DOKUZ EYLÜL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

İNSANSIZ BİR HAVA TAŞITININ KENTİÇİ HARİTA ÜZERİNDE KONUMLANDIRILMASI VE HEDEF NOKTA İÇİN YÖNELME ALGORİTMASININ GELİŞTİRİLMESİ

Kerem ALTINIŞIK

Temmuz, 2013
İZMİR

İNSANSIZ BİR HAVA TAŞITININ KENTİÇİ HARİTA ÜZERİNDE KONUMLANDIRILMASI VE HEDEF NOKTA İÇİN YÖNELME ALGORİTMASININ GELİŞTİRİLMESİ

Dokuz Eylül Üniversitesi Fen Bilimleri Enstitüsü

Yüksek Lisans Tezi

Mekatronik Mühendisliği Anabilim Dalı, Mekatronik Mühendisliği Programı

Kerem ALTINIŞIK

Temmuz, 2013

İZMİR

YÜKSEK LİSANS TEZİ SINAV SONUÇ FORMU

KEREM ALTINIŞIK, tarafından **YRD. DOÇ. DR. AYTAÇ GÖREN** yönetiminde hazırlanan **"İNSANSIZ BİR HAVA TAŞITININ KENTİÇİ HARİTA ÜZERİNDE KONUMLANDIRILMASI VE HEDEF NOKTA İÇİN YÖNELME ALGORİTMASININ GELİŞTİRİLMESİ"** başlıklı tez tarafımızdan okunmuş, kapsamı ve niteliği açısından bir Yüksek Lisans tezi olarak kabul edilmiştir.

Yrd. Doç. Dr. Aytaç GÖREN

Yönetici

Jüri Üyesi

Jüri Üyesi

Prof. Dr. Ayşe OKUR

Müdür

Fen Bilimleri Enstitüsü

TEŞEKKÜR

Bana yol ve sabır gösteren danışmanım Sayın Yrd. Doç. Dr. Aytaç GÖREN' e ve yüksek lisans eğitimimizin başından beri birbirimize destek olduğumuz Sayın Aydın AYTAÇ' a çok teşekkürler.

Ayrıca desteklerini esirgemeyen anneanneme, aileme ve iş arkadaşlarıma sonsuz teşekkürler.

Kerem ALTINIŞIK

İNSANSIZ BİR HAVA TAŞITININ KENTİÇİ HARİTA ÜZERİNDE KONUMLANDIRILMASI VE HEDEF NOKTA İÇİN YÖNELME ALGORİTMASININ GELİŞTİRİLMESİ

ÖZ

Daha önce tasarlanmış ve imal edilmiş dört rotorlu helikopterin harita üzerinde gösterimi ve yönelme algoritmasının yazılması amaçlanmıştır. Araçla haberleşmeyi sağlamak için GPS Evaluation Board üzerine 20 Kanal GPS alıcı anten EM-406A kullanılmıştır.

Seri Port üzerinden gelen araç enlem-boylam bilgileri işlenmiş, Google Maps API v3 kullanılarak harita üzerinde gösterimi sağlanmıştır.

Bir web servis yardımıyla hedef nokta adres bilgisi enlem boylam bilgisine çevrilmiş ve dört rotorlu helikoptere gönderilmek üzere hazırlanmıştır.

Yazılım C# programlama dilinde yazılmıştır. Her ne kadar masaüstü uygulaması olsa da mimari açıdan proje html ve javascript kodları da muhteva etmektedir.

Sistemin matematiksel modellemesi incelenmiş ve Simulink programında benzetimi gerçekleştirilmiştir.

Anahtar sözcükler: Dört rotorlu helikopter, enlem, boylam, konumlandırma, insansız hava taşıtı, yönelme algoritması, haritalama

POSITIONING AN UNMANNED AERIAL VEHICLE ON CITY MAP AND DEVELOPING FORMATION ALGORITHM FOR A DESTINATION POINT

ABSTRACT

Screening the designed and constructed four rotors helicopter on a map and writing the formation algorithm is purposed. To provide communication through vehicle, GPS receiver antenna EM-406A on GPS Evaluation Board is used.

The latitude-longitude information which comes through serial port, is parsed; screening on map using by Google Maps API v3 is provided.

Destination point address information is converted to longitude-latitude information by a web service and prepared to send to four rotors helicopter.

Software is written via C# programming language. Although it is an Windows Forms Application, architecturally project includes html and javascript codes as well.

Keywords: Four rotors helicopter**,** quadrocopter, latitude, longitude, positioning, unmanned aerial vehicle, formation algorithm, mapping

İÇİNDEKİLER

Sayfa

ŞEKİLLER LİSTESİ

Sayfa

BÖLÜM BİR
GİRİŞ

1.1 İnsansız Hava Araçları Hakkında Kısa Bilgi

İnsansız Hava Araçları (İHA) havacılık sektörünün başından bu yana var olmuştur. Son 30 yıldır insansız hava araçları saldırı amaçlı askeri uygulamalarda faydalı bulunmuştur.

İHA'lar yirminci yüzyılın başlarında uçak ve uçaksavar kuvvetlerin hedef almalarına hizmet etmek için icat edilmiştir (Taylor, 2004).

Görevlerine göre sınıflandırmak gerekirse sivil iha sistemleri ve askeri iha sistemleri olmak üzere iki grupta inceleyebiliriz. Sivil iha sistemleri de kendi içinde sivil taşımacılık, bilimsel/arazi izleme, keşif/gözetleme, uydu görevlerini bütünleyicilik ve acil durumlar alt başlıklarına ayrılır. (Güllü, 2012)

Bir uçak ya da helikoptere göre küçük olması, daha ucuz olması ve pilot ölüm riski taşımaması avantajlarıdır. Kötü havalarda haberleşme sorunları ve kullanım zorluğu dezavantajları arasında yer alabilir.

Dört rotorlu insansız hava araçlarının ise diğer insansız hava araçlarına göre avantajları, iyi manevra kabiliyeti ve basit mekanik yapısıdır. Dezavantaj olarak da yüksek enerji tüketimi gösterilebilir.

1.2 Amaç

İnsanoğlu gerek yiyecek ve su gerekse diğer ihtiyaçları için yer değiştirmeye başladığı zamanlardan beri rota tayini ve yol bulması gerekmiştir. Elektronik sistemlerin ve bilgisayarların gelişimiyle bu gereksinimlere otonom çözümler üretilmeye başlanmıştır. Bu tez çalışmasında daha önce mekanik tasarımı ve montajı

yapılmış insansız dört rotorlu helikopterin modellenmesi, haberleşmesi, harita üzerinde gösterimi ve yönelme algoritmasının yazılımı amaçlanmıştır.

