Karaelmas Fen ve Mühendislik Dergisi
Dergi web sayfası: http://fbd.beun.edu.tr

Araştırma Makalesi

İnsansız Hava Araçları İçin Yapay Arı Kolonisi Algoritması Kullanarak Rota Planlama

Route Planning using Artificial Bee Colony Algorithm for Unmanned Aerial Vehicles

Volkan Çavuş[1*], Adem Tuncer[2]

[1]Yalova Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Bölümü, Yalova, Türkiye
[2]Yalova Üniversitesi, Bilgisayar Mühendisliği, Yalova, Türkiye

Öz

İnsansız Hava Araçları (İHA) için rota planlama, başlangıç noktasından verilen hedef noktaya kadar engellerden ve tehlikeli bölgelerden sakınarak güvenli bir rotanın bulunmasıdır. İHA'ların rota planlama problemine çözüm bulmak için pek çok yöntem kullanılmaktadır. Bu çalışmada rota planlama için Yapay Arı Kolonisi (YAK) algoritması kullanılmıştır. YAK algoritması son yıllarda optimizasyon problemlerinde yaygın bir şekilde kullanılan sezgisel algoritmalardan bir tanesidir. Çalışmada, YAK algoritmasının benzetim uygulamalarını gerçekleştirmek amacı ile C# programlama dili kullanılarak kullanıcı etkileşimli bir arayüz tasarlanmıştır. Rota planlaması için yapılan deneysel çalışmalar, YAK algoritmasının uygun rotalar bulmada başarılı sonuçlar verdiğini göstermektedir.

Anahtar Kelimeler: İnsansız hava aracı, Rota planlama, Yapay arı kolonisi algoritması

Abstract

A route planning for Unmanned Aerial Vehicles (UAVs) is to find a feasible route from the starting point to the target point avoiding obstacles and dangerous areas. Many methods are used in order to solve route planning of UAVs. In this study, Artificial Bee Colony (ABC) algorithm is used for route planning. The ABC algorithm is one of the heuristic algorithms widely used in optimization problems in recent years. In the study, an interactive user interface is designed using C# programming language to implement the simulation of the ABC algorithm. Experimental studies for route planning show that the ABC algorithm provides successful results in finding suitable routes.

Keywords: Unmanned aerial vehicle, Route planning, Artificial bee colony algorithm

1. Giriş

Kendi güç sistemi olan, otomatik veya uzaktan kontrol sistemi ile uçurulan pilotsuz hava araçlarına İnsansız Hava Aracı (İHA) denilmektedir. İHA'lar "pilotsuz uçak", "robot uçak", "drones" gibi isimlerle adlandırılmaktadırlar (Akyürek vd. 2012). İHA'lar 20. yüzyılın başından beri aşırı dikkat gerektiren ve uzun süreli insan odaklanmasının azalacağı düşünülen tehlikeli ve riskli ortamlarda kullanılmaktadırlar (SSM, 2012). I. Dünya Savaşı'ndan bu yana özellikle askeri uygulamalar için kullanılan İHA'ların ilk yaygın kullanımı ABD tarafından Vietnam'daki savaş sırasında olmuştur (Ingham, 2008). Uzun yıllar istihbarat, keşif, gözetleme gibi askeri alanlarda kullanılan İHA'lar son yıllarda uzaktan algılama, veri toplama, arama-kurtarma, yangın söndürme, tarım uygulamaları, görüntüleme ve taşımacılık gibi sivil alanlarda da kendine uygulama alanı bulmuştur.

İHA'ların bahsedilen bu görevleri otonom olarak yerine getirebilmeleri için pek çok çalışma yapılmaktadır. Bu çalışmalardan önemli bir tanesi de rota planlamadır. Rota planlama İHA'nın bir başlangıç noktasından hedef noktaya kadar herhangi bir engele (dağ, tepe vb.) çarpmadan veya tehlikeli bir bölgeye (radar vb.) girmeden takip edebileceği en uygun güzergâhın bulunması ve İHA'nın bu güzergâh boyunca otonom olarak hedefe doğru ilerlemesidir. Otonom İHA'lar için rota planlama aktif olarak devam eden bir araştırma konusudur ve rota planlama problemine çözüm üretmek için farklı pek çok metot kullanılmaktadır. İHA'lar için rota planlama problemlerine çözüm çalışmalarında

*Sorumlu yazarın e-posta adresi: cvsvlkn@gmail.com

Geliş tarihi / Received : 14.11.2016
Kabul tarihi / Accepted : 30.12.2016

A-yıldız algoritması (Tulum 2009), tam sayılı programlama (Gencer vd 2009), grid tarama algoritması (Aydemir 2014) gibi algoritmalar kullanılmıştır. Her bir algoritmanın uygulama alanına bağlı olarak avantajı olduğu gibi yavaş hesaplama veya yerel minimum/maksimum noktalarına takılma gibi dezavantajları da olabilmektedir.

