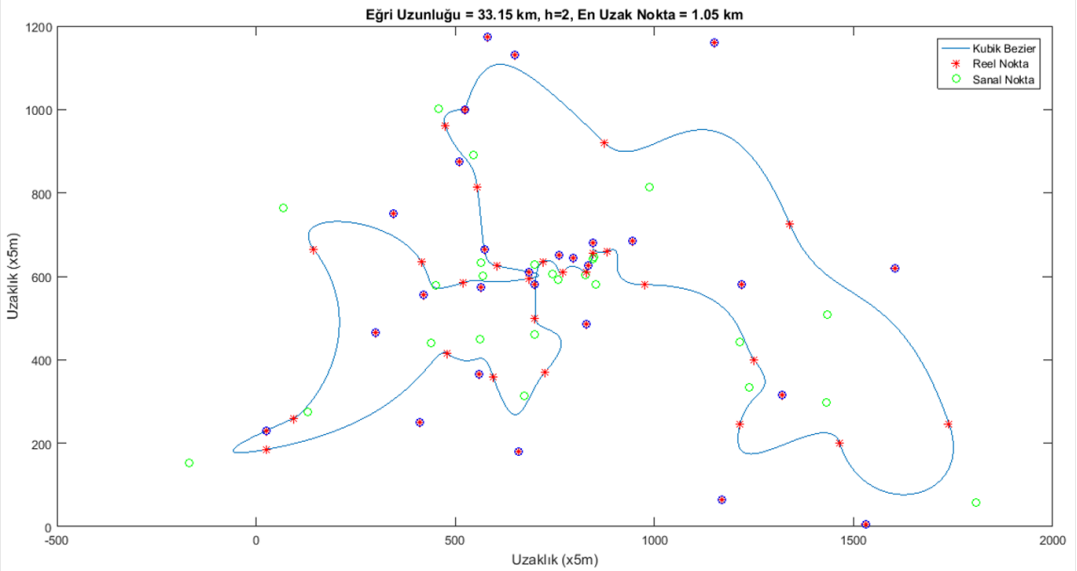
Şekil 6.6: Kuadratik Bezier Eğrisi ile KroA100 Probleminin Çözümü (h=2).

Kuadratik Bezier Eğrilerine eklenen sanal noktaların belirlenmesinde kullanılan değerlerin, oluşturulan rota üzerindeki etkisi Berlin52 problemi üzerinde

incelenmiştir. Elde edilen sonuçlar bu kısmın sonunda tablo ile gösterilmiştir.

6.2.2 Kübik Bezier Eğrileri

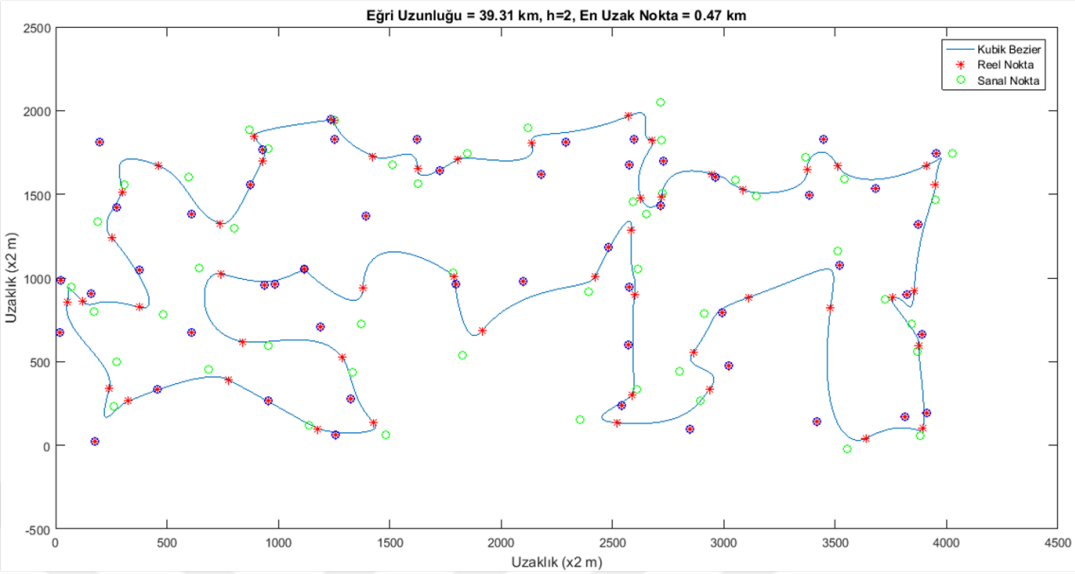
Kübik Bezier Eğrileri eklenen sanal noktalar ile birlikte C^1 sürekliliğini sağlayacak şekilde Berlin52 problemine uygulanmıştır. Oluşturulan İHA rotası Şekil 6.7'de sunulmuştur. Eğrinin uzunluğu h=2 seçildiği durumda 33,15 km olarak hesaplanmıştır. Kübik Bezier Eğrileri kullanılarak üretilen eğrinin ilk dört noktası kullanılarak oluşturulan bölümünde iki kontrol noktası eğrinin dışında kalmaktadır. Diğer eğri parçalarının her birinde ise arada kalan noktalardan biri sanal noktadır. Bu durumda bir kontrol noktası oluşturulan eğrinin dışında kalmaktadır. Dışarıda kalan noktaların eğri ile arasındaki uzaklıkları incelendiğinde maksimum uzaklık 1,05 km'dir. Toplam 26 nokta eğrinin dışında kalmaktadır. Kontrol noktalarının eğriye uzaklığının ortalaması 0,13 km olarak hesaplanmıştır.



Şekil 6.7: Kübik Bezier Eğrisi ile Berlin52 Problemi Çözümü (h=2).

KroA100 problemine Kübik Bezier Eğrileri ile yumuşatma uygulandıktan sonra oluşan rota Şekil 6.8 ile sunulmuştur. Rotanın uzunluğu h=2 için 39,31 km dir. Eğrinin en uzağındaki nokta 0,47 km uzaklıktadır. 51 kontrol noktası eğrinin

dışında kalmaktadır. Kontrol noktalarının eğriye ortalama uzaklığı 0,05 km'dir.

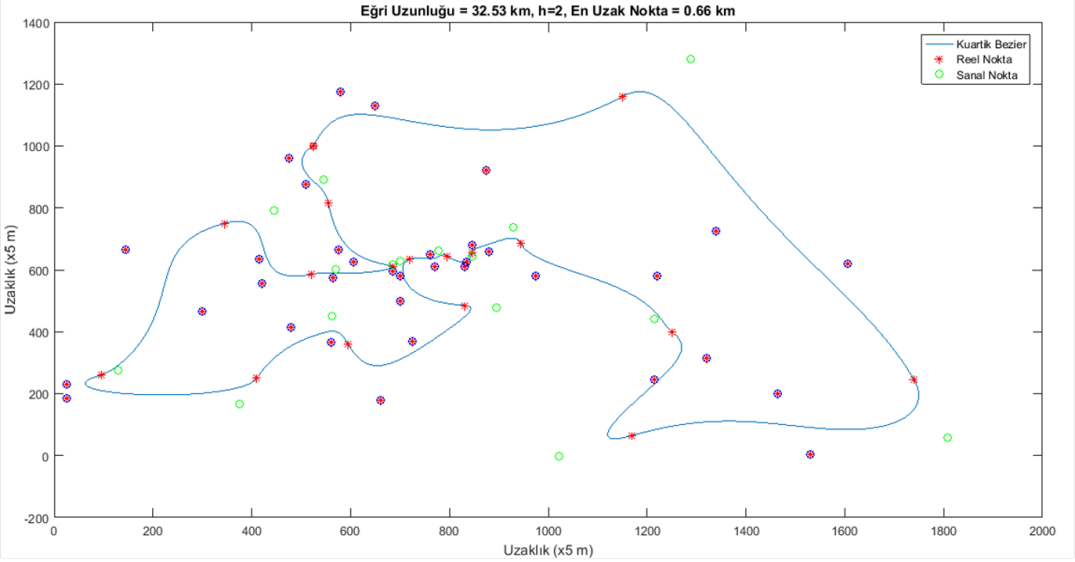


Şekil 6.8: Kübik Bezier Eğrisi ile KroA100 Problemi Çözümü (h=2).

Kübik Bezier Eğrilerine eklenen sanal noktaların belirlenmesinde kullanılan değerlerin, oluşturulan rota üzerindeki etkisi Berlin52 problemi üzerinde incelenmiştir. Elde edilen sonuçlar bu kısmın sonunda tablo ile gösterilmiştir.

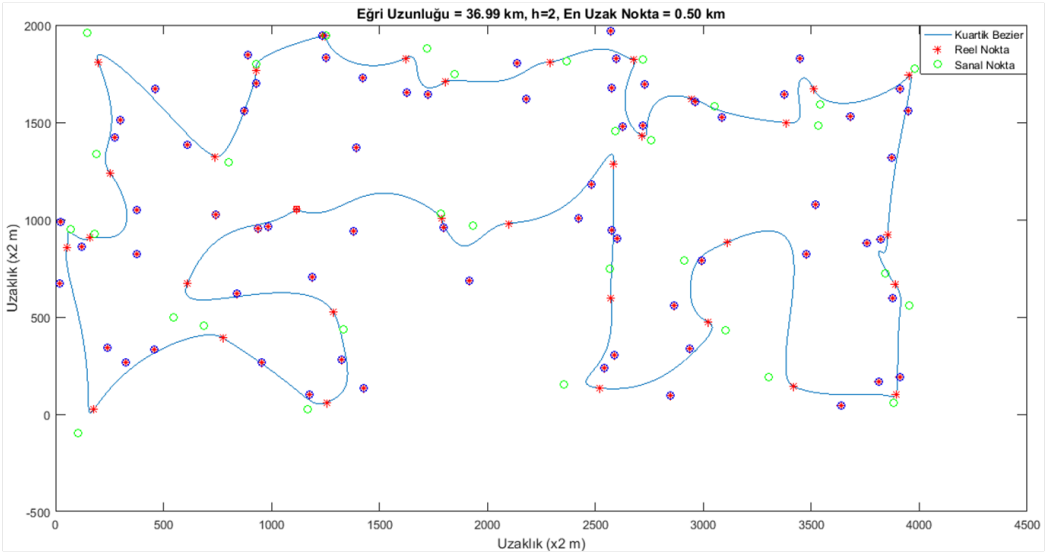
6.2.3 Kuartik Bezier Eğrileri

Kuartik Bezier Eğrileri C^1 sürekliliğini sağlayacak şekilde sanal noktalar eklenerek Berlin52 problemine uygulanmıştır. Elde edilen İHA rotası Şekil 6.9'da sunulmuştur. Eğrinin uzunluğu h=2 seçildiğinde 32,53 km olarak hesaplanmıştır. Kuartik Bezier Eğrileri kullanılarak üretilen eğrinin ilk parçasında üç kontrol noktası eğrinin dışında kalmaktadır. Diğer eğri parçalarında, sanal noktalar bulunduğu için, her eğri parçasında iki kontrol noktası eğrinin dışında kalmaktadır. Eğrinin üzerinden geçmediği en uzak kontrol noktası eğriye 0,66 km uzaklıktadır. Toplam 35 nokta eğrinin dışında kalmaktadır. Kontrol noktalarının eğriye olan uzaklığının ortalaması 0,16 km olarak hesaplanmıştır.



Şekil 6.9: Kuartik Bezier Eğrisi ile Berlin52 Problemi Çözümü (h=2).

Kuartik Bezier Eğrilerinin KroA100 problemine uygulanması sonucu oluşturulan rota Şekil 6.10 ile sunulmuştur. Rotanın uzunluğu h=2 için 36,99 km'dir. Eğrinin en uzağındaki nokta 0,50 km uzaklıktadır. 67 kontrol noktası eğrinin dışında kalmaktadır. Kontrol noktalarının eğriye uzaklıklarının ortalaması 0,08 km'dir.



Şekil 6.10: Kuartik Bezier Eğrisi ile KroA100 Problemi Çözümü (h=2).

Kuadratik, Kübik ve Kuartik Bezier Eğrileri ile üretilen rotalar incelendiğinde

problemin çözümü için kullanılan eğrinin derecesi arttıkça oluşan rotanın toplam uzunluğunun azaldığı görülmüştür. Bununla birlikte eğrinin üzerinden geçmediği kontrol noktası sayısı arttığı için maliyet fonksiyonu olarak belirlediğimiz kontrol noktaları ile eğri arasındaki ortalama uzaklığın arttığı görülmüştür.

Oransal uzaklıkta eklenecek sanal noktaların belirlenmesinde kullanılan h parametresi artırıldığında eklenen sanal noktaların eğriye etkisinin azaldığı gözlemlenmiştir. h değerleri çok fazla artırıldığında ise birleşim noktalarında eğrinin daha keskin dönüşler içerdiği görülmektedir. Bu nedenle Berlin52 problemi üzerinde h parametresi sırasıyla 2, 3, 5 ve 8 değerleri olarak seçilmiş ve elde edilen sonuçlar Tablo 6.2'de ifade edilmiştir. Tablo 6.2'de ifade edilen diğer hususlar ise oluşturulan eğrilerin uzunlukları, eğrinin dışında kalan kontrol noktalarından eğriye en uzak olanın uzaklığıdır. Ayrıca kontrol noktalarının eğriye uzaklıklarının ortalaması ortalama maliyet olarak ifade edilmiştir.