BÖLÜM İKİ
TEORİK ARKA PLAN

İnsansız hava aracı, üzerinde bulunan GPS modülünün aldığı enlem boylam bilgilerini bir radyo frekansı alıcı verici modülü vasıtasıyla bilgisayara iletmektedir. Bilgisayarın seri portu okunarak istenilen veriler süzülmektedir. Söz konusu veriler derece ve dakika cinsindendir ve ondalık sayıya çevrilmektedir. Akabinde matematiksel bir fonksiyonla yer yön tayini yapılmaktadır. Dönüş açıları hesaplanmaktadır. Nihayetinde gerekli motora ilgili bilgi aktarılmaktadır. Yazılımın mimari katmanlarını inceleyecek olursak proje temelde Windows Form Uygulamaları olmakla beraber, bir HTML sayfasını işler. Bu da geride Javascript kodu yazılan bölümdür. Bu mimari katmanların temel sebebi "Google Maps Api v3" kullanmak için javascript dili seçilmiştir. Diğer taraftan bilgisayar İHA bilgilerini seri port yoluyla alacaktır. Windows Form Uygulamaları bu aşamada çok uygundur ve kararlıdır.

2.1 Kullanılan Mantıksal ve Matematiksel Fonksiyonlar

2.1.1 Harita Üzerinde İki Nokta Arasındaki Uzaklık

Dünyanın küresel şeklinden dolayı harita üzerindeki iki nokta arasındaki uzaklık, koordinatları bilinen iki noktanın doğrusal uzaklığı biçiminde çözüldüğünde yeterince doğru sonuçlar vermeyebilir. Bu projede iki nokta arasındaki büyük daire uzaklığını hesaplamak için küresel trigonometrik bir yaklaşım olan Haversine formülü kullanılmıştır. Haversine formülü kısa mesafeler için bile yeterli sonuçlar vermektedir.

$$\text{haversine}\left(\frac{\text{d}}{\text{r}}\right) = \text{haversine}(\theta_1 - \theta_2) + \cos(\theta_1)\cos(\theta_2)\text{haversine}(\varphi_1 - \varphi_2) \quad (1)$$

Buradan programlama dilinde kullanıma dökersek;

$$\text{a} = \sin^2((\theta_2 - \theta_2)/2) + \cos(\theta_1)\cos(\theta_2)\sin^2((\varphi_1 - \varphi_2)/2) \quad (2)$$

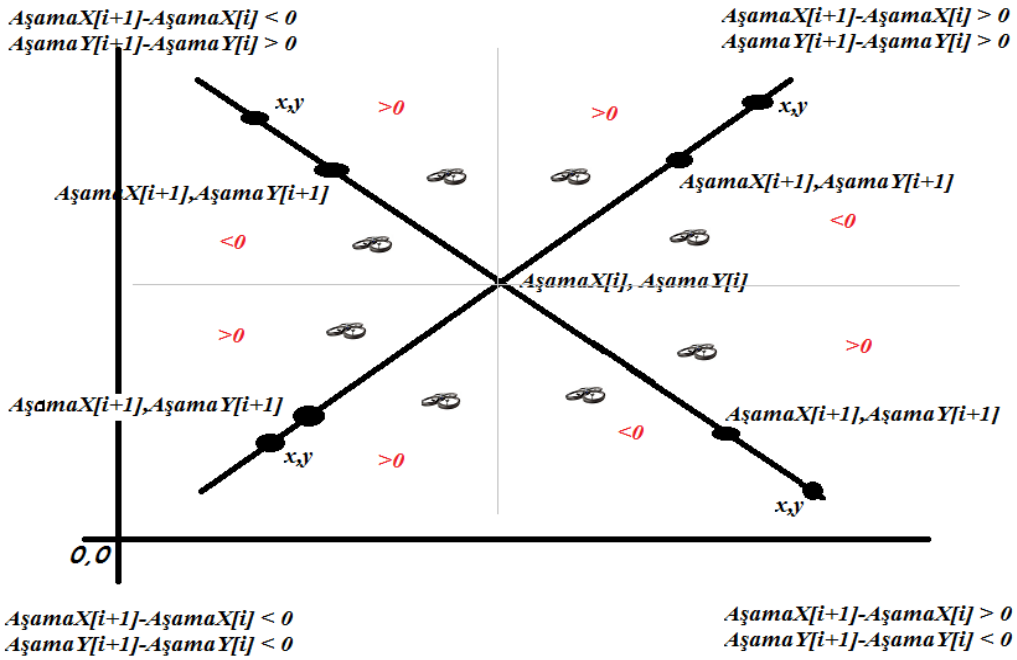
$$c = 2 * \operatorname{atan2}(\sqrt{a}, \sqrt{1-a}) \quad (3)$$

$$d = r * c \quad (4)$$

elde edilir. Burada d küre üzerindeki iki farklı nokta arasındaki uzaklıktır. r dünyanın yarıçapı; θ_1, θ_2 noktaların enlemleri ve φ_1, φ_2 noktaların boylamlarıdır.

2.1.2 Bir Noktanın Doğrulara Göre Düzlemsel Konumu

"Google Maps" web servisinden çekilen yer yön tarifi çeşitli aşamalar sonucunda bizi hedefe yönlendirmektedir. İHA, iki aşama noktası arasındaki doğruya göre yönünü belirlemektedir. Bir başka deyişle İHA'nın takip etmesi gereken doğru matematiksel olarak belirlenmelidir. Şekil 2.1'de (0,0) noktası referans noktası seçilmiştir. Düzlem dörde ayrılmıştır. Öncelikle, bir sonraki aşamanın düzlemin hangi kısmında olduğu tayin edilmiştir. Bu tayin de, iki aşamanın enlem, boylam noktaları farkını 0 değeri ile karşılaştırarak yapılmıştır. İki aşamanın (iki enlem boylam noktasının) bir doğru oluşturacağından mütevellit İHA'nın anlık enlem değeri bu doğru denkleminde yerine koyulmuştur. Çıkan boylam değeri ile İHA'nın anlık boylam değeri karşılaştırılmıştır ve yön tespiti gerçekleştirilmiştir.