Geleneksel algoritmaların yanı sıra rota planlama çalışmaları için sezgisel algoritmaların da kullanıldığı pek çok çalışma bulunmaktadır. Sezgisel algoritmaların kullanılmasındaki en önemli unsurlardan bir tanesi, kısa zamanda iyi sonuçların elde edilmesidir. Geleneksel algoritmaların genellikle hesaplama maliyetleri yüksektir. Rota planlama işlemi en kısa yolun bulunması olarak düşünüldüğünde aynı zamanda bir optimizasyon problemi olarak ta ele alınabilmektedir. Sezgisel algoritma olan genetik algoritma (Özalp 2013), karınca kolonisi algoritması (Çekmez 2014), reaktif tabu arama (Ryan vd 1998)'da rota planlama problemlerine çözüm aramak için kullanılmaktadır.

Bu çalışmada, İHA'lar için rota planlama problemi YAK algoritması kullanılarak çözülmüştür. İHA'ların engellerden ve tehlikeli bölgelerden sakınarak güvenli bir rotayı takip edebilmelerine yönelik planlama yapılmıştır. Rota planlaması işlemi yapılırken gerçek harita bilgisi kullanılmış ve haritadaki coğrafi koşullar dikkate alınmıştır. Çalışmada YAK algoritmasının benzetim uygulamalarını görsel olarak gerçekleştirmek amacıyla kullanıcı arayüzü tasarlanmıştır.

2. Gereç ve Yöntem

2.1.Yapay Arı Kolonisi (YAK) Algoritması

Sezgisel algoritmalar karmaşık problemler için en makul bir süre içerisinde en iyiye yakın çözümler üretebilen algoritmalardır. Yakınsama özelliğine sahiptirler fakat sezgisel yapılarından dolayı kesin çözümü garanti edemezler (Karaboğa 2004). Sezgisel algoritmaların alt başlıklarından biri olan sürü zekâ algoritmaları, termit, karınca, kuş, balık ve arı sürüleri gibi aralarında etkileşimi olan böceklerin veya diğer sosyal hayvanların topluluk halinde davranışlarından esinlenerek geliştirilen algoritmalardır (Karaboğa ve Akay 2007). Sürü zekâ temelli algoritmalardan biri olan YAK algoritması da son yıllarda optimizasyon problemlerinde yaygın olarak kullanılmaktadır.

Çok boyutlu ve çok modelli optimizasyon problemlerini çözmek için arı sürülerinin yiyecek arama davranışlarının taklit edilerek modellenmesine dayanan YAK algoritması 2005 yılında Karaboğa tarafından önerilmişti (Karaboğa 2005). Arıların yiyecek aramak için gittikleri kaynaklar, algoritmada çözülmek istenen problemin olası çözümlerini, kaynaklardaki nektar miktarı ise çözümün kalitesini (amaç fonksiyon) ifade etmektedir. YAK algoritmasında, bir kolonide işçi arı, gözcü arı ve kâşif arı olmak üzere üç çeşit arı bulunmaktadır (Karaboğa ve Akay 2007). Her bir kaynak için görevli bir arı bulunmaktadır. YAK algoritması en fazla nektara sahip kaynağı bularak olası çözümler arasından problemin maksimum ya da minimum noktasını bulmaya çalışmaktadır (Akay 2009). YAK algoritmasının adımları aşağıda detaylı olarak verilmektedir;

2.2. Başlangıç Yiyecek Kaynaklarının Üretilmesi

Algoritmanın başlangıç adımı olarak, kâşif arılar yiyecek kaynaklarını rastgele aramaya başlarlar. Denklem (1) kullanılarak her bir parametrenin alt ve üst sınırları arasında rasgele başlangıç yiyecek kaynakları, x_{ij}, üretilir (Akay 2009). Başlangıç nüfusu olarak üretilen bu kaynaklar aday çözüm kümesini ifade etmektedir.

$$x_{ij} = x_j^{\min} + rand(0,1)(x_j^{\max} - x_j^{\min}) \quad (1)$$

Denklemdeki eşitlikte $i=1,2,\ldots,SN$, $j=1,2,\ldots,D$'dir. SN aday çözüm sayısını ve D ise optimize edilecek parametre sayısını ifade etmektedir. $rand(0,1)$ ile 0 ile 1 arasında rastgele sayı üretilmektedir. x^{max} ve x^{min}, j. parametrenin alt ve üst sınır değerleridir. Bir kaynağın nektar miktarının tükenip tükenmediği geliştirilememe sayacı ile kontrol edilmektedir ve bu kontrol parametresi "limit" olarak adlandırılmaktadır. Önceden tanımlanan limit sayısı algoritmada önemli bir kontrol parametresidir (Karaboğa ve Akay 2009). Başlangıç aşamasında geliştirilememe sayacı sıfırlanmaktadır. Algoritmada durdurma kriteri olarak genellikle maksimum çevrim sayısı (iterasyon) veya önceden tanımlanmış bir kriter kullanılmaktadır (Karaboğa ve Akay 2009). Kâşif arılar yiyecek kaynağı bulduktan sonra görevli arılara dönüşmektedir.