Tablo 6.2: Bezier Eğrilerinin h Parametresine Göre Değişiminin İncelenmesi

	Düzgün Kuadratik Bezier (2.derece)			**Düzgün Kübik Bezier (3.derece)**			**Düzgün Kuartik Bezier (4.derece)**		
	Eğri Uzunluğu (km)	**En Uzak Nokta (km)**	**Ort. Maliyet (km)**	**Eğri Uzunluğu (km)**	**En Uzak Nokta (km)**	**Ort. Maliyet (km)**	**Eğri Uzunluğu (km)**	**En Uzak Nokta (km)**	**Ort. Maliyet (km)**
N=10000,h=2	46,80	0	0	33,15	1,05	0,13	32,53	0,66	0,16
N=10000,h=3	42,47	0	0	32,18	1,01	0,13	31,80	0,66	0,15
N=10000,h=5	40,22	0	0	31,51	0,98	0,13	31,30	0,66	0,15
N=10000,h=8	39,28	0	0	31,19	0,96	0,13	31,06	0,66	0,15

Düzgün Kuartik Bezier Eğrilerinde eğriye en uzak olan nokta birinci eğri parçasında bulunduğu için h değerinin değişiminden etkilenmemektedir.

Oransal uzaklıkta sanal nokta eklendiği durumda; sanal noktanın konumunu belirleyen iki kontrol noktasının arasındaki uzaklık fazla olduğu durumda, sanal noktanın etkisi bu uzaklık ile doğru orantılı olarak artmaktadır. h değerinin değiştirilmesi orantı sabitinin değişmesine neden olmaktadır.

Bezier Eğrilerinde sanal noktaların belirlenmesi için kullanılan bir diğer yöntem ise sabit uzaklıkta sanal noktaların eklenmesidir. Bu yöntemle sanal noktalar belirlenirken, bir önceki eğrinin sona erdiği kontrol noktasına sabit uzaklıkta bir nokta olarak seçilmiştir. Sanal nokta oluşturmak için kullanılan $h^{\star}$ parametresi sırasıyla 40, 35, 30 ve 20 değerleri olarak seçilmiş ve oluşan sonuçlar Tablo 6.3'te ifade edilmiştir.

Tablo 6.3'te tüm sanal noktaların aynı uzunluk kullanılarak eklenmesi sonucunda elde edilen eğri uzunlukları ve eğrinin üzerinden geçmediği eğriye en uzak olan kontrol noktasının uzaklığı belirtilmiştir. Ayrıca eğrinin üzerinden geçmediği noktaların eğriye uzaklıklarının toplamının toplam kontrol noktası sayısına bölünerek elde edilen ortalama maliyet değerleri gösterilmiştir.

Tablo 6.3: Bezier Eğrilerinin $h^\star$ Parametresine Göre Değişiminin İncelenmesi.

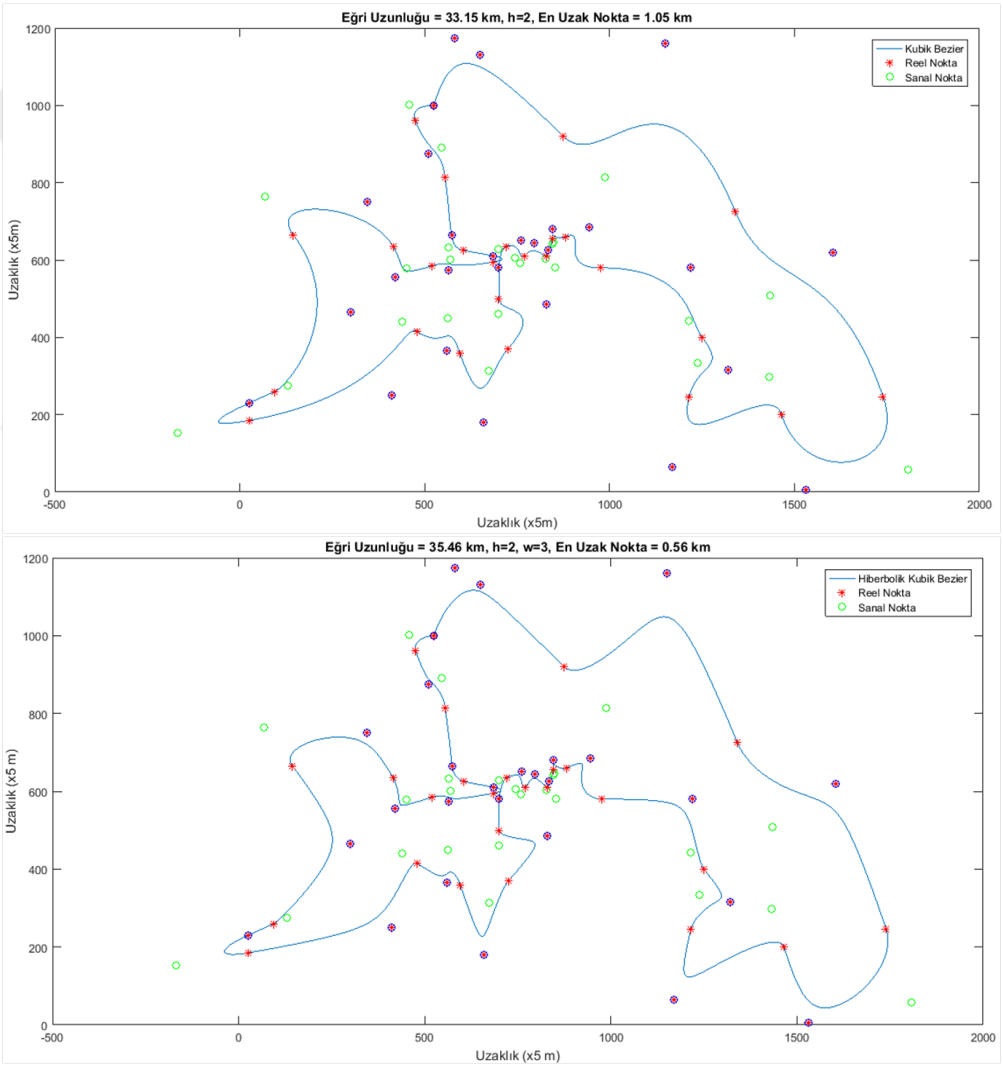
	Düzgün Kuadratik Bezier (2.derece)			**Düzgün Kübik Bezier (3.derece)**			**Düzgün Kuartik Bezier (4.derece)**		
	Eğri Uzunluğu (km)	**En Uzak Nokta (km)**	**Ort. Maliyet (km)**	**Eğri Uzunluğu (km)**	**En Uzak Nokta (km)**	**Ort. Maliyet (km)**	**Eğri Uzunluğu (km)**	**En Uzak Nokta (km)**	**Ort. Maliyet (km)**
N=10000,$h^\star$=40	41,27	0	0	31,57	0,96	0,13	31,34	0,66	0,15
N=10000,$h^\star$=35	40,73	0	0	31,43	0,96	0,13	31,25	0,66	0,15
N=10000,$h^\star$=30	40,23	0	0	31,31	0,95	0,13	31,16	0,66	0,15
N=10000,$h^\star$=20	39,36	0	0	31,09	0,95	0,13	31,00	0,66	0,15

Düzgün Kuartik Bezier Eğrilerinde eğriye en uzak olan nokta birinci eğri parçasında bulunduğu için $h^\star$ değerinin değişiminden etkilenmemektedir.

6.2.4 Rasyonel Bezier Eğrileri (Hiperbolik Bezier Eğrileri)

Rasyonel Bezier Eğrileri, kontrol noktalarının eğriye etkisini artırmak veya azaltmak için ağırlık katsayısı oluşturma işleminde kullanılmaktadır. Ağırlık katsayısının noktanın eğriye etkisini artırdığı durum Hiperbolik Bezier Eğrileridir. Eğrinin üzerinden geçmediği kontrol noktalarının ağırlık katsayıları artırılarak eğriye olan uzaklıkların azaltılması sağlanmıştır.

Kübik Bezier Eğrileri ve Hiperbolik Kübik Bezier Eğrileri uygulanması ile oluşturulan Berlin52 problemi rotaları Şekil 6.11'de sunulmuştur.



Şekil 6.11: Kübik Bezier Eğrisi ve Hiperbolik Kübik Bezier Eğrisi (h=2, w=3)

Kuadratik Bezier Eğrilerinde eğrinin dışında kontrol noktası kalmadığı için

Kübik ve Kuartik Bezier Eğrilerine rasyonel katsayı uygulanmıştır. Arada kalan noktalardan biri sanal nokta olduğu için sadece reel kontrol noktalarına hiperbolik katsayı uygulanmıştır. Ağırlığı artırmak için uygulanan katsayı w parametresi ile ifade edilmiştir. Eğrilerin uzunluğuna etki eden birden fazla parametre olduğu için Şekil 6.11'de sunulan iki çözümde de $h = 2$ olarak belirlenmiştir. Sanal noktalar ve eğrinin üzerinden geçtiği kontrol noktaları için ağırlık katsayısı 1 olarak seçilmiş olup, w=3 olarak seçilmiştir. Sanal noktalar ve reel kontrol noktaları farklı sembollerle gösterilmiştir. Hiperbolik Kübik Bezier Eğrileri kullanılarak üretilen çözümde, eğrinin dışında kalan kontrol noktasına bir yakınsama olduğu açıkça görülmektedir.

Ağırlık katsayısı w artırıldıkça ilgili kontrol noktasına doğru eğride oluşan yönelim artmaktadır. w parametresi sırasıyla 2, 3, 5 ve 8 değerleri seçilmiş ve elde edilen sonuçlar Tablo 6.4'te gösterilmiştir.

Tablo 6.4: Rasyonel Bezier Eğrilerinin w Katsayılarına Göre Değişimi.

	Hiperbolik Düzgün Kübik Bezier (3.derece) (w1)				**Hiperbolik Düzgün Kuartik Bezier (4.derece)(w1,w2)**			
N=10000,h=2	**Eğri Uzunluğu**	**En Uzak Nokta**	**Ort. Maliyet**	**Maliyet Kar Yüzdesi**	**Eğri Uzunluğu**	**En Uzak Nokta**	**Ort. Maliyet**	**Maliyet Kar Yüzdesi**
w1=w2=1	33,15	1,05	0,13	%0	32,47	0,66	0,16	%0
w1=w2=2	34,70	0,73	0,09	%30,77	32,84	0,54	0,13	%18,75
w1=w2=3	35,46	0,56	0,07	%46,15	33,09	0,47	0,12	%25,00
w1=w2=5	36,39	0,38	0,05	%61,54	33,46	0,44	0,11	%31,25
w1=w2=8	37,03	0,35	0,04	%69,23	33,83	0,45	0,09	%43,75

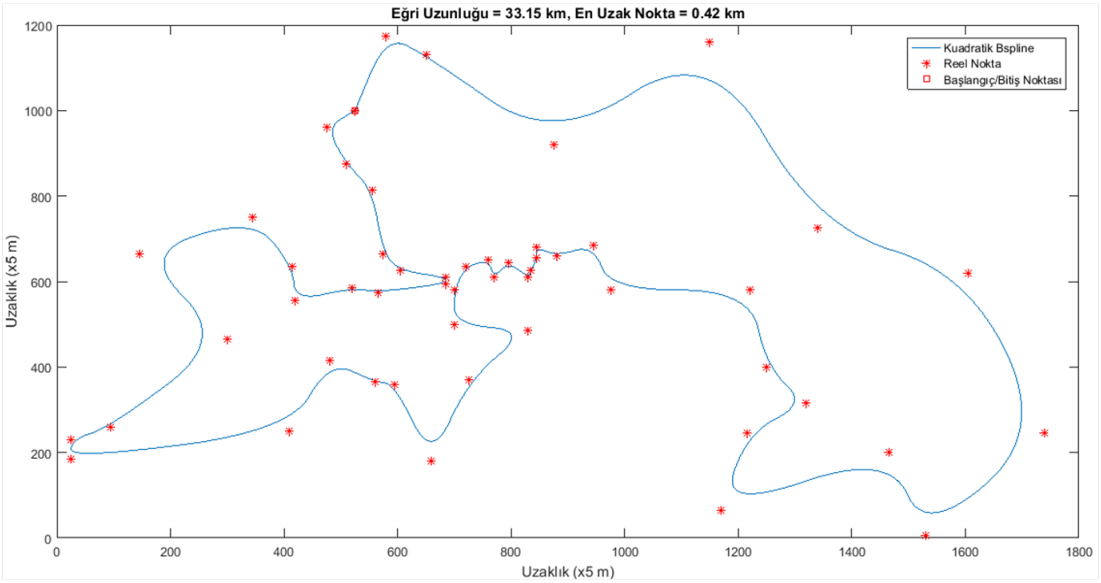
Tablo 6.2'de $h = 2$ olarak seçilmesi sonucu elde edilen sonuçlar göz önünde bulundurulmuş ve w katsayılarının artması sonucunda ortalama maliyette oluşan değişim incelenmiştir. Ortalama maliyette meydana gelen değişiklik ise "Maliyet Kar Yüzdesi" olarak belirtilmiştir. Eğrinin dışında kalan kontrol noktalarından en uzak olanın uzaklığındaki değişim incelenmiştir.

6.3 B-Spline Eğrileri

B-Spline Eğrileri eğriyi üreten polinomun derecesinin bir fazlası kadar kontrol noktası kullanılarak üretilen eğri parçalarından oluşmaktadır. B-Spline Eğrilerinin sürekliliği ise eğriyi üreten polinomun derecesi k olmak üzere C^{k-2}dir. Oluşturulan eğriler başlangıç noktası dışındaki kontrol noktalarının üzerinden geçmediği için kontrol noktalarının eğriye uzaklıklarının toplamı hesaplanmış ve bu sayı toplam kontrol noktası sayısına bölünerek ortalaması bulunmuştur. Noktaların eğriye uzaklığının ortalaması maliyet fonksiyonu olarak belirlenmiştir.