Şekil 2.1 İHA'nın düzlemde yönelme mantığı

$$\frac{\text{x}-\text{AsamaX[i]}}{\text{AsamaX[i+1]}-\text{AsamaX[i]}} = \frac{\text{y}-\text{AsamaY[i]}}{\text{AsamaY[i+1]}-\text{AsamaY[i]}} \quad (5)$$

$$\text{AnlıkY} - \left[\frac{AsamaY[i+1]-AsamaY[i]}{AsamaX[i+1]-AsamaX[i]}(AnlıkX - AsamaX[i]) + AsamaY[i]\right] \lessgtr ? \ 0 \quad (6)$$

2.1.3 Kosinüs Teoremi

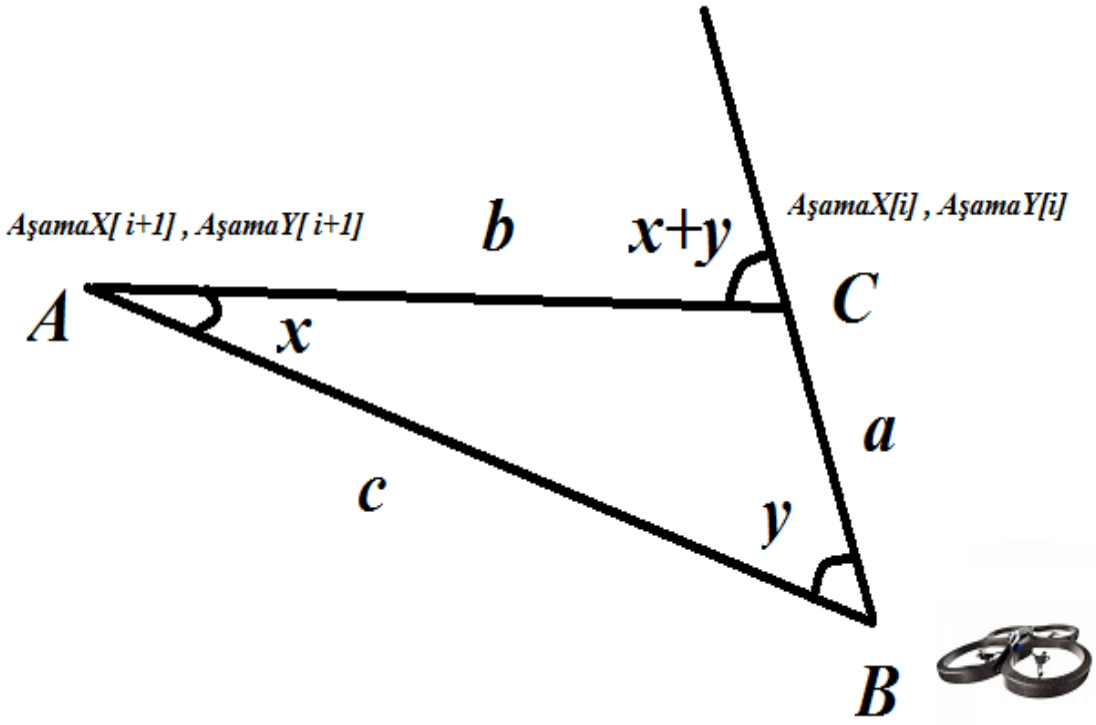
İHA bir sonraki ara adıma yaklaşırken bulunduğu nokta, önündeki aşama ve bir sonraki aşama arasında bir üçgen oluşturmaktadır. Dönüş açısını hesaplarken kosinüs teoremi kullanılmıştır. Resim 2.2'de görüldüğü üzere

$$a^2 = b^2 + c^2 - 2bc\cos x \quad (7)$$

$$b^2 = a^2 + c^2 - 2ac\cos y \quad (8)$$

$$x + y = \cos^{-1}((b^2 + c^2 - a^2)/2bc) + \cos^{-1}((a^2 + c^2 - b^2)/2ac \quad (9)$$

elde edilir.



Şekil 2.2 Kosinüs teoremi ile dönüş açısı hesaplama

2.2 Kullanılan Programlama Dili ve Yazılım Kütüphaneleri

2.2.1 C# ve .NET Framework

C#, .NET Framework üzerinde çalışan, güvenli ve sağlam uygulama çeşitliliği oluşturmak için geliştiriciler sağlayan nesne yönelimli bir dildir. Geleneksel "Windows" istemci uygulamaları, XML Web hizmetleri, dağıtılmış bileşenler, istemci-sunucu uygulamaları, veri tabanı uygulamaları gibi uygulamaları oluşturmak için C # kullanabilmektedir. "Visual C# 2010"; .NET Framework'ün 4.0 versiyonuna ve C # dili 4.0 versiyonuna dayalı, kolay uygulamalar geliştirmek için gelişmiş bir kod editörü, uygun kullanıcı ara yüzü tasarımcıları, bütünleşmiş debugger ve diğer birçok araç sunmaktadır.

Proje C# dilinde Windows Forms uygulaması olarak yazılmıştır.

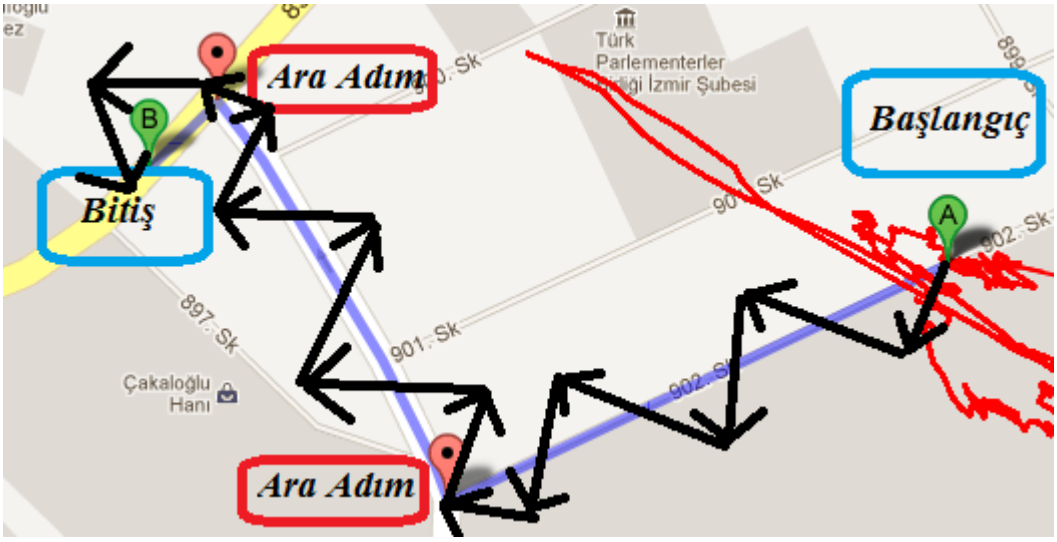
2.2.2 Google Maps Javascript Api

Javascript web tarayıcılarının üzerinde çalışan, web uygulamalarına işlerlik kazandırmak için geliştirilmiş nesne tabanlı bir betik dilidir. “Google Maps Api” ise “Google” şirketinin sunduğu haritayı kullanmak için geliştirilmiş bir javascript kütüphanesidir.