2.3. İşçi Arı Aşaması

İşçi arı, mevcut yiyecek kaynağının komşuluğunda yeni bir yiyecek kaynağı arar. Bulunan komşu yiyecek kaynağının kalitesini değerlendirir ve yeni kaynaktaki nektar miktarı eskisinden daha iyi ise bu kaynağı hafızasına alır ve eski kaynak bilgilerini siler. İşçi arının mevcut kaynağın komşuluğunda yeni bir yiyecek kaynağını bulması denklem (2) ile gerçekleştirilmektedir.

$$v_{ij}=x_{ij}+\varphi_{ij}(x_{ij}-x_{kj}) \quad (2)$$

Her bir x_i yiyecek kaynağı için [1,D] arasında rastgele seçilen j parametresi değiştirilerek oluşturulan v_i kaynağı bulunur (Akay 2009). φ_{ij}, [-1,1] arasında üretilen rastgele

bir sayıdır ve mevcut kaynağın komşuluğunda rastgele üretilen x_k kaynağı ile mevcut kaynağın farkı ile çarpılarak yeni bir yiyecek kaynağının bulunmasında rol oynamaktadır. Bulunan kaynağın kalitesi hesaplandıktan sonra kaynağa denklem (3) ile uygunluk değeri ($fitness_i$) atanır.

$$fitness_i = \begin{cases} \frac{1}{(1+f_i)}, fi \geq 0 \\ 1 + abs(f_i), fi < 0 \end{cases} \quad (3)$$

f_i değeri, komşu kaynak çözümün kalitesi yani amaç fonksiyon değeridir. Mevcut kaynak ile komşu kaynak arasında seçim işlemi yaparken uygunluk değerine göre açgözlü seçim yöntemi uygulanır. Bulunan v_i komşu çözümü mevcut x_i çözümünden daha kalitesi ise v_i çözümünü yeni kaynak olarak kullanır ve geliştirememe sayacı sıfırlanır, tersi durumda ise mevcut kaynak ile devam edilir ve sayaç bir arttırılır (Akay 2009).

2.4. Gözcü Arı Aşaması

Gözcü arılar tüm görevli arıların elde ettikleri kaynak bilgilerini toplarlar. Bu bilgiler ışığında gözcü arı kaynaktaki nektar miktarı ile orantılı olasılıkla bir kaynak seçer. Her bir kaynak için denklem (3) ile hesaplanan olasılık değeri, [0,1] arasında üretilen rastgele bir değerden büyük ise gözcü arılar denklem (2) kullanılarak yeni bir çözüm üretir ve çözümün kalitesi hesaplanır (Akay 2009). Mevcut çözüm ile yeni çözüm açgözlü seçim yöntemi ile karşılaştırılır.

$$p_i = \frac{fitness_i}{\sum_{i=1}^{SN} fitnes_i} \quad (4)$$

p_i *i.* çözümün olasılık değeri, $fitness_i$ ise *i.* çözümün uygunluk değeridir.

2.5. Kâşif Arı Aşaması

Yiyecek kaynağı, önceden belirlenmiş olan geliştirilememe sayacı "*limit*" eşik değerine ulaşmış ise, bu yiyecek kaynağı tükenmiş olarak kabul edilir ve bu noktadan sonra bu kaynaktaki görevli arı kâşif arıya dönüşür. Kâşif arı kaynağı tükenen x_i kaynağı yerine denklem (1) kullanılarak rastgele yeni bir yiyecek kaynağı üretir (Akay 2009). Algoritmadaki bu aşamalar önceden tanımlanan bir kriter veya maksimum çevrim sayısına ulaşıncaya kadar devam eder ve algoritma sonlandırılır.