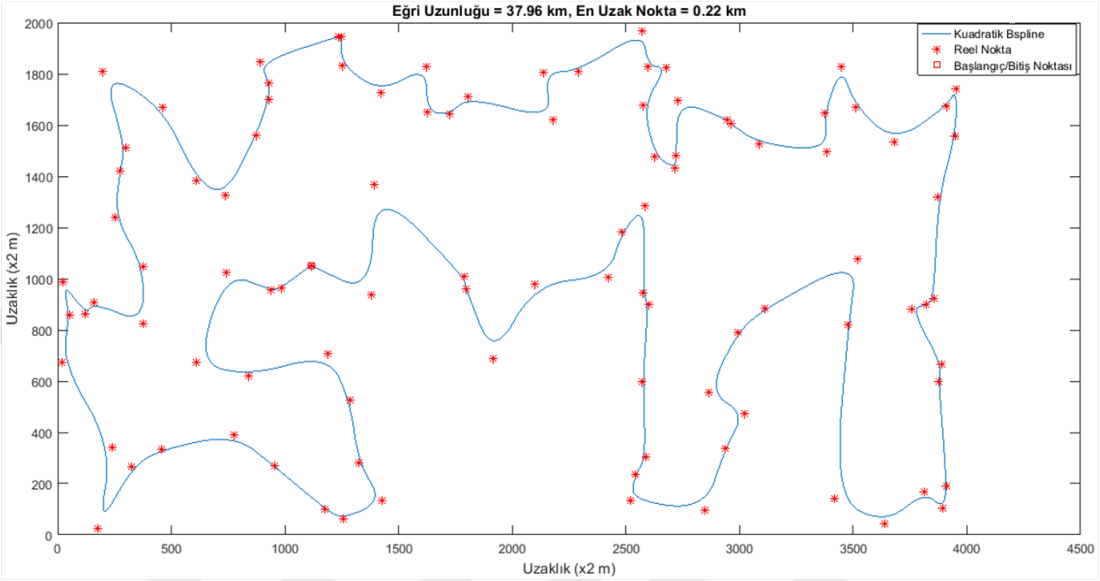
6.3.1 Kuadratik B-Spline Eğrileri

Kuadratik B-Spline Eğrilerinin Berlin52 Problemine uygulanması sonucunda elde edilen İHA rotası Şekil 6.12 ile sunulmuştur. Eğrinin uzunluğu 33,15 km olarak hesaplanmıştır. Eğriye en uzak kontrol noktasının uzaklığı 0,42 km olarak hesaplanmıştır. Kontrol noktalarının eğriye olan uzaklıklarının ortalaması 0,10 km olarak hesaplanmıştır.



Şekil 6.12: Kuadratik B-Spline Eğrisi ile Berlin52 Problemi Çözümü

KroA100 problemine Kuadratik B-Spline Eğrisi ile yumuşatma uygulanması sonucu oluşan rota Şekil 6.13 ile sunulmuştur. Oluşturulan eğrinin uzunluğu 37,96 km'dir. Eğriye en uzak kontrol noktası 0,22 km mesafededir. Kontrol noktalarının eğriye uzaklıklarının ortalaması 0,05 km'dir.

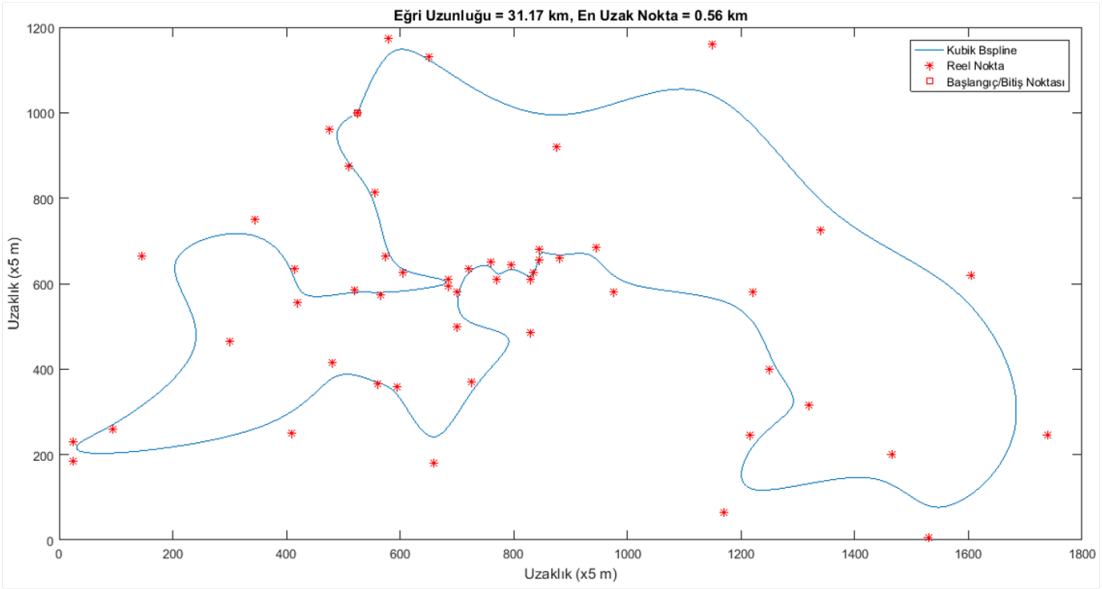


Şekil 6.13: Kuadratik B-Spline Eğrisi ile KroA100 Problemi Çözümü

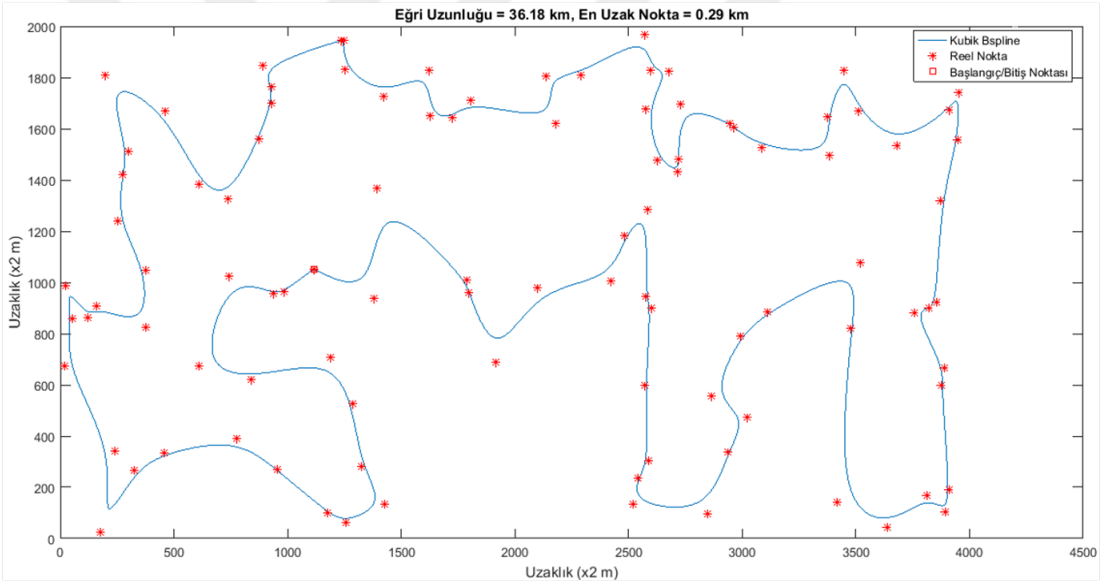
6.3.2 Kübik B-Spline Eğrileri

Kübik B-Spline Eğrilerinin Berlin52 problemine uygulanması sonucunda elde edilen İHA rotası Şekil 6.14 ile gösterilmiştir. Elde edilen eğrinin uzunluğu 31,17 km olarak hesaplanmıştır. Eğriye en uzak kontrol noktasının uzaklığı 0,56 km olarak hesaplanmıştır. Kontrol noktalarının eğriye olan uzaklıklarının ortalaması 0,13 km olarak hesaplanmıştır.

KroA100 problemine Kübik B-Spline Eğrisi uygulanması sonucunda elde edilen rota Şekil 6.15 ile sunulmuştur. Oluşturulan eğrinin uzunluğu 36,18 km'dir. Eğriye en uzak kontrol noktası 0,29 km uzaklıktadır. Kontrol noktalarının eğriye uzaklıklarının ortalaması 0,06 km'dir



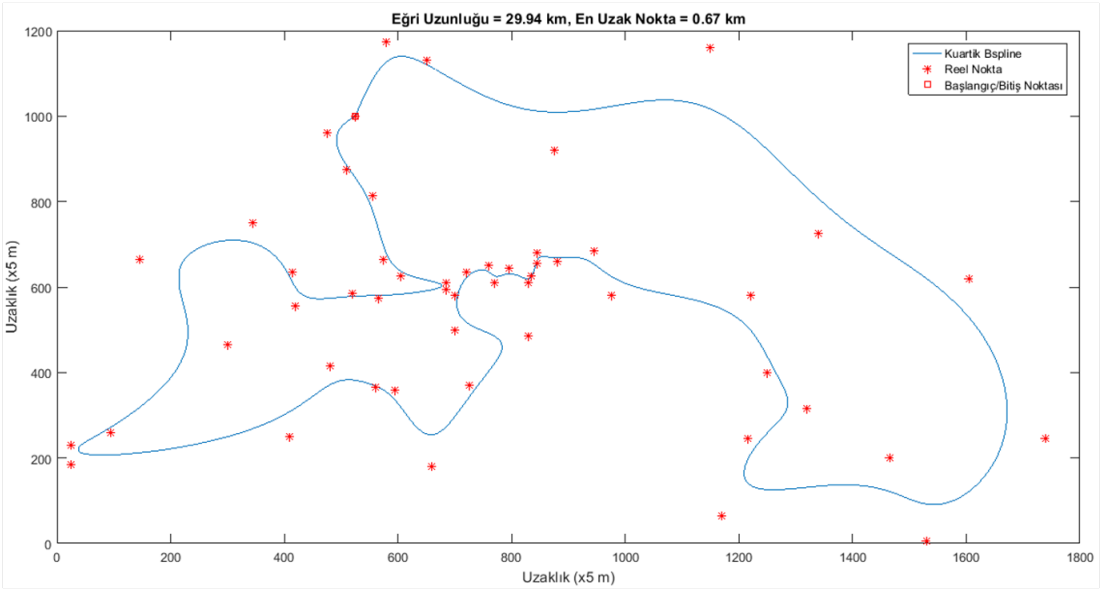
Şekil 6.14: Kübik B-Spline Eğrisi ile Berlin52 Problemi Çözümü



Şekil 6.15: Kübik B-Spline Eğrisi ile KroA100 Problemi Çözümü

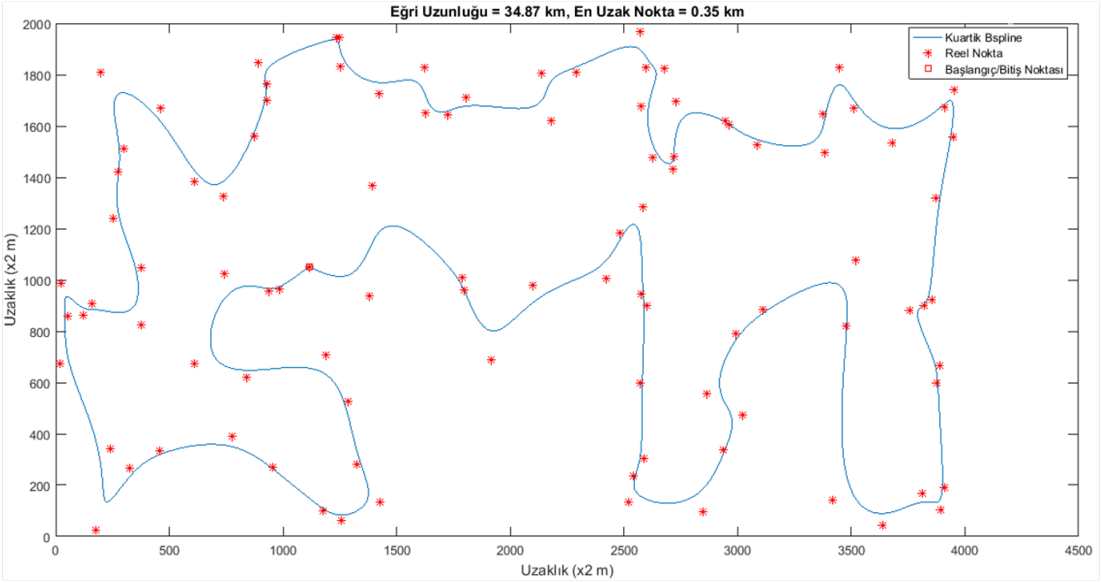
6.3.3 Kuartik B-Spline Eğrileri

Kuartik B-Spline Eğrilerinin Berlin52 problemine uygulanması sonucunda elde edilen İHA rotası Şekil 6.16 ile gösterilmiştir. Elde edilen eğrinin uzunluğu 29,94 km olarak hesaplanmıştır. Eğriye en uzak kontrol noktasının uzaklığı 0,67 km olarak hesaplanmıştır. Kontrol noktalarının eğriye olan uzaklıklarının ortalaması 0,16 km olarak hesaplanmıştır.



Şekil 6.16: Kuartik B-Spline Eğrisi ile Berlin52 Problemi Çözümü

KroA100 problemine Kuartik B-Spline Eğrisi uygulanması sonucunda elde edilen rota Şekil 6.17 ile sunulmuştur. Oluşturulan eğrinin uzunluğu 34,87 km'dir. Eğriye en uzak kontrol noktası 0,35 km uzaklıktadır. Kontrol noktalarının eğriye uzaklıklarının ortalaması 0,07 km'dir.



Şekil 6.17: Kuartik B-Spline Eğrisi ile KroA100 Problemi Çözümü

B-Spline Eğrileri ile Berlin52 probleminin çözümü sonrası elde edilen sonuçlar

Tablo 6.5'te gösterilmiştir.