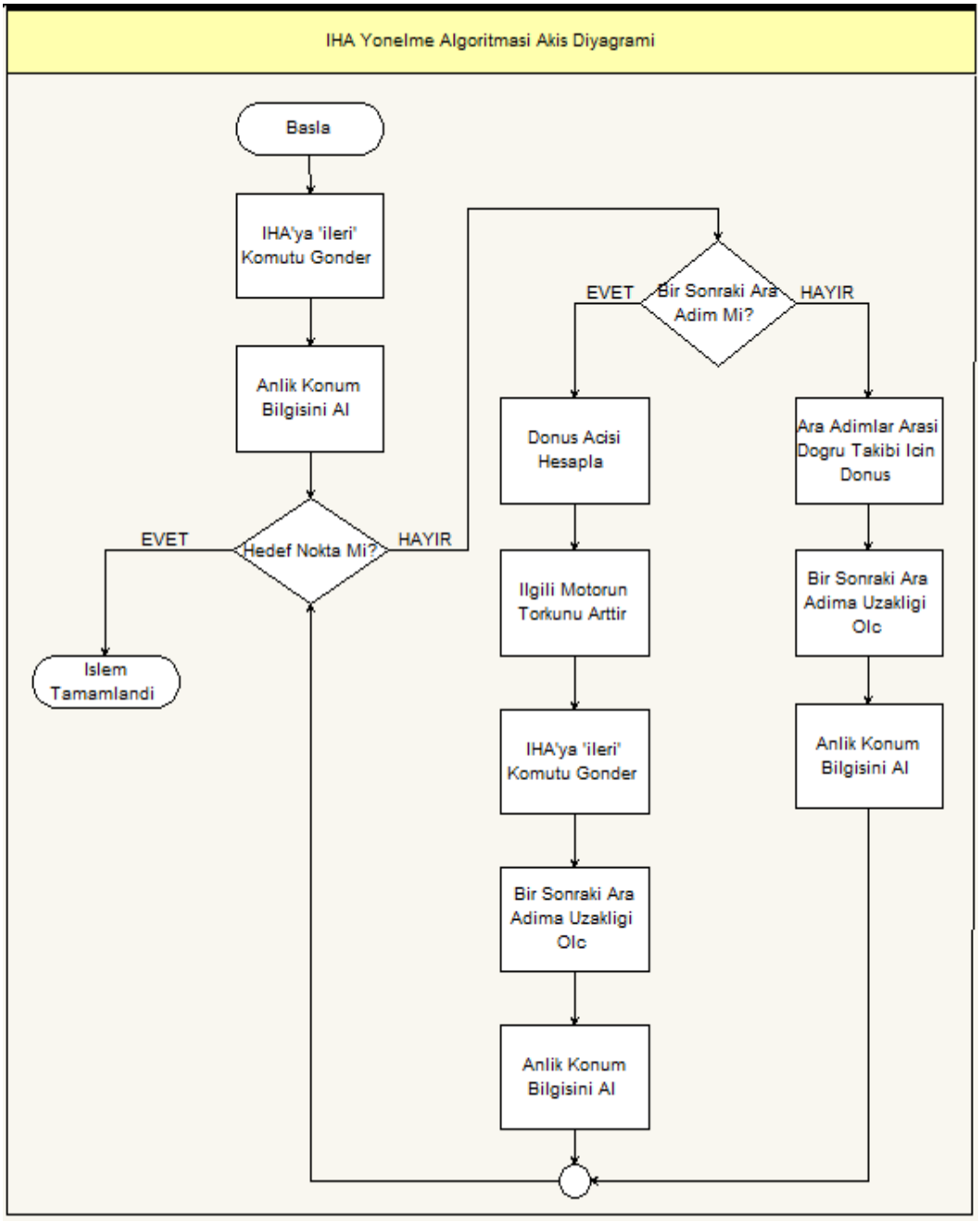
Harita üzerindeki hedef noktaya ulaşmak amacıyla yön tarifi desteği için “Directions Service” kullanılmıştır. “Google Maps Api” özellikleri ve bu özellikleri kullanan fonksiyonlar daha detaylı olarak yeri geldikçe üçüncü bölümde anlatılacaktır.

2.3 Yönelme Algoritması

İHA’nın hedefe ulaşması için sürekli kontrol etmesi gereken 2 nokta mevcuttur. Bu noktalar Şekil 2.3’te gösterildiği üzere bitiş noktası ve ara adımlardır. İHA ara adımlar arasında bir doğru hesaplayarak bu doğruya tabi olmaya çalışacaktır ve zikzak bir yol çizecektir. Söz konusu akış diyagramı Şekil 2.4’te gösterilmektedir.



Şekil 2.3 İHA’nın takip etmesi planlanan yol



Şekil 2.4 İHA yönelme algoritması akış diyagramı

2.4 Haberleşme Donanım Şekli ve Kullanılan Cihazlar

Bu bölümde kullanılan malzemelerden kontrol kartı, kumanda, radyo frekansı alıcı verici modülü ve sensör dağıtım soketi, GPS modülü ve anteni tanıtılacaktır.

2.4.1 GPS Modülü Ve Anteni

GPS modülü için “Sparkfun GPS Evalution Board” kullanılmıştır. Modül FT232RL üzerinden USB bağlantıya ve RS232 seri ara yüzüne sahiptir. Adaptörle olduğu gibi USB ile de beslenebilmektedir.

Modüle uyumlu anten olarak “EM-406A” tercih edilmiştir. 20 kanala sahiptir. 1 saniye “Hot Start”, 38 saniye “Warm Start” ve 42 saniye “Cold Start” sürelerine sahiptir.

2.4.2 Radyo Frekansı Alıcı Verici Modülü Ve Sensör Dağıtım Soketi

Radyo frekansı alıcı verici modül için “Frsky DJT 2.4 Ghz Combo Pack for JR w/ Telemetry Module & V8FR-II RX” tercih edilmiştir. 1.5 km ile 2.5 km arası menzile sahiptir. 8 kanallıdır.

2.4.3 Kontrol Kartı

Kontrol kartı için “HobbyKing Multi-Rotor Control Board v2.1” tercih edilmiştir. Üzerinde “Atmega168PA” bütünleşik devresi mevcuttur. İHA’yı dengede tutmak amacıyla kart üzerinde 3 adet gyro sensörü bulunmaktadır.

BÖLÜM ÜÇ
SİMÜLASYON

3.1 Simülasyonun Temel Aşamaları

Simülasyondan kasıt, İHA'ya bilgisayar ortamında program vasıtasıyla oluşturulmuş enlem-boylam noktaları silsilesini ileri sürerek programın davranışını incelmekten ibarettir.