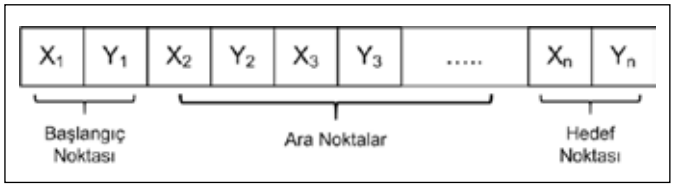
YAK algoritmasının temel adımları aşağıda belirtilmiştir;

- Rastgele başlangıç kaynakları oluştur, x_{ij}, $i = 1 \ldots SN$, $j=1 \ldots D$
- Her bir kaynağın amaç fonksiyonunu hesapla, f_i
- iterasyon = 1
- repeat
  - Denklem (2)'yi kullanarak v_i komşu kaynağını üret ve f_i değerini hesapla
  - v_i ile x_i arasında açgözlü seçim işlemi uygula
  - Denklem (4)'ü kullanarak kaynakların olasılıklarını belirle, p_i
  - p_i olasılığına göre x_i kaynağını seç
  - x_i komşuluğunda yeni bir v_i kaynağını üret ve f_i değerini hesapla
  - v_i ile x_i arasında açgözlü seçim işlemi uygula
  - Tükenen kaynak var ise o kaynak için denklem (1) ile rastgele yeni bir kaynak üret
  - Şimdiye kadar bulunan en iyi çözümü sakla
  - iterasyon = iterasyon + 1;
- until iterasyon = maksimum çevrim sayısı

3. Bulgular

3.1.Yapay Arı Kolonisi (YAK) Algoritması ile Rota Planlama

Rota planlama probleminde, İHA'nın başlangıç ve hedef noktaları arasında geçilecek bağlantı koordinat noktalarının tespit edilip, o noktalar üzerinden geçerek hedefe varması hedeflenmektedir. Geçilecek bağlantı noktaları belirlenirken iki temel kriter dikkate alınmıştır. Bu kriterler; rota üzerindeki engellerin dikkate alınması ve rotanın en kısa uzunlukta olacak şekilde belirlenmesi. Rota planlama problemine çözüm üretmek için kullanılan YAK algoritması ile başlangıç aşamasında aday rotalar denklem (1) kullanılarak rastgele üretilmektedir. Başlangıç ve hedef arasındaki mesafeye ve engel sayısına göre değişen haritalarda sabit bağlantı noktaların kullanılması dezavantaj oluşturabilmektedir. Bu bakımdan çalışmada, rota üretimi sırasında başlangıç ve hedef koordinatları arasında üzerinden geçilecek olan bağlantı koordinat noktalarının sayısı dinamik olarak belirlenmektedir. Örneğin; algoritma başlangıç olarak iki bağlantı koordinat noktası ile algoritmaya başlamakta, iki adet bağlantı koordinat noktası ile hedefe çözüm mümkün olmadığı durumda aradaki koordinat nokta sayısını birer arttırarak çözüm aramaya devam etmektedir. Her bir bağlantı koordinat noktası için rastgele enlem ve boylam bilgisi üretilmektedir. Algoritmada kullanılan rota yapısı Şekil 1'de gösterilmektedir.



Şekil 1. Rota yapısı.

Rastgele rotaların üretiminden sonra her bir rotanın maliyeti (amaç fonksiyon) hesaplanmaktadır. Maliyet hesaplanırken her bir rotanın uzunluğu ve geçiş güzergâhında engellerin olup olmadığı kontrol edilmektedir.

3.2. Amaç Fonksiyon

Her bir rotanın amaç fonksiyon değeri hesaplanırken, önce rotanın mesafesi hesaplanmakta daha sonra rota üzerinde herhangi bir engel olup olmadığı tespit edilip, var ise amaç fonksiyon değerine ek olarak engel maliyeti de eklenmektedir. GPS verileri üzerinden iki nokta arasındaki uzaklığı hesaplarken dünyanın geometrik şekli göz önüne alınmıştır ve bunun için Haversine (Montavont ve Noel 2006) formülü kullanılmıştır. Haversine formülü enlemi ve boylamı bilinen iki koordinat arasındaki mesafenin hesaplanması için kullanılmaktadır. Enlemler arasındaki fark Δ_{enlem}, boylamlar arasındaki fark Δ_{boylam} ve R dünyanın yarıçapı ($R=6,371km$) olarak ifade edilmiştir. İki koordinat noktası arasındaki mesafe d, aşağıdaki denklemler ile hesaplanmaktadır (Montavot ve Noel 2006).

$$h = haversin(\Delta_{enlem}) + \cos(enlem_1) \times \cos(enlem_2) \times haversin(\Delta_{boylam}) \quad (5)$$

$$haversin(\delta) = sin^2\left(\frac{\delta}{2}\right) \quad (6)$$

$$d = 2 \times R \times arcsin(\sqrt{h}) \quad (7)$$

Algoritmada başlangıç aşamasında rastgele rotalar üretildiğinden, rotalar üzerinde engelli bölgelerin de olma ihtimali bulunmaktadır. Öncelikle herhangi bir engele çarpmadan veya engelli bir alana girmeden rota planlaması yapılması gerektiğinden, engelli alanların belirlenmesi gerekmektedir. Rota üzerindeki engellerin tespiti için, denklemi bilinen bir doğrunun bir noktaya olan uzaklığı formülü kullanılmıştır. Örneğin; $A(x_1,y_1)$ ve $B(x_2,y_2)$ noktaları arasındaki doğrunun herhangi bir engelden geçip geçmediğini bulmak için öncelikle denklem (8) kullanılarak doğrunun denklemi edilmektedir.