Tablo 6.5: B-Spline Eğrileri Kullanılarak Elde Edilen Sonuçlar.

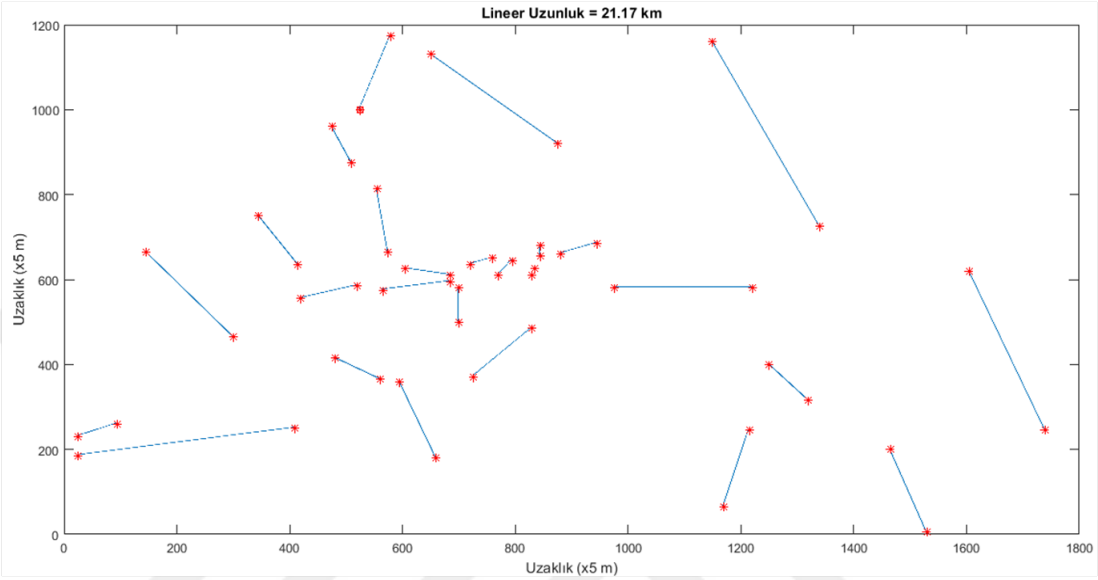
Kuadratik B-Spline (2.derece) N=10000			**Kübik B-Spline (3.derece) N=10000**			**Kuartik B-Spline (4.derece) N=10000**		
Eğri Uzunluğu (km)	**En Uzak Nokta (km)**	**Ort. Maliyet (km)**	**Eğri Uzunluğu (km)**	**En Uzak Nokta (km)**	**Ort. Maliyet (km)**	**Eğri Uzunluğu (km)**	**En Uzak Nokta (km)**	**Ort. Maliyet (km)**
33,15	0,42	0,10	31,17	0,56	0,13	29,94	0,67	0,16

Tablo 6.5 incelendiğinde B-Spline Eğrilerinin derecesi arttıkça eğrinin uzunluğunun azaldığı gözlemlenmiştir. Ancak bunun yanında ortalama maliyetin arttığı da görülmektedir. Burada karar verirken kısıtlarımız devreye girecektir. Kısıtlarımızı karşılayan en uygun yöntem bizim tercih edeceğimiz yöntem olacaktır.

6.4 Dubins Yolu

TSP problemlerinin GA ile çözülmesi sonrasında kontrol noktalarının sırasını ve konum bilgilerini içeren metin tabanlı bir dosya oluşturmaktadır. Ancak Dubins Yollarının oluşturulması sırasında konum bilgisinin yanında İHA'nın yön bilgisine de ihtiyaç duyulmaktadır. Yön bilgisinin oluşturulması için başlangıç noktası ile ikinci sıradaki kontrol noktası doğrusal olarak birleştirilir. Bu durumda birinci sıradaki kontrol noktasının yönü ikinci sıradaki kontrol noktasına doğru oluşturulmuş olur. İkinci sıradaki kontrol noktası da aynı doğru üzerinde olduğundan birinci kontrol noktası ile aynı yöne sahip olur. Benzer şekilde üçüncü kontrol noktasının yönü dördüncü kontrol noktasına doğru belirlenir. Yön açıları vektörler kullanılarak hesaplanır. Toplam kontrol noktası sayısının n olduğu durumda $s = \left[\left|\frac{n}{2}\right|\right]$ olarak belirlenir. $i = 1, 2 \ldots, s$ olmak üzere P_{2i-1} ve P_{2i} noktaları arasında $\vec{V_i} = P_{2i} - P_{2i-1}$ vektörleri tanımlanır. Oluşturulan vektörün yönü $\theta_i = atan2(\vec{V_i})$ fonksiyonu ile hesaplanır. P_{2i-1} ve P_{2i}

kontrol noktaları aynı vektör üzerinde olduğundan yönleri de aynı olacaktır. Bu durumda kontrol noktaları $P_{2i-1} = (x_{2i-1}, y_{2i-1}, \theta_i)$ ve $P_{2i} = (x_{2i}, y_{2i}, \theta_i)$ şeklinde düzenlenir. Berlin52 probleminde yapılan bu ilk düzenleme ile oluşturulan doğrusal yollar Şekil 6.18'de sunulmuştur.

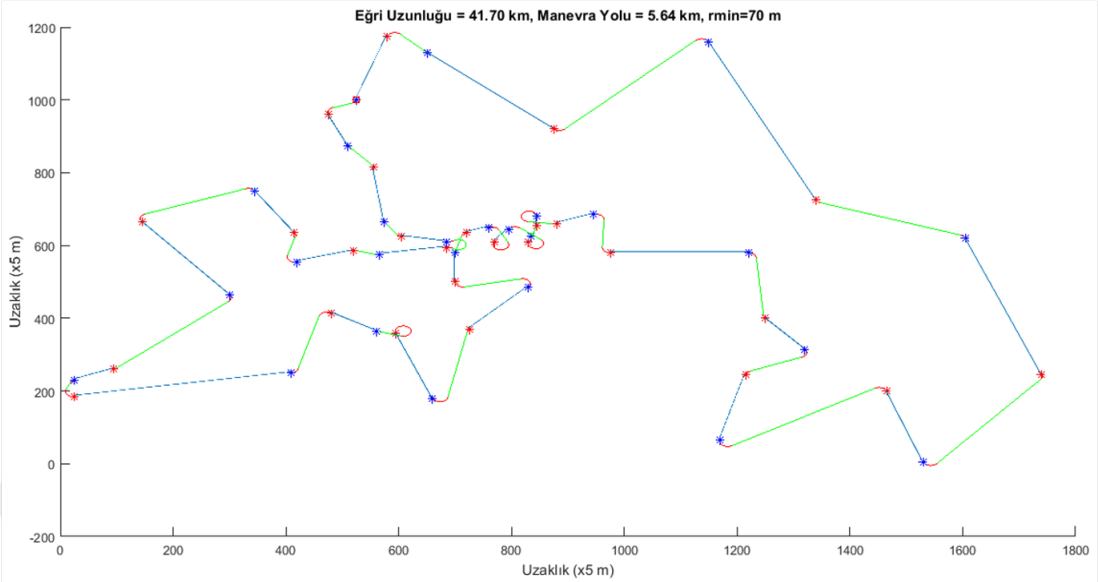


Şekil 6.18: Dubins Yolu İçin Berlin52 Problemi Ön Hazırlık

Kontrol noktası sayısının çift sayı olması durumunda herhangi bir nokta açıkta kalmayacaktır. Tek sayı olması durumunda başlangıç noktası ile son kontrol noktasının ortasına sanal bir kontrol noktası eklenir. Bu sanal kontrol noktasının eğrinin uzunluğuna etkisi olmayacaktır.

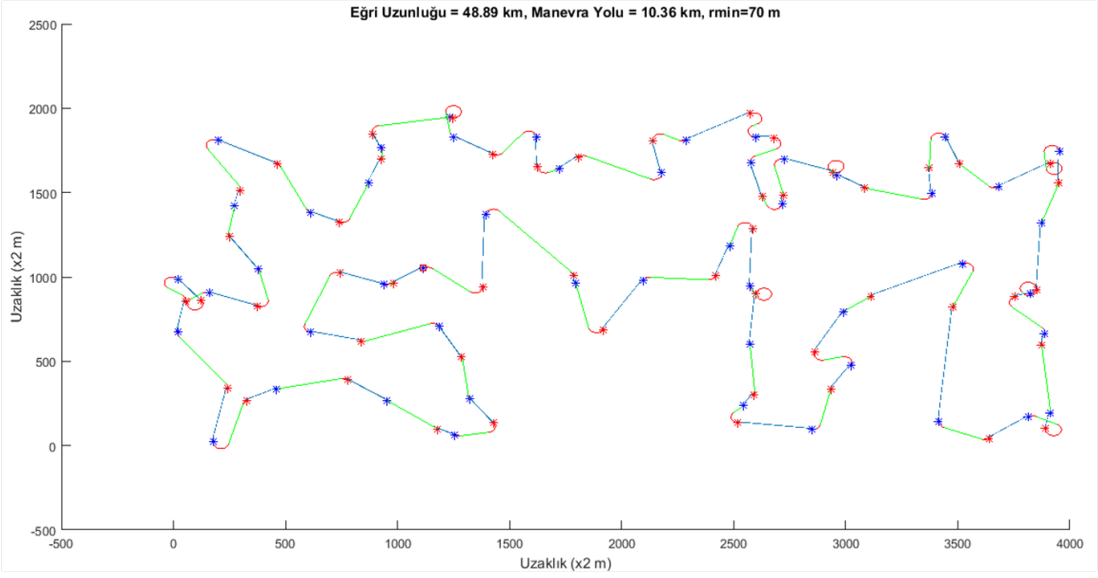
Oluşturulan doğrusal yolların arasında kalan kısımlar Dubins Yolları kullanılarak birleştirilecektir. Doğrusal yolların birleştirilmesi sırasında altı Dubins yolundan en uygunu (en kısa olanı) seçilerek birleştirilir. İHA'nın minimum dönüş yarıçapı hesaba katılarak, yollar tanımlı oldukları bölgelerde MATLAB programı ile üretilmiş ve en kısa olan yol tercih edilmiştir. Yollar hesaplanırken dönüş kapsamında gerçekleştirilen yollar ayrıca hesaplanmıştır. CCC yollarının tamamı dönüşlerden oluşurken, CLC yolları başlangıçta ve sonda dönüş ortada ise doğrusal bir yoldan oluşmaktadır. İHA'nın minimum dönüş yarıçapı $r_{min} = 70$ m seçilerek oluşturulan Berlin52 çözümü Şekil 6.19'da sunulmuştur. Oluşturulan eğrinin uzunluğu 41,70 km olarak hesaplanmıştır.

Eğri problemde belirtilen tüm kontrol noktalarının üzerinden geçmektedir.



Şekil 6.19: Dubins Yolu ile Berlin52 Problemi Çözümü ($r_{min} = 70$ m)

KroA100 problemine Dubins Yolu uygulanması sonucunda elde edilen rota Şekil 6.20 ile sunulmuştur. Oluşturulan eğrinin uzunluğu, $r_{min} = 70$ m seçildiği durumda 48,89 km olarak hesaplanmıştır. Yolun manevralardan oluşan kısmı 10,36 km'dir.



Şekil 6.20: Dubins Yolu ile KroA100 Problemi Çözümü ($r_{min} = 70$ m)

Berlin52 Probleminin ayrıntılı uzaklıkları ve İHA'nın minimum dönüş yarıçapı, r_{min}, sırasıyla 40, 55, 70, 85 ve 100 m olarak seçildiğinde alınan sonuçlar Tablo 6.6 ile sunulmuştur.

Tablo 6.6: Dubins Yolu Kullanılarak Elde Edilen Sonuçlar

r_{min} (m)	Doğrusal Uzunluk (km)	Dubins Yolu		Eğri Uzunluğu (km)
		Dubins Manevra Uzunluğu (km)	Dubins "S" Yolu Uzunluğu (km)	
40	21,17	2,83	15,55	39,55
55	21,17	4,25	14,78	40,20
70	21,17	5,64	14,89	41,70
85	21,17	7,82	14,21	43,19
100	21,17	9,86	13,35	44,38

BÖLÜM 7

SONUÇ VE ÖNERİLER

Bu çalışmada kısıtlara bağlı matematiksel modelleme ile İHA için yumuşatılmış rota planlama problemi ele alınmıştır. Belirlenen kontrol noktalarının tam olarak üzerinden veya yakınından uçması istenen İHA için, oluşturulan rotanın eğrilik kısıtını karşılaması sağlanmıştır. Üç farklı matematiksel yöntem kullanılarak örnek problemler üzerinde rotalar oluşturulmuştur. Kullanılan yöntemler ile elde edilen rotaların birbirleri ile karşılaştırılması sağlanmıştır. Problem iki aşamada incelenmiştir. Birinci aşamada ziyaret edilmesi istenen kontrol noktalarının hangi sıra ile ziyaret edileceği problemi ele alınmıştır. Bu problem 19'uncu yüzyıl başlarından itibaren popülerliğini koruyan TSP'nin bir uygulamasıdır. NP-Hard olarak sınıflandırılan TSP uygun bir zaman içerisinde, optimal sonuca yakın bir sonuç bulan GA yöntemi kullanılarak çözülmüştür. Bu konuda detaylı bilgi 4'üncü Bölüm'de verilmiştir.

GA ile çözülen TSP ziyaret edilmesi planlanan kontrol noktalarını birbirine doğrusal olarak bağlayan bir rota oluşturmuştur. Oluşturulan bu rotanın yumuşatılması için problemin ikinci aşamasında; matematiksel yöntemler kullanılarak rotaya eğrilik kazandırılmıştır. Yumuşatma işlemi için yaygın olarak kullanılan üç yöntem tercih edilmiştir. Bunlar Bezier Eğrileri, B-Spline Eğrileri ve Dubins Yolu yöntemleridir.