3.1.1 Haritada Gösterim

"Windows Form Uygulamaları" projesinin çizim kısmına "WebBrowser" aracı eklenmiştir. Böylece bir tarayıcı yardımı ile geride bir HTML dosyasının çalışması sağlanmaktadır.

```
Uri uri = new Uri(di.Parent.Parent.FullName + "\\map_routebuffer1.html");
webBrowser1.Navigate(uri);
```

Şekil 3.1 Harita gösterimi için C# kodu

3.1.2 Yol Tarifi

"Google Maps Api" kütüphanesinin "Directions Service" isimli servisi kullanılarak kaynak ve hedef noktalar arası yol tarifi alınmaktadır. Bunun için HTML tarafında javascript ile bu servis çağırılmaktadır. Şekil 3.3'te "origin" ve "destination" değerleri form uygulamasında yazı kutusu aracından alınmaktadır. Bunun için Şekil 3.2'de gösterildiği üzere "StreamReader" sınfı kullanılarak HTML sayfası okunmakta ve yazılmaktadır.

```
StreamReader objReader = new StreamReader(di.Parent.Parent.FullName + "\\map_route.html");
string line = "";
line = objReader.ReadToEnd();
objReader.Close();
line = line.Replace("[origin]", txt_currentpoint.Text);
line = line.Replace("[destination]", textBox_coords.Text);
```

Şekil 3.2 Javascript ile kaynak ve hedef nokta atanması C# kodu

```
function calcRoute() {
    var selectedMode = document.getElementById("mode").value;
    var request = {
        origin: origin,
        destination: destination,
        // Note that Javascript allows us to access the constant
        // using square brackets and a string value as its
        // "property."
        travelMode: google.maps.TravelMode[selectedMode]
    };
    directionsService.route(request, function (response, status) {
        if (status == google.maps.DirectionsStatus.OK) {
            directionsDisplay.setDirections(response);
            showSteps(response);
            CSyegonder(response);
        }

    });

}
```

Şekil 3.3 "Google Maps Api Directions Service" ile yol tarifi alan javascript kodu

3.1.3 Yol Tarifi Ara Adımlarının Alınması

"Directions Service" servisinden çekilen ara adım bilgileri (yön) form uygulamasında "DataGridView" aracının listesine aktarılmaktadır. Bu durum Şekil 3.4'te gösterilmiştir. Şekil 3.6 servisten sonuçları çeken fonksiyonu, Şekil 3.5 ise bu sonuçları form uygulamasına aktaran fonksiyonu göstermektedir.

No	Yön Komutlan	HedefEnlemBoylam
1	ileri	38.40791, 27.11415999999997
2	sola	38.40916, 27.11417000000006
3	sağa	38.40915, 27.114649999999983
4	sola	38.40974000000001, 27.1145999999

Şekil 3.4 Yön tarifi ara aşamalarının listesi

“Google Maps Directions” servisi yön bilgisi talebine “status” meta verisi ve “routes” katarıyla cevap vermektedir. Bu projede “routes” katarıyla ilgilenilmektedir. “routes” katarı ise kendi içinde; “summary”, “legs”, “waypoint_order”, “overview_polyline”, “bounds”, “copyrights” ve “warnings” alanları bulundurmaktadır. Bu noktada ise “legs” alanı tahlil edilmektedir. “legs” ise kendi içinde “steps”, “distance”, “duration”, “duration_in_traffic”, “arrival_time”, “departure_time”, “start_location”, “end_location”, “start_address” ve “end_address” alanlarını muhteva etmektedir. Şekil 3.6’da “steps” yani bu yazıda hep ara adım olarak nitelendirilen alanın “instructions” bölümünden yön tarif bilgilerini kırpıp yön komutlarının nasıl elde edildiği gösterilmiştir. Şekil 3.5’te ise bu verilerin form kısmına aktaran ve “DataGridView” aracına aktaran kod parçacığı gösterilmektedir.

```
void VeriGetir()
{
    object z = webBrowser1.Document.InvokeScript("Sonuclar");
                                .
                                .
                                .
    dataGridView1.Rows.Add(j + 1, YonKomut[j], YonLatLon[j]);
```

Şekil 3.5 “Google Maps Directions” servisinden çağırılan verilerin form kısmına aktaran ve “DataGridView” listesine ekleyen C# kodu

```
function showSteps(directionResult) {
    var myRoute = directionResult.routes[0].legs[0];
    for (var i = 0; i < myRoute.steps.length; i++) {
        myRoute.steps[i].instructions = myRoute.steps[i].instructions.toLowerCase();
        if (myRoute.steps[i].instructions.indexOf("on the right") !== -1 ||
         myRoute.steps[i].instructions.indexOf("on the left") !== -1) {
            if (myRoute.steps[i].instructions.indexOf("head") !== -1) {
                yonArray.push("ileri *");
            }
            else if (myRoute.steps[i].instructions.indexOf("left") !== -1) {
                yonArray.push("sola *");
            }
            else if (myRoute.steps[i].instructions.indexOf("right") !== -1) {
                yonArray.push("sağa *");
            }
            else {
                yonArray.push("ileri *");
            }
        }
        else if (myRoute.steps[i].instructions.indexOf("right") !== -1 ||
        myRoute.steps[i].instructions.indexOf("sağa") !== -1) {
            yonArray.push("sağa *");
        }
        else if (myRoute.steps[i].instructions.indexOf("left") !== -1 ||
        myRoute.steps[i].instructions.indexOf("sola") !== -1) {
            yonArray.push("sola *");
        }
        else if (myRoute.steps[i].instructions.indexOf("head") !== -1 ||
        myRoute.steps[i].instructions.indexOf("devam edin") !== -1 ||
        myRoute.steps[i].instructions.indexOf("continue straight onto") !== -1 ||
        myRoute.steps[i].instructions.indexOf("ilerleyin") !== -1 ||
        myRoute.steps[i].instructions.indexOf("continue onto") !== -1) {
            yonArray.push("ileri *");
        }
        else if (myRoute.steps[i].instructions.indexOf("vapuru") !== -1 ||
        myRoute.steps[i].instructions.indexOf("feribotuna") !== -1) {
            yonArray.push("tanımsız *");
            alert("'" + myRoute.steps[i].instructions + "'  " + "ifadesinde deniz yolu seçili,
             lütfen mode of travel kısmından yolculuk biçimini değiştiriniz.");
        }
        else if (myRoute.steps[i].instructions.indexOf("döner kavşaktan") !== -1 ||
        myRoute.steps[i].instructions.indexOf("döner kavşağından") !== -1) {
            yonArray.push("ileri *");
            alert("'" + myRoute.steps[i].instructions + " ' " + "ifadesinde döner kavşak geçmektedir,
             ileri komutu verilecek...");
        }
        else if (myRoute.steps[i].instructions.indexOf("çıkışına girin") !== -1) {
            yonArray.push("ileri *");
            alert("'" + myRoute.steps[i].instructions + " ' " + "ifadesinde çıkışına girin geçmektedir
            , ileri komutu verilecek...");
        }
        else {
            alert(myRoute.steps[i].instructions + "ifadesinde tanımlı yön bulunmamaktadır,
            yorumlayıp ekleyiniz");
            yonArray.push("tanımsız *");
        }
    }
```