$$\frac{y_2 - y_1}{x_2 - x_1} = \frac{y - y_2}{x - x_2} \quad (8)$$

Denklemin uygulanması sonucunda, $a=(y_2-y_1)$, $b=(x_2-x_1)$ ve $c=(y_1 \times x_2 - y_2 \times x_1)$ olarak bulunmaktadır. Bu durumda, $O(x_3,y_3)$ merkezli bir engelin AB doğrusuna uzaklığı denklem (9) ile hesaplanmaktadır.

$$d_{engel} = \frac{|ax_3 + by_3 + c|}{\sqrt{a^2 + b^2}} \quad (9)$$

d_{engel}, engelin yarıçap değerinden küçük veya eşit olduğunda rotanın engelli bir alandan geçtiğini ifade etmektedir. Rota üzerindeki noktalar arasında bulunan engel sayısı kadar, mevcut rotanın amaç fonksiyon değerine engel maliyeti eklenmektedir. Algoritma en az maliyetli amaç fonksiyon değerlerini bulmaya çalıştığından, engel maliyetli amaç fonksiyon değerlerine sahip rotalar algoritmanın iteratif çalışması ile birlikte silinecektir. Bu bilgilere dayanarak, bir rotanın amaç fonksiyon değeri denklem (10) ile hesaplanmaktadır.

$$f_i = d_i + E_i \quad (10)$$

f_i değeri $i.$ rotanın amaç fonksiyon değerini, d_i değeri $i.$ rotanın uzunluğunu ifade etmektedir. E_i değeri ise $i.$ rota üzerindeki toplam engel sayısı ile engel maliyetinin çarpımını göstermektedir.

Çalışmada, İHA için en uygun rota bulunduktan sonra, takip edilecek rota üzerinde bulunan coğrafi engeller (dağ, tepe vb.) de dikkate alınmaktadır. Coğrafi engellerin koordinat, yükseklik bilgileri Google gMapControl eklentisi aracılığıyla tespit edilmektedir. Belli adımlarda, yükseklik bilgileri alınarak rota üzerinde bulunan coğrafi engellerin tespit edilmesi için tarama mesafesi kullanılmıştır. Yüksekliklerin belirlemesi için başlangıç noktasından hedef noktasına kadar olan rota üzerinde herhangi bir coğrafi engel olup olmadığının kontrolü için yükseklik matrisi oluşturulmuştur. Tarama mesafesi, rota uzunluğuna göre orantısal olarak değişmekle birlikte, sabit olarak ta kullanılmıştır. Benzetim çalışmaları için tarama mesafesi 50m olarak kullanılmıştır. Bu durumda rota boyunca her 50m'de bir coğrafi engel olup olmadığı ve varsa yüksekliği kontrol edilmektedir. Rota üzerinde coğrafi engelli bölgeye yaklaşıldığında, bu engellerden sakınım için, İHA'nın kendi etrafında dönerek engelleri aşması hedeflenmiştir. İHA'nın kendi etrafında dönerek ulaşması gereken coğrafi engeli aşması için (11) numaralı denklemden yararlanılmıştır.

$$h = 2\pi r \times \tan(\propto) \quad (11)$$

h, İHA'nın kendi etrafında bir tur döndüğünde aldığı yükseklik mesafesidir. α yükseliş açısı ve r ise İHA'nın kendi etrafında döneceği minimum yarıçap değeridir. Coğrafi engelin yüksekliği bilindiğinde, h değeri ile İHA'nın kendi etrafında kaç tur dönmesi gerektiği tespit edilmektedir.

3.3. Rota Planlama için Kullanıcı Arayüzü

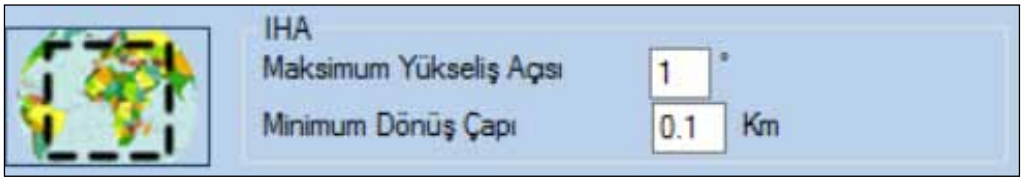
Rota planlama için geliştirilen kullanıcı arayüzü üç aşamadan oluşmaktadır. Birinci aşamada; rotanın başlangıç ve bitiş konumları girilerek harita üzerinde tespit edilmekte, ikinci aşamada; enlem-boylam bilgileri harita üzerinde fare yardımıyla veya el ile girilerek sanal engeller oluşturulmakta, üçüncü aşamada; İHA ve YAK için gerekli parametre ayarları yapılmaktadır. C# programlama dili kullanılarak geliştirilen uygulama Şekil 2'de gösterilmektedir.