Bezier Eğrileri, derecesi kontrol noktası sayısının bir eksiğine eşit olan yaklaştırma eğrileridir. Örnek problemlere 2'nci, 3'üncü ve 4'üncü derece Bezier Eğrileri uygulanmıştır. Bezier Eğrileri, eğriyi oluşturan kontrol noktalarından

baştakine ve sondakine tam olarak "dokunurken" arada kalan kontrol noktalarına yakınsamaktadır. Bezier Eğrileri'nin parçalı olarak kullanılması sonucu iki eğrinin birleşim noktasında karşılaşılabilecek sivri ve pürüzlü yollar sanal nokta eklenerek giderilmiştir. Bezier Eğrileri belirlenen iki farklı sanal nokta ekleme yöntemi ile C^1 sürekli hale getirilmiştir. Sanal noktalar parametrelere bağlı olarak eklenmiş ve parametrelerdeki değişime göre sanal noktaların eğrinin uzunluğuna etkileri incelenmiştir. Sanal noktaların eğriye etkilerinin azaltılması durumunda eğrinin uzunluğu azalmaktadır. Ancak bunun yanında eğrinin şeklinde bozulma olmaktadır. Ziyaret edilmeyen sadece bir kontrol noktası kalarak üretilen Kuadratik Bezier Eğrileri ekstra sanal noktaların araya eklenmesi ile ziyaret edilmesi planlanan tüm kontrol noktalarının üzerinden geçecek şekilde düzenlenmiştir.

İki polinomun birbirine bölünmesi ile oluşan Rasyonel Bezier Eğrileri ve eğri üzerinde oluşturdukları etki incelenmiştir. Özel olarak arada kalan kontrol noktalarına eğrinin daha yakından geçmesini sağlayan Hiperbolik Bezier Eğrileri bir parametreye bağlı olarak oluşturulmuştur. Oluşturulan parametre değerleri incelenmiş ve sonuçları tablo ile verilmiştir. Ziyaret edilmeyen kontrol noktasına eğrinin yaklaştırılması ile birlikte eğrinin uzunluğu da artmaktadır. Ayrıca parametrenin çok yüksek değerde seçilmesi sonucunda eğrinin şeklinin bozulmaya başladığı görülmüştür.

Bezier Eğrileri'nin derecesi yükseldikçe oluşturulan rotanın uzunluğu azalmaktadır. Ancak ziyaret edilmeyen kontrol noktası sayısı ve kontrol noktaları ile eğri arasında bulunan uzaklık ortalaması artmaktadır. Bu durumda kullanılacak olan İHA'nın görevi ve kullanacağı teçhizat önem arz edecektir.

Rotanın yumuşatılmasında kullanılan diğer bir yöntem ise B-Spline Eğrileri'dir. B-Spline Eğrileri'nin derecesi kontrol noktası sayısından bağımsızdır. B-Spline Eğrileri; 3'üncü, 4'üncü ve 5'inci mertebe olarak sırasıyla 2'nci, 3'üncü ve 4'üncü derece polinomlardan oluşmaktadır. B-Spline Eğrileri mertebesinin k olduğu durumda, C^{k-2} sürekliliğine sahiptir. Bölgesel tanımlı olarak üretilen eğrilerde, kontrol noktalarından birinin değişmesi sonucunda sadece ilgili bölgede oluşturulan eğri parçası değişir. B-Spline Eğrileri'nin genliği 1'dir. Yani bir noktada verilen eğrilerin toplamı 1'dir.

B-Spline Eğrileri derecesi bilinen eğriler olduğundan, süreklilik ve genlik özellikleri kullanılarak polinom katsayıları elde edilmiştir. Oluşturulan B-Spline Eğrileri aynı zamanda bitiş noktası olan, başlangıç noktasına ulaşmamıştır. Ancak bu sorun noktanın birkaç kez eklenmesi ile çözülmüştür. B-Spline Eğrileri Bezier Eğrilerine göre üzerinden geçmedikleri kontrol noktalarına daha fazla yakınsamaktadırlar. Ancak B-Spline Eğrileri tarafından oluşturulan rota kontrol noktalarının üzerinden geçmemektedir. Bunun sonucunda B-Spline Eğrileri kullanılarak üretilen rota Bezier Eğrileri kullanılarak üretilen rotadan daha kısadır.

Bir diğer rota planlama yöntemi ise Dubins Yolu'dur. İki nokta arasında eğrilik içeren en kısa yol olarak tanımlanan Dubins Yolu, altı yol kümesinden oluşmaktadır. Noktaların konumlarının yanı sıra, yön bilgisine de ihtiyaç duyan Dubins Yolu yöntemi tüm noktaların ziyaret edilmesini garanti altına almaktadır. Belirlenen minimum dönüş yarıçapı ile çemberler oluşturulan yöntem kodlama açısından zorlu bir süreç gerektirmektedir. Geometri ve Analitik Geometri kullanılarak geliştirilen yöntemin uygulanması sırasında ters trigonometrik fonksiyonlar kullanıldığı için, eğrinin tanımlı olmadığı aralıkta üretilmeye çalışılması durumunda hata oluşmaktadır. Bu durumun önüne geçmek için eğrilerin tanımlı oldukları bölgeler belirlenmiştir.

Dubins Yollarının hesaplanması sırasında izlenecek rotanın ne kadarının manevra için kullanıldığı hesaplanmıştır. Bu durumda gerçekleştirilecek manevralar için bir maliyet hesabı yapılabilir. İHA'nın minimum dönüş yarıçapına bağlı olarak değişen Dubins Yolu uzunluğu, $r_{min} = 40$ m seçildiğinde Berlin52 için tüm noktaları ziyaret eden en kısa yolu vermektedir. Kuadratik Bezier Eğrilerinde h ve $h^{\star}$ değerlerinde yapılan düzenleme ile Dubins Yolu ile hemen hemen aynı uzunlukta rotalar elde edilmektedir. Ancak toplam yol uzunluğu olarak düşünüldüğünde B-Spline ve Bezier Eğrilerinden daha uzun bir rota oluşmaktadır.

Sonuç olarak; rota planlamanın ikinci kısmı olan yumuşatma yöntemlerinin aynı problemler üzerinde uygulanması ve sonuçlarının hesaplanması, yöntemler arasında karşılaştırma yapılabilmesini sağlar. Tanımlanan görevin gereksinimlerine göre tercih yapılması ve yapılan tercihin olumlu ve olumsuz

yönlerinin bilinmesi görev planlama için önemli bir husustur. Örneğin; bir İHA görevi tüm noktaların üzerinden geçilmesi koşulunu barındırıyorsa bu durumda Dubins Yolu veya düzenlenen Kuadratik Bezier Eğrisi ile üretilen rota görev için uygun tercih olacaktır. Kontrol noktalarına belirli bir uzaklıktan geçilmesinin yeterli olduğu durumda, Kuartik B-Spline Eğrileri kullanılarak en kısa rota elde edilmektedir. Bu durumda eğriye en uzak noktanın görev tanımına uygun olması gerekir. Bunun yanı sıra yerde bulunan sensörlerden veri toplama, belli bir yükseklikten kamera açısına göre görüntü alma, belirlenen hedeflere uzaktan ateş etme, ormanda yangın kontrolü için gerçekleştirilen güvenlik uçuşları, insan kaçakçılığı ve benzeri kontroller için gerçekleştirilen sınır kontrol devriyesi, doğal yaşam alanında hayvan popülasyonu belirleme, trafik denetleme-radar uygulaması, su kaynaklarının kontrolünün sağlanması, hava kargo gönderimi, miting alanı güvenliği sağlama vb. kontrol noktalarına belli bir mesafe dahilinde icra edilebilecek bir çok görevde görev tanımına uygun olan yöntem tercih edilebilir. Kullanılacak İHA'nın havada kalış süresi, maksimum menzili, gözetleme görevinde ise yakınlaştırma özelliğinin bulunması, hedef kontrol noktası ile arasındaki mesafenin üst sınırını belirlemek için, yangın söndürme görevinde ise etki mesafesi, veri toplama görevi için sinyal menzili vb. hususlar tercih edilecek yöntemin belirlenmesinde önem arz etmektedir.

Dikkat edilecek olursa kontrol noktaları ile oluşturulan eğri arasındaki mesafe azaldıkça eğrinin uzunluğu artmaktadır. Ayrıca oluşturulan eğri ile en uzak kontrol noktasının arasındaki mesafe hesaplanmıştır. Bu sayede en kötü senaryo olarak adlandıracağımız durum kontrolümüz altında olacaktır.

7.1 Öneriler

Çalışmada incelenen yol yumuşatma yöntemleri iki boyutlu koordinat düzlemi üzerinde ele alınmıştır. Zemin seviyesinden sabit yükseklikte ve yer nesneleri ile çarpışmasına engel olacak kadar yüksek irtifada uçan İHA için bu sorun olmayacaktır. Ancak yere yakın uçuşlar gerçekleştirmesi gereken görevlerde göz önünde bulundurularak 3B rota planlamasının yapılmasının uygun olacağı

değerlendirilmektedir. Kontrol noktası sayısının artırılarak çok daha fazla veri içeren örnekler kullanılması gelecek çalışmalar için önerilmektedir. İHA rota planlaması için dağlar, binalar, hava savunma sistemleri, radarlar vb. 3B engeller eklenerek sistem geliştirilebilir.

KAYNAKÇA

[1] S. O'Donnell. (2017) A short history of unmanned aerial vehicles. [Online]. Available: https://consortiq.com/media-centre/blog/short-history-unmanned-aerial-vehicles-uavs.htm

[2] E. Darack. (2011) A brief history of unmanned aircraft. [Online]. Available: https://www.airspacemag.com/photos/a-brief-history-of-unmanned-aircraft-174072843/?page=2.htm

[3] T. Scheve. (2008) A brief history of uavs. [Online]. Available: https://science.howstuffworks.com/reaper.htm

[4] H. Duan and P. Li, *Bio-inspired Computation in Unmanned Aerial Vehicles*. Springer, 2014, ch. History of UAVs.

[5] W. Cook. (2007) History of the tsp. [Online]. Available: http://www.math.uwaterloo.ca/tsp/history/index.html

[6] J. Giesen, "Curve reconstruction, the traveling salesman problem and menger's theorem on length," in *Proceedings of the Fifteenth Annual Symposium on Computational Geometry*, ser. SCG '99. New York, NY, USA: ACM, 1999, pp. 207–216. [Online]. Available: http://ezproxy.iku.edu.tr:2344/10.1145/304893.304973

[7] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem," *Journal of the Operations Research Society of America*, vol. 2, no. 4, pp. 393–410, 1954. [Online]. Available: https://doi.org/10.1287/opre.2.4.393

[8] R. M. Karp, *Reducibility among Combinatorial Problems.* Boston, MA: Springer US, 1972, pp. 85–103.

[9] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management Science*, vol. 6, no. 1, pp. 80–91, 1959. [Online]. Available: https://doi.org/10.1287/mnsc.6.1.80

[10] O. K. Sahingoz, "Flyable path planning for a multi-uav system with genetic algorithms and bezier curves," in *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*, May 2013, pp. 41–48.

[11] M. Li, Z. Yi, and M. Zhu, "Solving tsp by using lotka–volterra neural networks," *Neurocomputing*, vol. 72, no. 16, pp. 3873 – 3880, 2009, financial Engineering Computational and Ambient Intelligence (IWANN 2007). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231209001660

[12] X. Zhang and Y. Ma, "Solving tsp problems with hybrid estimation of distribution algorithms," in *Intelligent Computing Theory*, D.-S. Huang, V. Bevilacqua, and P. Premaratne, Eds. Cham: Springer International Publishing, 2014, pp. 73–81.

[13] I. Pozniak-Koszalka, I. Kulaga, and L. Koszalka, "Gisness system for fast tsp solving and supporting decision making," in *Management of Convergence Networks and Services*, Y.-T. Kim and M. Takano, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 586–589.

[14] A. Zhou, L. Kang, and Z. Yan, "Solving dynamic tsp with evolutionary approach in real time," in *The 2003 Congress on Evolutionary Computation, 2003. CEC '03.*, vol. 2, Dec 2003, pp. 951–957 Vol.2.

[15] X. Zhang and H. Duan, "An improved constrained differential evolution algorithm for unmanned aerial vehicle global route planning," *Applied Soft Computing*, vol. 26, pp. 270 – 284, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494614004992

[16] I. K. Nikolos, K. P. Valavanis, N. C. Tsourveloudis, and A. N. Kostaras, "Evolutionary algorithm based offline/online path planner for uav navigation," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 33, no. 6, pp. 898–912, Dec 2003.

[17] Y. Yu, Y. Chen, and T. Li, "A new design of genetic algorithm for solving tsp," in *2011 Fourth International Joint Conference on Computational Sciences and Optimization*, April 2011, pp. 309–313.

[18] F. Liu and G. Zeng, "Study of genetic algorithm with reinforcement learning to solve the tsp," *Expert Systems with Applications*, vol. 36, no. 3, Part 2, pp. 6995 – 7001, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417408006064

[19] S. S. Ray, S. Bandyopadhyay, and S. K. Pal, "New genetic operators for solving tsp: Application to microarray gene ordering," in *Pattern Recognition and Machine Intelligence*, S. K. Pal, S. Bandyopadhyay, and S. Biswas, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 617–622.

[20] F. Yu, X. Fu, H. Li, and G. Dong, "Improved roulette wheel selection-based genetic algorithm for tsp," in *2016 International Conference on Network and Information Systems for Computers (ICNISC)*, April 2016, pp. 151–154.