Şekil 3.6 “Google Maps Directions” servisinden veri çağıran javascript kodu

3.1.4 Simulasyon İçin Ara Noktaların Hesaplanması Ve Harita Üzerinde Gösterimi

Tarifi bilinen yolun, bizim belirlediğimiz simulasyon noktaları arasını istenen sayıda noktayla doldurarak, harita üzerinde gösterimi sağlanmıştır. Sonrasında bu noktalar algoritmaya yapay olarak oluşturulmuş bir istikamet girdisi olarak verilmiş ve sonucunda algoritmanın davranışı test edilmiştir. İstenilen sıklıkta, seçilen iki noktanın arasını dolduran kod parçacığı Şekil 3.7'de gösterilmiştir.

```
        void EnlemBoylamUret()
        {
            int orneklemeSayisi = Convert.ToInt16(txt_ornek.Text);
                                  .
                                  .
                                  .
araLatlar = (double.Parse(SimLat[dolsay + 1], CultureInfo.InvariantCulture)
    - double.Parse(SimLat[dolsay], CultureInfo.InvariantCulture)) / orneklemeSayisi;
araLonlar = (double.Parse(SimLon[dolsay + 1], CultureInfo.InvariantCulture)
    - double.Parse(SimLon[dolsay], CultureInfo.InvariantCulture)) / orneklemeSayisi;
                                  .
                                  .
   SimLat[dolsay] = (double.Parse(SimLat[dolsay], CultureInfo.InvariantCulture)
       + araLatlar).ToString();
   SimLon[dolsay] = (double.Parse(SimLon[dolsay], CultureInfo.InvariantCulture)
       + araLatlar).ToString();
```

Şekil 3.7 Şeçilen noktaların arasının dolduran C# kodu

Bu noktaların harita üzerinde gösterimini sağlayan kod parçacığı ise Şekil 3.8'de gösterilmiştir. Elle ara noktaları seçerken farenin sağ tuşuna basmak suretiyle bir simge oluşturan kod parçacığı ise Şekil 3.9'da gösterilmiştir.

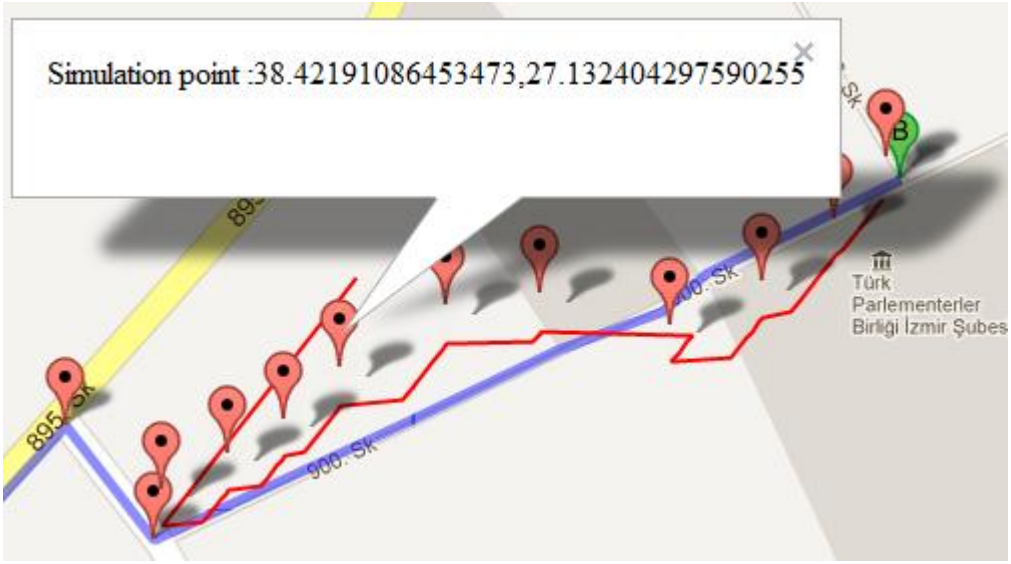
```
function YolCiz() {
    try {
        var flightPlanCoordinates = eval(document.getElementById("hdnCoords").value);
        var flightPath = new google.maps.Polyline({
            path: flightPlanCoordinates,
            strokeColor: "#FF0000",
            strokeOpacity: 1.0,
            strokeWeight: 2
        });
        flightPath.setMap(map);
    } catch (e) {
        alert("GPS Modül bağlantısını kontrol ediniz.");
        return;
    }
}
```

Şekil 3.8 Noktaların harita üzerinde gösteren javascript kodu

```
google.maps.event.addListener(map, 'rightclick', function (event) {
    SagTikGetir(event.latLng);
});


function SagTikGetir(location) {
    var marker = new google.maps.Marker({
        position: location,
        map: map
    });
    marker.setAnimation(google.maps.Animation.BOUNCE);
    attachInstructionText(marker, "Simulation point :");
    SimLatLon.push(marker.getPosition());
}
```

Şekil 3.9 Farenin sağ tuşuna basmak suretiyle yapay seyir noktası üreten javascript kodu



Şekil 3.10 Elle verilmiş simulasyon noktalarının arası 10 örnekle doldurulmuş ve üzerine tıklandığında konum bilgisi vermesi sağlanmıştır