Geliştirilen uygulama ile harita üzerinde işlemler yapabilmek için Google gMapControl eklentisi kullanılmıştır. Bu eklenti ile Google haritası kullanılarak gerçek zamanlı bilgilerden faydalanılmıştır. Başlangıç ve hedef koordinat noktalarını belirlerken, arayüzde bulunan adres satırına başlangıç ve hedef noktalarının isimlerinin girilmesi ile gMapControl eklentisinden faydalanılarak bu noktalar harita üzerinde işaretlenmektedir.

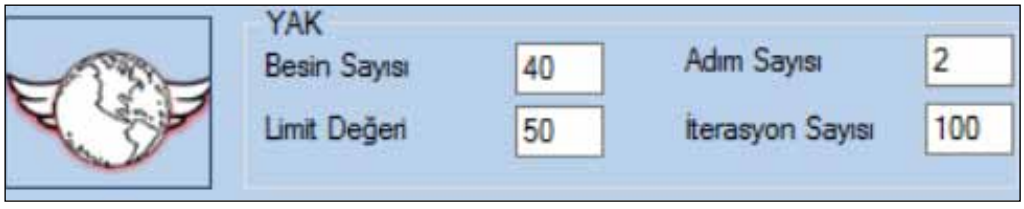
Geliştirilen kullanıcı arayüzünde İHA için yapılacak ayarlar Şekil 3'de gösterilmektedir. Arayüzde bulunan Maksimum Yükseklik Açısı, İHA'nın dikey eksende yapacağı açıyı (yunuslama) belirtmektedir. Her İHA'nın yükseliş açısı farklı olduğundan, kullanılacak olan İHA'ya ait bu açının kullanıcı tarafından girilmesi gerekmektedir. Arayüzde bulunan Minimum Dönüş Çapı ile İHA'nın kendi etrafında dönerken yapacağı minimum dairesel hareketin çap bilgisi belirtilmektedir.

YAK algoritması için gerekli olan ayarların yapıldığı ekran görüntüsü Şekil 4'de gösterilmektedir. Besin Sayısı aday

Şekil 2. YAK ile rota planlama için kullanıcı arayüzü ekranı.



Şekil 3. İHA ayarları.



Şekil 4. YAK algoritması ayarları.

rota sayısını, Adım Sayısı ise bağlantı koordinat noktası sayısını ifade etmektedir. Adım sayısı kullanıcı tarafından girilen bir değerden başlayabildiği gibi, herhangi bir değer girilmediği takdirde algoritma tarafından dinamik olarak ta belirlenmektedir.

Uygulama, en uygun rotayı oluşturduktan sonra otomatik olarak bir rapor oluşturmaktadır. Bu raporda, İHA ve YAK ayarları, başlama ve bitiş koordinat noktaları, algoritma ile bulunan bağlantı koordinat noktaları ve sanal engellerin koordinatları bulunmaktadır. Rota üzerindeki coğrafi engellerin koordinatları, yükseklikleri ve İHA'nın kendi etrafında dönerek ne kadar tur atması gerektiği ile ilgili bilgiler de raporda kullanıcıya sunulmaktadır.

3.4. Benzetim Uygulamaları

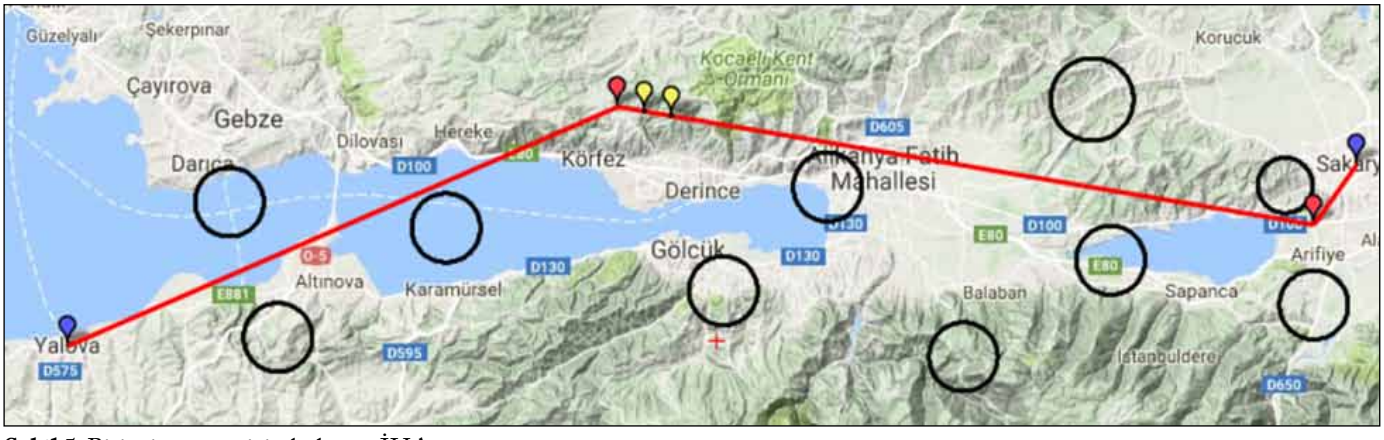
Çalışmada, YAK ile rota planlama probleminin çözümü ve kullanıcı arayüzünün benzetim uygulamaları için iki farklı senaryo ele alınmıştır. Benzetim çalışmalarında rota üzerinde görülen sarı işaretli noktalar coğrafi engelleri ifade etmektedir. İHA'nın rota üzerinde ilerlerken bu noktalara geldiği zaman kendi etrafında dönerek engeli aşması gerektiği öngörülmektedir.