[21] Y. Yi and Q. sheng Fang, "The improved hybrid genetic algorithm for solving tsp based on handel-c," in *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE)*, vol. 3, Aug 2010, pp. V3–330–V3–333.

[22] L. Wang, J. Zhang, and H. Li, "An improved genetic algorithm for tsp," in *2007 International Conference on Machine Learning and Cybernetics*, vol. 2, Aug 2007, pp. 925–928.

[23] P. Chen, "An improved genetic algorithm for solving the traveling salesman problem," in *2013 Ninth International Conference on Natural Computation (ICNC)*, July 2013, pp. 397–401.

[24] B. A. Buran, S. H. Caglar, and O. K. Sahingoz, "A comparative study of smoothing a vehicle's trajectory which is calculated by an evolutionary algorithm," *International Journal of Computer Science and Information Security*, vol. 14, no. 6, p. 491, 2016.

[25] D. G. Macharet, A. A. Neto, and M. F. M. Campos, "Feasible uav path planning using genetic algorithms and bézier curves," in *Advances in Artificial Intelligence – SBIA 2010*, A. C. da Rocha Costa, R. M. Vicari, and F. Tonidandel, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 223–232.

[26] G. Vanegas, F. Samaniego, V. Girbes, L. Armesto, and S. Garcia-Nieto, "Smooth 3d path planning for non-holonomic uavs," in *2018 7th International Conference on Systems and Control (ICSC)*, Oct 2018, pp. 1–6.

[27] K. E. May, H. J. Sien, Y. S. Ping, and S. Z. Hai, "An evolutionary algorithm for multiple waypoints planning with b-spline trajectory generation for unmanned aerial vehicles (uavs)," in *International Conference on Computational Problem-Solving*, Dec 2010, pp. 77–81.

[28] I. K. Nikolos, N. C. Tsourveloudis, and K. P. Valavanis, *Evolutionary Algorithm Based Path Planning for Multiple UAV Cooperation.* Dordrecht: Springer Netherlands, 2007, pp. 309–340.

[29] X. Song and S. Hu, "2d path planning with dubins-path-based A^* algorithm for a fixed-wing uav," in *2017 3rd IEEE International Conference on Control Science and Systems Engineering (ICCSSE)*, Aug 2017, pp. 69–73.

[30] A. Ismail, E. Tuyishimire, and A. Bagula, "Generating dubins path for fixed wing uavs in search missions," in *Ubiquitous Networking*, N. Boudriga,

M.-S. Alouini, S. Rekhis, E. Sabir, and S. Pollin, Eds. Cham: Springer International Publishing, 2018, pp. 347–358.

[31] S. Subchan, B. White, A. Tsourdos, M. Shanmugavel, and R. Żbikowski, "Dubins path planning of multiple uavs for tracking contaminant cloud," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 5718 – 5723, 2008, 17th IFAC World Congress. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667016398573

[32] N. King. (2015) Fixed wing versus rotary wing for uav mapping applications. [Online]. Available: https://www.questuav.com/media/case-study/fixed-wing-versus-rotary-wing-for-uav-mapping-applications/

[33] S. Akyürek, M. Yılmaz, and M. Taşkıran, "İnsansız hava araçları muharebe alanında ve terörle mücadelede devrimsel dönüşüm," *Bilge Adamlar Stratejik Araştırmalar Merkezi Yayınları, İstanbul*, vol. 2, 2012.

[34] T. C. S. S. Başkanlığı. (2017) Bayraktar silahlı İnsansız hava aracı. [Online]. Available: https://www.ssb.gov.tr/WebSite/contentlist.aspx?PageID=365&LangID=1

[35] Admin. (2017) What is "bank angle" of a drone? [Online]. Available: http://www.roboticmagazine.com/drones/bank-angle-drone

[36] J. N. Ostler, W. J. Bowman, D. O. Snyder, and T. W. McLain, "Performance flight testing of small, electric powered unmanned aerial vehicles," *International Journal of Micro Air Vehicles*, vol. 1, no. 3, pp. 155–171, 2009. [Online]. Available: https://doi.org/10.1260/175682909789996177

[37] H. Grankvist, *Autopilot Design and Path Planning for a UAV*. Defence and Security, Systems and Technology, Swedish Defence Research . . . , 01 2006, p. 21.

[38] M. Di Luca, S. Mintchev, G. Heitz, F. Noca, and D. Floreano, "Bioinspired morphing wings for extended flight envelope and roll control of small drones," *Interface focus*, vol. 7, no. 1, p. 20160092, 2017.

[39] A. Ahmadzadeh, A. Jadbabaie, V. Kumar, and G. J. Pappas, "Multi-uav cooperative surveillance with spatio-temporal specifications," in *Proceedings of the 45th IEEE Conference on Decision and Control*, Dec 2006, pp. 5293–5298.

[40] A. Ryan and J. K. Hedrick, "A mode-switching path planner for uav-assisted search and rescue," in *Proceedings of the 44th IEEE Conference on Decision and Control*, Dec 2005, pp. 1471–1476.

[41] E. Semsch, M. Jakob, D. Pavlicek, and M. Pechoucek, "Autonomous uav surveillance in complex urban environments," in *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, vol. 2, Sep. 2009, pp. 82–85.

[42] P. B. Sujit, J. M. George, and R. W. Beard, "Multiple uav coalition formation," in *2008 American Control Conference*, June 2008, pp. 2010–2015.

[43] X. Zhang, X. Lu, S. Jia, and X. Li, "A novel phase angle-encoded fruit fly optimization algorithm with mutation adaptation mechanism applied to uav path planning," *Applied Soft Computing*, vol. 70, pp. 371 – 388, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S156849461830303X

[44] G.-G. Wang, H. E. Chu, and S. Mirjalili, "Three-dimensional path planning for ucav using an improved bat algorithm," *Aerospace Science and Technology*, vol. 49, pp. 231 – 238, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1270963815003843

[45] R. Cimurs, J. Hwang, and I. H. Suh, "Bezier curve-based smoothing for path planner with curvature constraint," in *2017 First IEEE International Conference on Robotic Computing (IRC)*, April 2017, pp. 241–248.

[46] M. Elhoseny, A. Tharwat, and A. E. Hassanien, "Bezier curve based path planning in a dynamic field using modified genetic algorithm," *Journal of*

Computational Science, vol. 25, pp. 339 – 350, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877750317308906

[47] W. Wu, X. Wang, and N. Cui, "Fast and coupled solution for cooperative mission planning of multiple heterogeneous unmanned aerial vehicles," *Aerospace Science and Technology*, vol. 79, pp. 131 – 144, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1270963817306934

[48] M. Xu, L. Dou, B. Xin, Y. Wang, H. Fang, and T. Cai, "Curvature-constrained uav path planning in tracking a moving air target," in *2018 IEEE 14th International Conference on Control and Automation (ICCA)*, June 2018, pp. 582–587.

[49] C. YongBo, M. YueSong, Y. JianQiao, S. XiaoLong, and X. Nuo, "Three-dimensional unmanned aerial vehicle path planning using modified wolf pack search algorithm," *Neurocomputing*, vol. 266, pp. 445 – 457, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231217309220

[50] W. Yu, W. Shuo, W. Rui, and T. Min, "Generation of temporal–spatial bezier curve for simultaneous arrival of multiple unmanned vehicles," *Information Sciences*, vol. 418-419, pp. 34 – 45, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S002002551630723X

[51] X. Pan, X. Wu, and X. Hou, "Research on global path planning for autonomous underwater vehicle considering ocean current," in *2018 2nd IEEE Advanced Information Management,Communicates,Electronic and Automation Control Conference (IMCEC)*, May 2018, pp. 790–793.

[52] B. Y. Q. J. J. Liang, H. Song and Z. F. Liu, "Comparison of three different curves used in path planning problems based on particle swarm optimizer," *Mathematical Problems in Engineering*, vol. 2014, p. 15 pages, 2014.

[53] G. Kavuran, "Path planning for nonholonomic mobile robot based on bézier curve," in *2017 International Artificial Intelligence and Data Processing Symposium (IDAP)*, Sep. 2017, pp. 1–5.

[54] D. Yulong, X. Bin, C. Jie, F. Hao, Z. Yangguang, G. Guanqiang, and D. Lihua, "Path planning of messenger uav in air-ground coordination," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 8045 – 8051, 2017, 20th IFAC World Congress. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405896317317366

[55] P. Váňa, J. Faigl, J. Sláma, and R. Pěnička, "Data collection planning with dubins airplane model and limited travel budget," in *2017 European Conference on Mobile Robots (ECMR)*, Sep. 2017, pp. 1–6.

[56] Z. Ming, Z. Lingling, S. Xiaohong, M. Peijun, and Z. Yanhang, "A cooperative path planning and smoothing algorithm for uavs in three dimensional environment," in *2014 Fourth International Conference on Instrumentation and Measurement, Computer, Communication and Control*, Sep. 2014, pp. 274–278.

[57] H. D. H. W. L. L. Gaige Wang, Lihong Guo and M. Shao, "A hybrid metaheuristic de/cs algorithm for ucav three-dimension path planning," *The Scientific World Journal*, vol. 2012, p. 11 pages, 2012.

[58] Y. Chen, Y. Zhao, and H. Wang, "Real time path planning for uav based on focused d," in *The Fourth International Workshop on Advanced Computational Intelligence*, Oct 2011, pp. 80–85.

[59] K. Wu, T. Xi, and H. Wang, "Real-time three-dimensional smooth path planning for unmanned aerial vehicles in completely unknown cluttered environments," in *TENCON 2017 - 2017 IEEE Region 10 Conference*, Nov 2017, pp. 2017–2022.

[60] K. R. Simba, N. Uchiyama, and S. Sano, "Real-time smooth trajectory generation for nonholonomic mobile robots using bézier curves," *Robotics and Computer-Integrated*

Manufacturing, vol. 41, pp. 31 – 42, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0736584516300552

[61] J. Keller, D. Thakur, J. Gallier, and V. Kumar, "Obstacle avoidance and path intersection validation for uas: A b-spline approach," in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2016, pp. 420–429.

[62] D. G. Macharet, J. W. G. Monteiro, G. R. Mateus, and M. F. M. Campos, "Time-optimized routing problem for vehicles with bounded curvature," in *2016 XIII Latin American Robotics Symposium and IV Brazilian Robotics Symposium (LARS/SBR)*, Oct 2016, pp. 145–150.

[63] T. Li, J. Jiang, Z. Zhen, and C. Gao, "Mission planning for multiple uavs based on ant colony optimization and improved dubins path," in *2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC)*, Aug 2016, pp. 954–959.

[64] C. Gao, Z. Zhen, and H. Gong, "A self-organized search and attack algorithm for multiple unmanned aerial vehicles," *Aerospace Science and Technology*, vol. 54, pp. 229 – 240, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1270963816301158

[65] C. Chen, Y. He, C. Bu, J. Han, and X. Zhang, "Quartic bézier curve based trajectory generation for autonomous vehicles with curvature and velocity constraints," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 6108–6113.

[66] C. Darwin, *On the Origin of Species, 1859*. London: Routledge, 2003.

[67] K. De Jong, D. Fogel, and H.-P. Schwefel, *A history of evolutionary computation*. IOP Publishing Ltd., 01 1997, pp. A2.3:1–12.

[68] M. Mitchell, *An Introduction to Genetic Algorithms*, ser. A Bradford book. Bradford Books, 1998. [Online]. Available: https://books.google.com.tr/books?id=0eznlz0TF-IC

[69] H. Pohlheim, "Genetic and Evolutionary Algorithm Toolbox for Matlab," in *Evolutionäre Algorithmen: Verfahren, Operatoren und Hinweise für die Praxis*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 157–170.

[70] J.-Z. Li, W.-X. Liu, Y. Han, H.-W. Xing, A.-M. Yang, and Y.-H. Pan, "Tsp problem based on artificial ant colony algorithm," in *Lecture Notes in Real-Time Intelligent Systems*, J. Mizera-Pietraszko and P. Pichappan, Eds. Cham: Springer International Publishing, 2018, pp. 196–202.

[71] Y. Wang, "Improving artificial bee colony and particle swarm optimization to solve tsp problem," in *2018 International Conference on Virtual Reality and Intelligent Systems (ICVRIS)*, Aug 2018, pp. 179–182.

[72] H. Wei, Z. Hao, H. Huang, G. Li, and Q. Chen, "A real adjacency matrix-coded differential evolution algorithm for traveling salesman problems," in *Bio-inspired Computing – Theories and Applications*, M. Gong, L. Pan, T. Song, and G. Zhang, Eds. Singapore: Springer Singapore, 2016, pp. 135–140.

[73] W. Xueyuan, "Research on solution of tsp based on improved genetic algorithm," in *2018 International Conference on Engineering Simulation and Intelligent Control (ESAIC)*, Aug 2018, pp. 78–82.

[74] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: An autocatalytic optimizing process," *Technical Report 91-016*, 1991.

[75] A. Colorni, M. Dorigo, V. Maniezzo *et al.*, "Distributed optimization by ant colonies," in *Proceedings of the first European conference on artificial life*, vol. 142. Paris, France, 1991, pp. 134–142.

[76] V. Maniezzo and A. Carbonaro, *Ant Colony Optimization: An Overview*. Boston, MA: Springer US, 2002, pp. 469–492.

[77] A. Uğur and D. Aydin, "An interactive simulation and analysis software for solving tsp using ant colony optimization algorithms," *Advances in Engineering Software*, vol. 40, no. 5, pp. 341 – 349, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0965997808001099

[78] M. Mavrovouniotis, F. M. Müller, and S. Yang, "Ant colony optimization with local search for dynamic traveling salesman problems," *IEEE Transactions on Cybernetics*, vol. 47, no. 7, pp. 1743–1756, July 2017.