3.1.5 Hedef Nokta Belirleme ve Simgelere Konum Bilgisi Eklenmesi

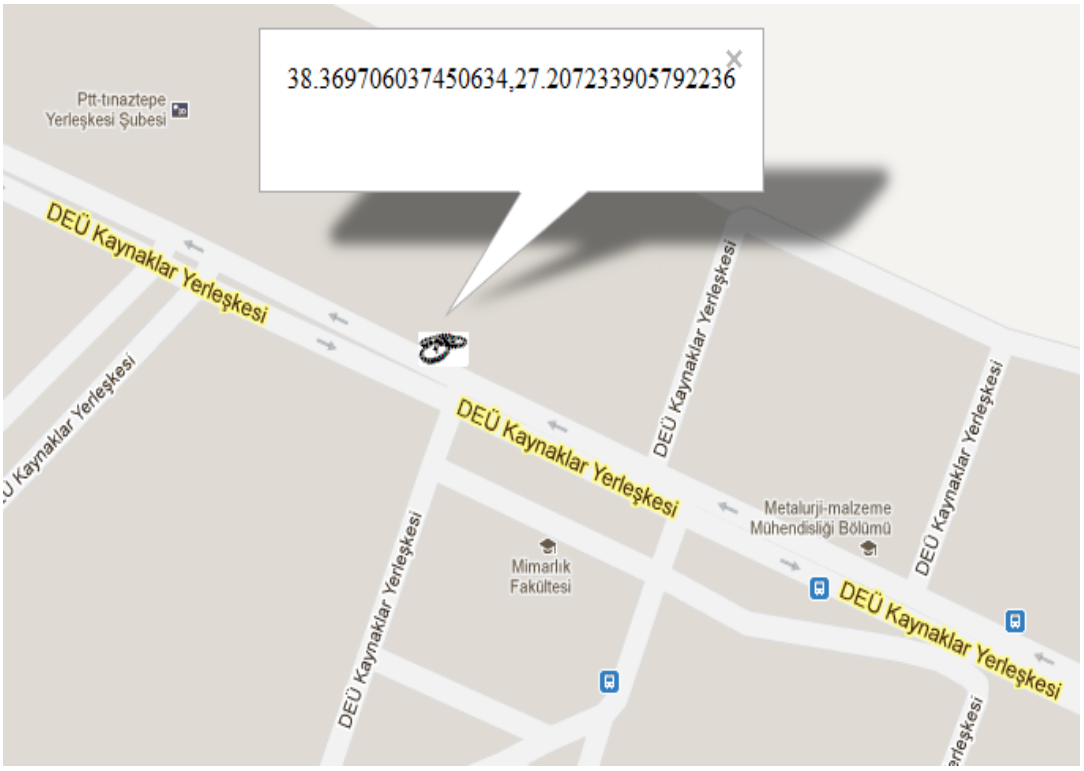
Hedef noktanın belirlenmesi için "Google Maps" kütüphanesinin bazı hareketleri dinleyen metodları kullanılmıştır. Şekil 3.10'da farenin çift tıklanması ile simge ekleyen ve bu simgeye tıklamak suretiyle ilgili konum bilgisini görmeyi mümkün kılan kod parçacığı gösterilmektedir. Konum bilgisi kopyalanıp ilgili hedef yazı kutusuna yapıştırılarak "Harita" butonuna basılır ve yol bilgisi gösterilir. Farenin sol tuşuna çift tıklayarak veya sağ tuşuna tıklayarak konum bilgisine ulaşılabilir. Aralarındaki fark ise, simülasyonda sağ tıklama arka tarafta javascript kodunda bir katara bilgi aktarmaktadır.

```
google.maps.event.addListener(map, 'dblclick', function (event) {
    AnlikIcon(event.latLng);
});

function AnlikIcon(ikonlar) {
    var marker1 = new google.maps.Marker({
        position: ikonlar,
        map: map,
        icon: g_url
    });
    marker1.setMap(map);
    attachInstructionText(marker1, "");
}

function attachInstructionText(marker, text) {
    google.maps.event.addListener(marker, 'click', function () {
        stepDisplay.setContent(text + this.position.lat() + "," + this.position.lng());
        stepDisplay.open(map, marker);
    });
}
```

Şekil 3.11 Hedef nokta belirleyen ve simgelere konum bilgisi ekleyen javascript kodu



Şekil 3.12 Harita üzerinde hedef nokta için oluşturulmuş simge ve konum bilgisi

3.1.6 Elle Oluşturulmuş Konum Bilgilerinin Sisteme Girişi

Fonksiyonla oluşturulmuş noktalar, zamanlayıcı aracı ile belli zaman aralıklarında "RichTextBox" isimli zengin metin kutusu aracına yazılmaktadır.

```
void CurrentTexteYaz()
{
    txt_currentpoint.Text = simcoords1Lat[currentTextYazSayac] + "," + simcoords1Lon[currentTextYazSayac];

    if (currentTextYazSayac == simcoords1Lat.Length - 1)
    {
        timer2.Stop();
    }
    currentTextYazSayac++;
}
```

Şekil 3.13 Konum bilgilerinin metin kutusuna yazdırılması C# kodu

Sistem bir taraftan belli zaman aralıklarında bu verileri yazarken bir taraftan da simülasyonun aşama, hedef kontrolü, açı ve tork hesabı gibi fonksiyonlarını çalıştırmaktadır. Buna olanak yaratmak için "BackGroundWorker" aracı kullanılmıştır.

"BackGroundWorker" simülasyonun temel iş parçacığını üstlenmektedir. Anlık konum değerleri bu iş parçacığı çalışmaya başladığında alınmaktadır.

```
AnlikX = double.Parse(simcoords1Lat[QCThreadSayac],CultureInfo.InvariantCulture);
AnlikY = double.Parse(simcoords1Lon[QCThreadSayac],CultureInfo.InvariantCulture);
```

Şekil 3.14 Anlık konum bilgisi C# kodu

3.1.7 Hedef Nokta Tespiti

Google Maps servisinden ara adımlar çekilip "DataGridView" aracına aktarıldıktan sonra son değer olarak hedef nokta konumu eklenmiştir. Böylece hedef nokta son ara adım olarak değerlendirilmiştir. Hedef noktanın anlık noktayla kıyaslanması sonucu programı bitirmek ya da devam etmek üzere doğru ya da yanlış bilgisi döndürülmüştür.

```
public bool HedefNoktaMi()
{
    if ((Math.Abs(AsamaX[AsamaX.Length - 1] - AnlikX) < 0.00003) &&
        (Math.Abs(AsamaY[AsamaY.Length - 1] - AnlikY) < 0.00003))
    {
        return true;
    }
    else
    {
        return false;
    }
}
```