Birinci senaryo; Yalova'dan Sakarya'ya gidecek olan İHA için farklı yerlere on adet sanal engel eklenmiştir. İHA'nın maksimum yükseliş açısı 1°C, kendi etrafında dönüş çapı 0.1km, YAK algoritması için besin sayısı 30, limit değeri 60 ve iterasyon sayısı 200 olarak ele alındığında bulunan en uygun rota Şekil 5'de gösterilmektedir. İki bağlantı noktası kullanılarak ulaşılan hedefte İHA'nın coğrafi koşullardan dolayı kendi etrafında yükselerek aşması gereken iki farklı bölgenin bulunduğu görülmektedir. Şekilde görüldüğü gibi, başlangıç ve bitiş noktaları arasındaki engelli bölgelere girmeden elde edilen rotanın uzunluğu 100.106km olarak bulunmuştur.

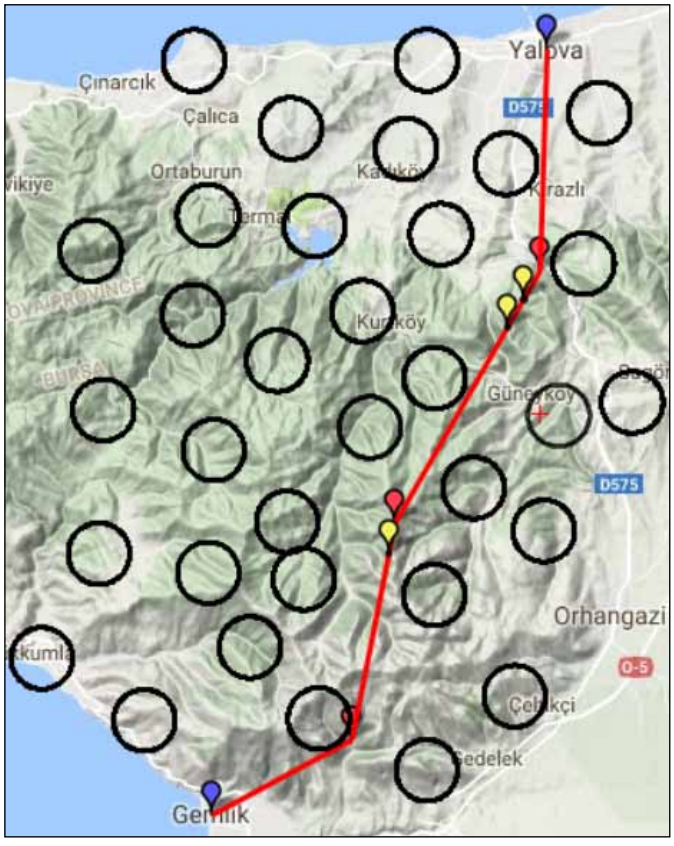
İkinci senaryo; Yalova'dan Gemlik'e hareket edecek olan İHA için 26 adet sanal engel yerleştirilmiştir. İHA'nın maksimum yükseliş açısı 4°C, kendi etrafında dönüş çapı 0.1km, YAK algoritması için besin sayısı 20, limit değeri 50, iterasyon sayısı 200 olarak ele alındığında bulunan en uygun rota Şekil 6'da gösterilmektedir. Üç bağlantı noktası kullanılarak ulaşılan hedefte İHA'nın coğrafi koşullardan dolayı kendi etrafında yükselerek aşması gereken üç farklı bölgenin bulunduğu görülmektedir. Şekilde görüldüğü gibi, başlangıç ve bitiş noktaları arasındaki engelli bölgelere girmeden elde edilen rotanın uzunluğu 27.790km olarak bulunmuştur.