[79] P. Li, H. Wang, and X. Li, "Improved ant colony algorithm for global path planning," in *AIP Conference Proceedings*, vol. 1820, no. 1. AIP Publishing, 2017, p. 080013.

[80] J. Gupta and C. Diwaker, "Vehicle routing problem using swarm optimization techniques." *International Journal of Advanced Research in Computer Science*, vol. 8, no. 5, 2017.

[81] W. Zhang, X. Gong, G. Han, and Y. Zhao, "An improved ant colony algorithm for path planning in one scenic area with many spots," *IEEE Access*, vol. 5, pp. 13 260–13 269, 2017.

[82] J. Liu, J. Yang, H. Liu, X. Tian, and M. Gao, "An improved ant colony algorithm for robot path planning," *Soft Computing*, vol. 21, no. 19, pp. 5829–5839, Oct 2017.

[83] N. Qingbin, "Research on robot obstacle avoidance and path tracking under dynamically unknown environment," in *2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, March 2017, pp. 2607–2610.

[84] Y. Wang, J. Ma, and Y. Wang, "Research on the ant colony algorithm in robot path planning," *AIP Conference Proceedings*, vol. 1839, no. 1, p. 020189, 2017. [Online]. Available: https://aip.scitation.org/doi/abs/10.1063/1.4982554

[85] L. Wang, C. Luo, M. Li, and J. Cai, "Trajectory planning of an autonomous mobile robot by evolving ant colony system," *International Journal of Robotics and Automation*, vol. 32, 08 2017.

[86] X. Wang, L. Xue, Y. Yan, and X. Gu, "Welding robot collision-free path optimization," *Applied Sciences*, vol. 7, no. 2, 2017. [Online]. Available: http://www.mdpi.com/2076-3417/7/2/89

[87] M. R. Jabbarpour, H. Zarrabi, J. J. Jung, and P. Kim, "A green ant-based method for path planning of unmanned ground vehicles," *IEEE Access*, vol. 5, pp. 1820–1832, 2017.

[88] Z. Li, "Multi-uav coordinate communication path planning based on grid map and ant-algorithm," in *2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, March 2017, pp. 1584–1587.

[89] Y. Zhou, F. He, N. Hou, and Y. Qiu, "Parallel ant colony optimization on multi-core simd cpus," *Future Generation Computer Systems*, vol. 79, pp. 473 – 487, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X16304289

[90] Y. Yan, H. suk Sohn, and G. Reyes, "A modified ant system to achieve better balance between intensification and diversification for the traveling salesman problem," *Applied Soft Computing*, vol. 60, pp. 256 – 267, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494617303927

[91] A. Amuthan and K. D. Thilak, "Improved ant colony algorithms for eliminating stagnation and local optimum problem — a survey," in *2017 International Conference on Technical Advancements in Computers and Communications (ICTACC)*, April 2017, pp. 97–101.

[92] H. Eldem and E. Ülker, "The application of ant colony optimization in the solution of 3d traveling salesman problem on a sphere," *Engineering Science and Technology, an International Journal*, vol. 20, no. 4, pp. 1242 – 1248, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2215098617309783

[93] J. Liu, S. Xu, F. Zhang, and L. Wang, "A hybrid genetic-ant colony optimization algorithm for the optimal path selection," *Intelligent Automation & Soft Computing*, vol. 23, no. 2, pp. 235–242, 2017. [Online]. Available: https://doi.org/10.1080/10798587.2016.1196926

[94] S. Maity, A. Roy, and M. Maiti, "An intelligent hybrid algorithm for 4- dimensional tsp," *Journal of Industrial Information Integration*, vol. 5, pp. 39 – 50, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2452414X16300164

[95] T. Keskintürk, "Diferansiyel gelişim algoritması," *İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi*, vol. 5, no. 9, pp. 85–99, 2006.

[96] R. Storn and K. Price, "Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces [r]," *Berkeley: ICSI*, 1995.

[97] D. Karaboğa and S. Ökdem, "A simple and global optimization algorithm for engineering problems: differential evolution algorithm," *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 12, no. 1, pp. 53–60, 2004.

[98] P. Shiakolas, D. Koladiya, and J. Kebrle, "On the optimum synthesis of six-bar linkages using differential evolution and the geometric centroid of precision positions technique," *Mechanism and Machine Theory*, vol. 40, no. 3, pp. 319 – 335, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0094114X04001260

[99] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, Feb 2011.

[100] D. Adhikari, E. Kim, and H. Reza, "A fuzzy adaptive differential evolution for multi-objective 3d uav path optimization," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, June 2017, pp. 2258–2265.

[101] I. Chatterjee and M. Zhou, "Differential evolution algorithms under multi-population strategy," in *2017 26th Wireless and Optical Communication Conference (WOCC)*, April 2017, pp. 1–7.

[102] Z. Ming, Z. Lingling, S. Xiaohong, M. Peijun, and Z. Yanhang, "Improved discrete mapping differential evolution for multi-unmanned aerial vehicles

cooperative multi-targets assignment under unified model," *International Journal of Machine Learning and Cybernetics*, vol. 8, no. 3, pp. 765–780, Jun 2017. [Online]. Available: https://doi.org/10.1007/s13042-015-0364-3

[103] S. MahmoudZadeh, A. Yazdani, K. Sammut, and D. Powers, "Online path planning for auv rendezvous in dynamic cluttered undersea environment using evolutionary algorithms," *Applied Soft Computing*, vol. 70, pp. 929 – 945, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494617306300

[104] S. S. Jadon, R. Tiwari, H. Sharma, and J. C. Bansal, "Hybrid artificial bee colony algorithm with differential evolution," *Applied Soft Computing*, vol. 58, pp. 11 – 24, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494617301904

[105] R. Chai, A. Savvaris, A. Tsourdos, and S. Chai, "Multi-objective trajectory optimization of space manoeuvre vehicle using adaptive differential evolution and modified game theory," *Acta Astronautica*, vol. 136, pp. 273 – 280, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0094576516305872

[106] D. Jagodziński and J. Arabas, "A differential evolution strategy," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, June 2017, pp. 1872–1876.

[107] M. Leon and N. Xiong, "Alopex-based mutation strategy in differential evolution," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, June 2017, pp. 1978–1984.

[108] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, Nov 1995, pp. 1942–1948 vol.4.

[109] R. C. Eberhart and Y. Shi, "Comparison between genetic algorithms and particle swarm optimization," in *Evolutionary Programming VII*, V. W.

Porto, N. Saravanan, D. Waagen, and A. E. Eiben, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 611–616.

[110] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm Intelligence*, vol. 1, no. 1, pp. 33–57, Jun 2007. [Online]. Available: https://doi.org/10.1007/s11721-007-0002-0

[111] X.-F. Xie, W.-J. Zhang, and Z.-L. Yang, "Dissipative particle swarm optimization," in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*, vol. 2, May 2002, pp. 1456–1461 vol.2.

[112] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, May 1998, pp. 69–73.

[113] Q. Bai, "Analysis of particle swarm optimization algorithm," *Computer and Information Science*, vol. 3, no. 1, p. 180, May 2010.

[114] R. Liu, J. Li, J. fan, C. Mu, and L. Jiao, "A coevolutionary technique based on multi-swarm particle swarm optimization for dynamic multi-objective optimization," *European Journal of Operational Research*, vol. 261, no. 3, pp. 1028 – 1051, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377221717302709

[115] Y. Zhou, Z. Li, and C. Hu, "A differential bpso-ga hybrid algorithm for discrete combination optimization," in *2017 36th Chinese Control Conference (CCC)*, July 2017, pp. 2686–2690.

[116] Y. Marinakis, A. Migdalas, and A. Sifaleras, "A hybrid particle swarm optimization – variable neighborhood search algorithm for constrained shortest path problems," *European Journal of Operational Research*, vol. 261, no. 3, pp. 819 – 834, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377221717302357

[117] L. Blasi, S. Barbato, and E. D'Amato, "A mixed probabilistic–geometric strategy for uav optimum flight path identification based on bit-coded basic manoeuvres," *Aerospace Science and Technology*, vol. 71, pp. 1 – 11, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1270963817304662

[118] A. T. Hafez, M. A. Kamel, P. T. Jardin, and S. N. Givigi, "Task assignment/trajectory planning for unmanned vehicles via hflc and pso," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2017, pp. 554–559.

[119] A. Ayari and S. Bouamama, "A new multiple robot path planning algorithm: dynamic distributed particle swarm optimization," *Robotics and Biomimetics*, vol. 4, no. 1, p. 8, Nov 2017. [Online]. Available: https://doi.org/10.1186/s40638-017-0062-6

[120] A. Asma and B. Sadok, "Collision-free optimal paths for multiple robot systems using a new dynamic distributed particle swarm optimization algorithm," in *2017 18th International Conference on Advanced Robotics (ICAR)*, July 2017, pp. 493–497.

[121] ——, "Dynamic distributed pso joints elites in multiple robot path planning systems: theoretical and practical review of new ideas," *Procedia Computer Science*, vol. 112, pp. 1082 – 1091, 2017, knowledge-Based and Intelligent Information and Engineering Systems: Proceedings of the 21st International Conference, KES-20176-8 September 2017, Marseille, France. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050917314850

[122] Y. Wang, F. Cai, and Y. Wang, "Dynamic path planning for mobile robot based on particle swarm optimization," *AIP Conference Proceedings*, vol. 1864, no. 1, p. 020024, 2017. [Online]. Available: https://aip.scitation.org/doi/abs/10.1063/1.4992841

[123] M. D. Phung, C. H. Quach, T. H. Dinh, and Q. Ha, "Enhanced discrete particle swarm optimization path planning for uav vision-based surface inspection," *Automation in Construction*, vol. 81, pp. 25 – 33, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0926580517303825

[124] X. Luo, J. Wang, and X. Li, "Joint grid network and improved particle swarm optimization for path planning of mobile robot," in *2017 36th Chinese Control Conference (CCC)*, July 2017, pp. 8304–8309.

[125] H. Huang and T. Zhuo, "Multi-model cooperative task assignment and path planning of multiple ucav formation," *Multimedia Tools and Applications*, vol. 78, no. 1, pp. 415–436, Jan 2019. [Online]. Available: https://doi.org/10.1007/s11042-017-4956-7

[126] T. Xue, R. Li, M. Tokgo, J. Ri, and G. Han, "Trajectory planning for autonomous mobile robot using a hybrid improved qpso algorithm," *Soft Computing*, vol. 21, no. 9, pp. 2421–2437, May 2017. [Online]. Available: https://doi.org/10.1007/s00500-015-1956-2

[127] I. Dubey and M. Gupta, "Uniform mutation and spv rule based optimized pso algorithm for tsp problem," in *2017 4th International Conference on Electronics and Communication Systems (ICECS)*, Feb 2017, pp. 168–172.

[128] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Technical report-tr06, Erciyes university, engineering faculty, computer . . . , Tech. Rep., 2005.

[129] D. Karaboga and B. Gorkemli, "A combinatorial artificial bee colony algorithm for traveling salesman problem," in *2011 International Symposium on Innovations in Intelligent Systems and Applications*, June 2011, pp. 50–53.

[130] W. li Xiang, Y. zhen Li, R. chun He, M. xia Gao, and M. qing An, "A novel artificial bee colony algorithm based on the cosine similarity," *Computers*

& *Industrial Engineering*, vol. 115, pp. 54 – 68, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0360835217305077

[131] L. Cui, G. Li, X. Wang, Q. Lin, J. Chen, N. Lu, and J. Lu, "A ranking-based adaptive artificial bee colony algorithm for global numerical optimization," *Information Sciences*, vol. 417, pp. 169 – 185, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020025516307289

[132] X. Li, H. Yang, M. Yang, X. Yang, and G. Yang, "Accelerating artificial bee colony algorithm with neighborhood search," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, June 2017, pp. 1549–1556.

[133] Z. B. Özger, B. Bolat, and B. Diri, "Ibitabc: Improved binary artificial bee colony algorithm with local search," in *2017 International Conference on Computer Science and Engineering (UBMK)*, Oct 2017, pp. 165–170.

[134] M. Luqman, M. Saeed, J. Ali, and M. F. Tabassum, "Radial artificial bee colony algorithm for constrained engineering problems," *Pakistan journal of science*, vol. 69, pp. 127–135, 03 2017.

[135] T. K. Sharma and M. Pant, "Distribution in the placement of food in artificial bee colony based on changing factor," *International Journal of System Assurance Engineering and Management*, vol. 8, no. 1, pp. 159–172, Mar 2017. [Online]. Available: https://doi.org/10.1007/s13198-016-0495-2

[136] S. Mingprasert and R. Masuchun, "Adaptive artificial bee colony algorithm for solving the capacitated vehicle routing problem," in *2017 9th International Conference on Knowledge and Smart Technology (KST)*, Feb 2017, pp. 23–27.

[137] S. S. Jadon, R. Tiwari, H. Sharma, and J. C. Bansal, "Hybrid artificial bee colony algorithm with differential evolution," *Applied Soft Computing*, vol. 58, pp. 11 – 24, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494617301904

[138] Y. Zhong, J. Lin, L. Wang, and H. Zhang, "Hybrid discrete artificial bee colony algorithm with threshold acceptance

criterion for traveling salesman problem," *Information Sciences*, vol. 421, pp. 70 – 84, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020025517302748

[139] J. H. Holland, "Outline for a logical theory of adaptive systems," *J. ACM*, vol. 9, no. 3, pp. 297–314, Jul. 1962. [Online]. Available: http://ezproxy.iku.edu.tr:2344/10.1145/321127.321128

[140] H. John, "Holland. 1975. adaptation in natural and artificial systems," *Ann Arbor: University of Michigan Press*, 1975.

[141] K. De Jong, "Learning with genetic algorithms: An overview," *Machine Learning*, vol. 3, no. 2, pp. 121–138, Oct 1988. [Online]. Available: https://doi.org/10.1007/BF00113894

[142] J. R. Koza, *Genetic programming: on the programming of computers by means of natural selection*. MIT press, 1992, vol. 1.

[143] M. Mitchell, *An Introduction to Genetic Algorithms*. MIT press, 1998.

[144] Z. Michalewicz, *GAs: What Are They?* Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 13–30.

[145] N. M. Razali, J. Geraghty *et al.*, "Genetic algorithm performance with different selection strategies in solving tsp," in *Proceedings of the world congress on engineering*, vol. 2, no. 1. International Association of Engineers Hong Kong, 2011, pp. 1–6.

[146] R. L. Haupt and S. E. Haupt, *Practical genetic algorithms*. John Wiley & Sons, 2004.

[147] P. A. Diaz-Gomez and D. Hougen, "Initial population for genetic algorithms : A metric approach." in *GEM*, 01 2007, pp. 43–49.

[148] M. pulat and I. Deveci Kocakoç, "Gezgin satıcı probleminin çözümünde kullanılan genetik algoritmanın parametrelerinin incelenmesi," *Uluslararası İktisadi ve İdari İncelemeler Dergisi*, pp. 21–36, 09 2017.

[149] W. Ellili, M. Samet, and A. Kachouri, "Traveling salesman problem of optimization based on genetic algorithms," in *2017 International Conference on Smart, Monitored and Controlled Cities (SM2C)*, Feb 2017, pp. 123–127.

[150] A. Mitra, B. Sarkar, and S. Bhattacharyya, "A combined genetic algorithm for symmetric travelling salesman problem," *International Journal of Advanced Research in Computer Science*, vol. 8, no. 5, 2017.

[151] V. RAMAN and N. SINGH GILL, "A swarmed ga algorithm for solving travelling salesman problem." *Journal of Theoretical & Applied Information Technology*, vol. 95, no. 16, 2017.

[152] M. Elhoseny, A. Tharwat, and A. E. Hassanien, "Bezier curve based path planning in a dynamic field using modified genetic algorithm," *Journal of Computational Science*, vol. 25, pp. 339 – 350, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877750317308906

[153] A. Hussain, Y. S. Muhammad, M. Nauman Sajid, I. Hussain, A. Mohamd Shoukry, and S. Gani, "Genetic algorithm for traveling salesman problem with modified cycle crossover operator," *Computational intelligence and neuroscience*, vol. 2017, p. 7 pages, 2017.

[154] A. V. Eremeev and Y. V. Kovalenko, "Genetic algorithm with optimal recombination for the asymmetric travelling salesman problem," in *Large-Scale Scientific Computing*, I. Lirkov and S. Margenov, Eds. Cham: Springer International Publishing, 2018, pp. 341–349.

[155] C. Tajani, A. Otman, and J. Abouchabaka, "Optimization approach based on immigration strategies for symmetric traveling salesman problem," *International Journal of Open Problems in Computer Science and Mathematics*, vol. 10, pp. 38–53, 07 2017.

[156] Xu, Mingji, Li, Sheng, and Guo, Jian, "Optimization of multiple traveling salesman problem based on simulated annealing genetic algorithm,"

MATEC Web Conf., vol. 100, p. 8 pages, 2017. [Online]. Available: https://doi.org/10.1051/matecconf/201710002025

[157] H. Chai, R. He, C. Ma, C. Dai, and K. Zhou, "Path planning and vehicle scheduling optimization for logistic distribution of hazardous materials in full container load," *Discrete Dynamics in Nature and Society*, vol. 2017, p. 13 pages, 2017.

[158] M. A. Mohammed, M. K. A. Ghani, R. I. Hamed, S. A. Mostafa, M. S. Ahmad, and D. A. Ibrahim, "Solving vehicle routing problem by using improved genetic algorithm for optimal solution," *Journal of Computational Science*, vol. 21, pp. 255 – 262, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877750317303848

[159] C.-C. Shih, M.-F. Horng, T.-S. Pan, J.-S. Pan, and C.-Y. Chen, "A genetic-based effective approach to path-planning of autonomous underwater glider with upstream-current avoidance in variable oceans," *Soft Computing*, vol. 21, no. 18, pp. 5369–5386, Sep 2017. [Online]. Available: https://doi.org/10.1007/s00500-016-2122-1

[160] W. Cai, M. Zhang, and Y. Zheng, "Task assignment and path planning for multiple autonomous underwater vehicles using 3d dubins curves," *Sensors*, vol. 17, no. 7, p. 1607, 2017.

[161] Y. Cao, W. Wei, Y. Bai, and H. Qiao, "Multi-base multi-uav cooperative reconnaissance path planning with genetic algorithm," *Cluster Computing*, Aug 2017. [Online]. Available: https://doi.org/10.1007/s10586-017-1132-9

[162] J. Li, Y. Huang, Z. Xu, J. Wang, and M. Chen, "Path planning of uav based on hierarchical genetic algorithm with optimized search region," in *2017 13th IEEE International Conference on Control Automation (ICCA)*, July 2017, pp. 1033–1038.

[163] C. Ramirez-Atencia, G. Bello-Orgaz, M. D. R-Moreno, and D. Camacho, "Solving complex multi-uav mission planning problems using multi-objective genetic algorithms," *Soft Computing*, vol. 21, no. 17, pp.

4883–4900, Sep 2017. [Online]. Available: https://doi.org/10.1007/s00500-016-2376-7

[164] M. Peker, "A fully customizable hardware implementation for general purpose genetic algorithms," *Applied Soft Computing*, vol. 62, pp. 1066 – 1076, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494617305835

[165] C.-C. Li, C.-H. Lin, and J.-C. Liu, "Parallel genetic algorithms on the graphics processing units using island model and simulated annealing," *Advances in Mechanical Engineering*, vol. 9, no. 7, pp. 1–14, 2017.

[166] Y. Zhou, Z. Li, and C. Hu, "A differential bpso-ga hybrid algorithm for discrete combination optimization," in *2017 36th Chinese Control Conference (CCC)*, July 2017, pp. 2686–2690.

[167] C. Changdar, R. K. Pal, and G. S. Mahapatra, "A genetic ant colony optimization based algorithm for solid multiple travelling salesmen problem in fuzzy rough environment," *Soft Computing*, vol. 21, no. 16, pp. 4661–4675, Aug 2017. [Online]. Available: https://doi.org/10.1007/s00500-016-2075-4

[168] M. M. Alipour, S. N. Razavi, M. R. Feizi Derakhshi, and M. A. Balafar, "A hybrid algorithm using a genetic algorithm and multiagent reinforcement learning heuristic to solve the traveling salesman problem," *Neural Computing and Applications*, vol. 30, no. 9, pp. 2935–2951, Nov 2018. [Online]. Available: https://doi.org/10.1007/s00521-017-2880-4

[169] M. Ma and H. Li, "A hybrid genetic algorithm for solving bi-objective traveling salesman problems," in *Journal of Physics: Conference Series*, vol. 887, no. 1. IOP Publishing, 2017, p. 012065.

[170] I. Khan and M. K. Maiti, "A novel hybrid algorithm for generalized traveling salesman problems in different environments," *Vietnam Journal of Computer Science*, vol. 5, no. 1, pp. 27–43, Feb 2018. [Online]. Available: https://doi.org/10.1007/s40595-017-0099-z

[171] I. İlhan, "An application on mobile devices with android and ios operating systems using google maps apis for the traveling salesman problem," *Appl. Artif. Intell.*, vol. 31, no. 4, pp. 332–345, Apr. 2017.

[172] S. Maity, A. Roy, and M. Maiti, "An intelligent hybrid algorithm for 4- dimensional tsp," *Journal of Industrial Information Integration*, vol. 5, pp. 39 – 50, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2452414X16300164

[173] D. Gaifang, F. Xueliang, L. Honghui, and X. Pengfei, "Cooperative ant colony-genetic algorithm based on spark," *Computers & Electrical Engineering*, vol. 60, pp. 66 – 75, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0045790616304359

[174] R. Singh, "Genetic-variable neighborhood search with thread replication for mobile cloud computing," *Int. J. Parallel Emerg. Distrib. Syst.*, vol. 32, no. 5, pp. 486–501, Sep. 2017.

[175] Y. Bouzid, Y. Bestaoui, and H. Siguerdidjane, "Two-scale geometric path planning of quadrotor with obstacle avoidance: First step toward coverage algorithm," in *2017 11th International Workshop on Robot Motion and Control (RoMoCo)*, July 2017, pp. 166–171.

[176] A. Mazidi and E. Damghanijazi, "Meta-heuristic approaches for solving travelling salesman problem," *International Journal of Advanced Research in Computer Science*, vol. 5, 03 2017.

[177] M. E. Mortenson, *Geometric Modeling (2Nd Ed.)*. New York, NY, USA: John Wiley & Sons, Inc., 1997.

[178] F. Zhou, B. Song, and G. Tian, "Bézier curve based smooth path planning for mobile robot," *Journal of Information and Computational Science*, vol. 8, pp. 2441–2450, 12 2011.

[179] H. Oruç and G. M. Phillips, "q-bernstein polynomials and bézier curves," *Journal of Computational and Applied Mathematics*,

vol. 151, no. 1, pp. 1 – 12, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377042702007331

[180] J.-w. Choi and G. H. Elkaim, "Bézier curves for trajectory guidance," in *World Congress on Engineering and Computer Science, WCECS*. Citeseer, 2008, pp. 22–24.

[181] L.-Z. LU and Y.-Y. QIU, "Explicit g2-constrained merging of a pair of bézier curves by control point optimization," *Acta Automatica Sinica*, vol. 40, no. 7, pp. 1505 – 1509, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1874102914600161

[182] R. Lee and Y. J. Ahn, "Limit curve of h-bézier curves and rational bézier curves in standard form with the same weight," *Journal of Computational and Applied Mathematics*, vol. 281, pp. 1 – 9, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377042714005494

[183] M. G. Cox, "The numerical evaluation of b-splines," *IMA Journal of Applied Mathematics*, vol. 10, no. 2, pp. 134–149, 1972.

[184] C. De Boor, "On calculating with b-splines," *Journal of Approximation theory*, vol. 6, no. 1, pp. 50–62, 1972.

[185] N. M. Patrikalakis, T. Maekawa, and W. Cho. (2009) Shape İnterrogation for computer aided design and manufacturing. [Online]. Available: http://web.mit.edu/hyperbook/Patrikalakis-Maekawa-Cho/

[186] S. Mittal and K. Deb, "Three-dimensional offline path planning for uavs using multiobjective evolutionary algorithms," in *2007 IEEE Congress on Evolutionary Computation*, Sep. 2007, pp. 3195–3202.

[187] A. Chawla. (2009) Lecture - 33 modelling of b-spline curves. Youtube. [Online]. Available: https://www.youtube.com/watch?v=wAHaJ9ASPo8

[188] I. C. James and P. Shawn. (2004) B-spline curves. [Online]. Available: https://tr.scribd.com/document/183525651/b-splines-04-pdf

[189] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *American Journal of Mathematics*, vol. 79, no. 3, pp. 497–516, 1957. [Online]. Available: http://www.jstor.org/stable/2372560

[190] J. Le Ny, E. Frazzoli, and E. Feron, "The curvature-constrained traveling salesman problem for high point densities," in *2007 46th IEEE Conference on Decision and Control*, Dec 2007, pp. 5985–5990.

[191] P. R. Giordano and M. Vendittelli, "Shortest paths to obstacles for a polygonal dubins car," *IEEE Transactions on Robotics*, vol. 25, no. 5, pp. 1184–1191, Oct 2009.

[192] S. Subchan, B. White, A. Tsourdos, M. Shanmugavel, and R. Żbikowski, "Dubins path planning of multiple uavs for tracking contaminant cloud," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 5718 – 5723, 2008, 17th IFAC World Congress. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667016398573

[193] G. Cho and J. Ryeu, *An Efficient Method to Find a Shortest Path for a Car-Like Robot*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1000–1011. [Online]. Available: https://*doi.org*/10.1007/978 − 3 − 540 − 37256 − $1_1$31

[194] W. Cai and M. Zhang, "3d dubins curves based path programming for mobile sink in underwater sensor networks," *Electronics Letters*, vol. 53, no. 1, pp. 48–50, 2017.

[195] A. Giese. (2012) A comprehensive, step-by-step tutorial on computing dubin's curves. [Online]. Available: https://gieseanw.wordpress.com/2012/10/21/a-comprehensive-step-by-step-tutorial-to-computing-dubins-paths/

[196] S. Hota and D. Ghose, "A modified dubins method for optimal path planning of a miniature air vehicle converging to a straight line path," in *2009 American Control Conference*, June 2009, pp. 2397–2402.

ÖZGEÇMİŞ

Bayram Ali BURAN, 30 Haziran 1984 Kocasinan/Kayseri doğumludur. 1998 yılında Kadı Burhanettin Orta Okulundan mezun olmuş, aynı yıl başladığı Mimar Sinan Anadolu Öğretmen Lisesinden 2002 yılında mezun olmuştur. Yine 2002 yılında başladığı Karadeniz Teknik Üniversitesi Fatih Eğitim Fakültesi Orta Öğretim Matematik Öğretmenliği-Tezsiz Yüksek Lisans programından 2008 yılında mezun olmuştur. 2009 yılında Hava Harp Okulu'nda Öğretim Görevlisi olarak göreve başlamış, burada 8 yıl süreyle Calculus I-II ve Lineer Cebir dersleri vermiştir. 2012 yılı Temmuz ayında İstanbul Kültür Üniversitesi Matematik-Bilgisayar Ana Bilim Dalı Matematik programında doktora öğrenciliği başlamıştır. Halen özel bir firmada yurtdışı eğitim ve danışmanlık alanında görevine devam etmektedir. Evli ve iki çocuk babasıdır.