4. Sonuç

Bu çalışmada, otonom İHA'lar için rota planlaması problemi YAK algoritması kullanılarak çözülmüştür. Rota üzerinde bulunan bağlantı koordinat noktaları dinamik olarak belirlenmiştir. Bu sayede algoritma engellerin sayısına göre bağlantı koordinat sayısını değiştirerek en uygun yolu bulmaya çalışmaktadır. C# programlama dili kullanılarak kullanıcı etkileşimli bir arayüz geliştirilmiştir. Geliştirilen arayüz uygulamasında Google çevrimiçi harita kullanılmış ve rota için gerekli olan koordinat ve yükseklik bilgileri bu harita ile sağlanmıştır. Arayüz yardımıyla kullanıcıların başlangıç, hedef ve engellerin konumlarını girerek, İHA ile YAK algoritması için gerekli ayarları yaparak benzetim uygulamaları yapabilmelerine olanak sağlanmıştır. Rota üzerindeki dağ, tepe gibi coğrafi bölgeler harita üzerinde kullanıcıya gösterilmiş ve bu bölgelerde İHA'nın kendi etrafında dönüp yükselerek coğrafi engeli aşması gerektiği tespit edilmektedir. Yapılan farklı benzetim uygulamaları, İHA için rota planlama probleminde YAK algoritmasının başarılı sonuçlar verdiğini göstermektedir



Şekil 5. Birinci senaryo için bulunan İHA rotası.



Şekil 6. İkinci senaryo için bulunan İHA rotası.

5. Kaynaklar

Akay, B. 2009. Nümerik Optimizasyon Problemlerinde Yapay Arı Kolonisi (Artificial Bee Colony) Algoritmasının Performans Analizi, Doktora tezi, Erciyes Üniversitesi, 2009.

Akyürek, S., Yılmaz, MA., Taşkıran, M. 2012. İnsansız Hava Araçları Muharebe Alanında ve Terörle Mücadelede Devrimsel Dönüşüm, Bilge Adamlar Stratejik Araştırmalar Merkezi, İstanbul, 2012.

Aydemir, H. 2014. İnsansız Hava Araçlarında Rotalama Problemi için Simülasyon Tabanlı Karar Destek Sistemi, Doktora tezi, Kara Harp Okulu, 2014.

Çekmez, U. 2014. İnsansız Hava Araçlarında Büyük Ölçekli Yol Planlamada Problemlerinin GPU Üzerinde CUDA Yardımı ile Çözümü, Yüksek lisans tezi, Hava Harp Okulu, 2014.

Gencer, C., Aydoğan, EK., Kocabaş, S. 2009. İnsansız Hava Araçlarının Rota Planlaması için Bir Karar Destek Sistemi, Hava Harp Okulu Savunma Bilimleri Dergisi, pp.59-73, 2009.

Ingham, LA. 2008. Considerations For A Roadmap For The Operation Of Unmanned Aerial Vehicles (UAV) in South African Airspace, Doktora Tezi, Stellenbosch University, 2008.

Karaboğa, D. 2004. Yapay Zeka Optimizasyon Algoritmaları, Atlas Yayın Dağıtım.

Karaboğa, D. 2005. An Idea Based on Honey Bee Swarm for Numerical Optimizasyon Technical Report-TR06, Erciyes Üniversitey, Engineering Faculty, Computer Engineering Department, 2005.

Karaboğa, D., Akay, B. 2007. Yapay Arı Koloni (Artificial Bee Colony, ABC) Algoritması ile Yapay Sinir Ağlarının Eğitilmesi, 15th Signal Processing and Communications Applications, 2007

Karaboğa, D., Akay, B. 2009. A comparative study of Artificial Bee Colony algorithm. *Appl. Math. Comput.,* 214:108-132.

Montavont, J., Noel, T. 2006. IEEE 802.11 Handovers Assisted by GPSInformation, IEEE International Conference on Wireless and Mobile Computing Networking and Communications, 2006.

Özalp, N. 2013. 3 Boyutlu Arazi Üzerinde Çoklu Otonom İnsansız Hava Aracı Rota Planlaması, Yüksek lisans tezi, Hava Harp Okulu, 2013.

Ryan, JL., Bailey, TG., Moore, JT., Carlton, WB. 1998. Reactive Tabu Search in Unmanned Aerial Reconnasissance Simulations, Procededings of the 1998 Winter Simulation Confrence, pp.873-879.

Savunma Sanayi Müsteşarlığı 2012. İHA Sistemleri Yol Haritası (2011-2030), Ankara, 2012.

Tulum, K. 2009. Route Planning For Unmanned AirVehicle, Yüksek lisans tezi, Orta Doğu Teknik Üniversitesi, 2009.