Şekil 3.15 Hedef nokta tespiti C# kodu

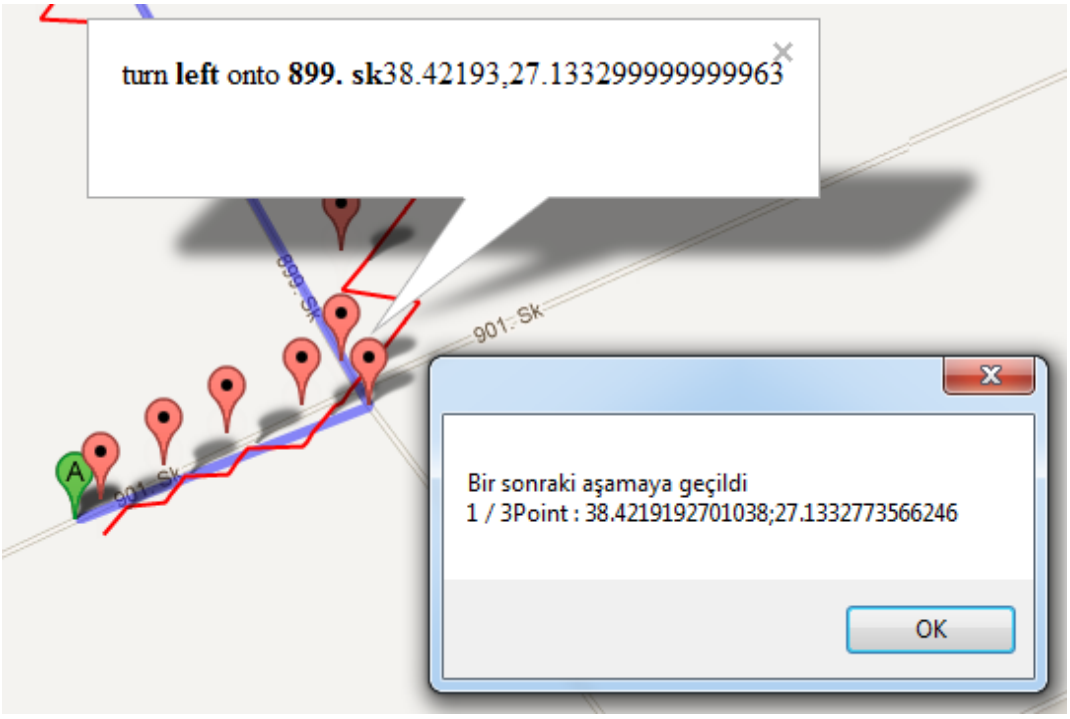
3.1.8 Bir Sonraki Ara Adımın Tespiti

İHA'nın anlık konumu ile sırasıyla "DataGridView" listesinden alınan bir sonraki aşamanın konumu karşılaştırılmaktadır. Dönüş açısı hesabı için belirli miktarda uzaklık payı bırakılmıştır ve bu değer isteğe bağlı olarak değiştirilebilmektedir.

```
public bool BirSonrakiAsamaMi()
{
    if ((Math.Abs(AsamaX[KacinciAsama + 1] - AnlikX) < 0.00003) &&
        (Math.Abs(AsamaY[KacinciAsama + 1] - AnlikY) < 0.00003))
    {
        KacinciAsama = KacinciAsama + 1;
        return true;
    }
    else
    {
        return false;
    }

}
```

Şekil 3.16 Bir sonraki aşama tespiti C# kodu



Şekil 3.17 Simülasyonda ara adım geçişinden bir kesit

3.1.9 Mesafe Ölçümü

Bölüm 2.6'da da bahsedildiği üzere iki nokta arasındaki uzaklık Haversine formülü kullanılarak hesaplanmıştır. Ölçülen değerler uzak ve yakın mesafeler için Google harita cetveline yakın ve tutarlı değerler sunmaktadır.

```
public double Mesafe(double lat1, double lat2, double lon1, double lon2)
{

    int R = 6371; // km (change this constant to get miles)
    var dLat = (lat2 - lat1) * Math.PI / 180;
    var dLon = (lon2 - lon1) * Math.PI / 180;
    var a = Math.Sin(dLat / 2) * Math.Sin(dLat / 2) +
        Math.Cos(lat1 * Math.PI / 180) * Math.Cos(lat2 * Math.PI / 180) *
        Math.Sin(dLon / 2) * Math.Sin(dLon / 2);
    var c = 2 * Math.Atan2(Math.Sqrt(a), Math.Sqrt(1 - a));
    var d = R * c;
    if (d > 1)
    {
        txt_mesafe.Text = Math.Round(d) + "km";
        return d * 1000;
    }
    else
    {
        txt_mesafe.Text = Math.Round(d * 1000) + "m";
        return d * 1000;
    }
}
```

Şekil 3.18 Haversine formülü ile mesafe ölçümü C# kodu

3.1.10 İki Ara Adım Arası Doğru Takibi

Bölüm 2.2'de belirtildiği gibi İHA'nın bir önceki ve bir sonraki ara adımlar arasındaki doğruyu takibi için, öncelikle bu ara adımların enlem ve boylam farkları 0 değerine göre karşılaştırılmıştır. Bu şekilde bölge tespitinin yapılmasının ardından İHA'nın anlık enlem değeri denklemde yerine konulmuştur. Sonra İHA'nın anlık boylam değerinden denklem sonucunda ulaşılmış boylam değeri çıkartılarak 0 değerine göre karşılaştırılmıştır.

```
public int GenelAciHesap()
{
    if ((AsamaX[KacinciAsama + 1] - AsamaX[KacinciAsama] >= 0) &&
        (AsamaY[KacinciAsama + 1] - AsamaY[KacinciAsama] > 0))
    {
        if (AnlikY - ((((AsamaY[KacinciAsama + 1] - AsamaY[KacinciAsama]) /
            (AsamaX[KacinciAsama + 1] - AsamaX[KacinciAsama])) *
            (AnlikX - AsamaX[KacinciAsama])) + AsamaY[KacinciAsama]) < 0)
        {
            return sagdakiMotor;
        }
        else
        {
            return soldakiMotor;
        }
    }
    else if ((AsamaX[KacinciAsama + 1] - AsamaX[KacinciAsama] > 0) &&
        (AsamaY[KacinciAsama + 1] - AsamaY[KacinciAsama] < 0))
    {
        if (AnlikY - ((((AsamaY[KacinciAsama + 1] - AsamaY[KacinciAsama]) /
            (AsamaX[KacinciAsama + 1] - AsamaX[KacinciAsama])) *
            (AnlikX - AsamaX[KacinciAsama])) + AsamaY[KacinciAsama]) < 0)
        {
            return sagdakiMotor;
        }
        else
        {
            return soldakiMotor;
        }
    }
    else if ((AsamaX[KacinciAsama + 1] - AsamaX[KacinciAsama] < 0) &&
        (AsamaY[KacinciAsama + 1] - AsamaY[KacinciAsama] < 0))
    {
        if (AnlikY - ((((AsamaY[KacinciAsama + 1] - AsamaY[KacinciAsama]) /
            (AsamaX[KacinciAsama + 1] - AsamaX[KacinciAsama])) *
            (AnlikX - AsamaX[KacinciAsama])) + AsamaY[KacinciAsama]) < 0)
        {
            return soldakiMotor;
        }
        else
        {
            return sagdakiMotor;
        }
    }
    else if ((AsamaX[KacinciAsama + 1] - AsamaX[KacinciAsama] < 0) &&
        (AsamaY[KacinciAsama + 1] - AsamaY[KacinciAsama] > 0))
    {
        if (AnlikY - ((((AsamaY[KacinciAsama + 1] - AsamaY[KacinciAsama]) /
            (AsamaX[KacinciAsama + 1] - AsamaX[KacinciAsama])) *
            (AnlikX - AsamaX[KacinciAsama])) + AsamaY[KacinciAsama]) < 0)
        {
            return soldakiMotor;
        }
        else
        {
            return sagdakiMotor;
```

Şekil 3.19 İHA'nın iki ara adım arası doğru takibi C# kodu

3.1.11 Ara Adımlarda Dönüş Açısı Hesaplama

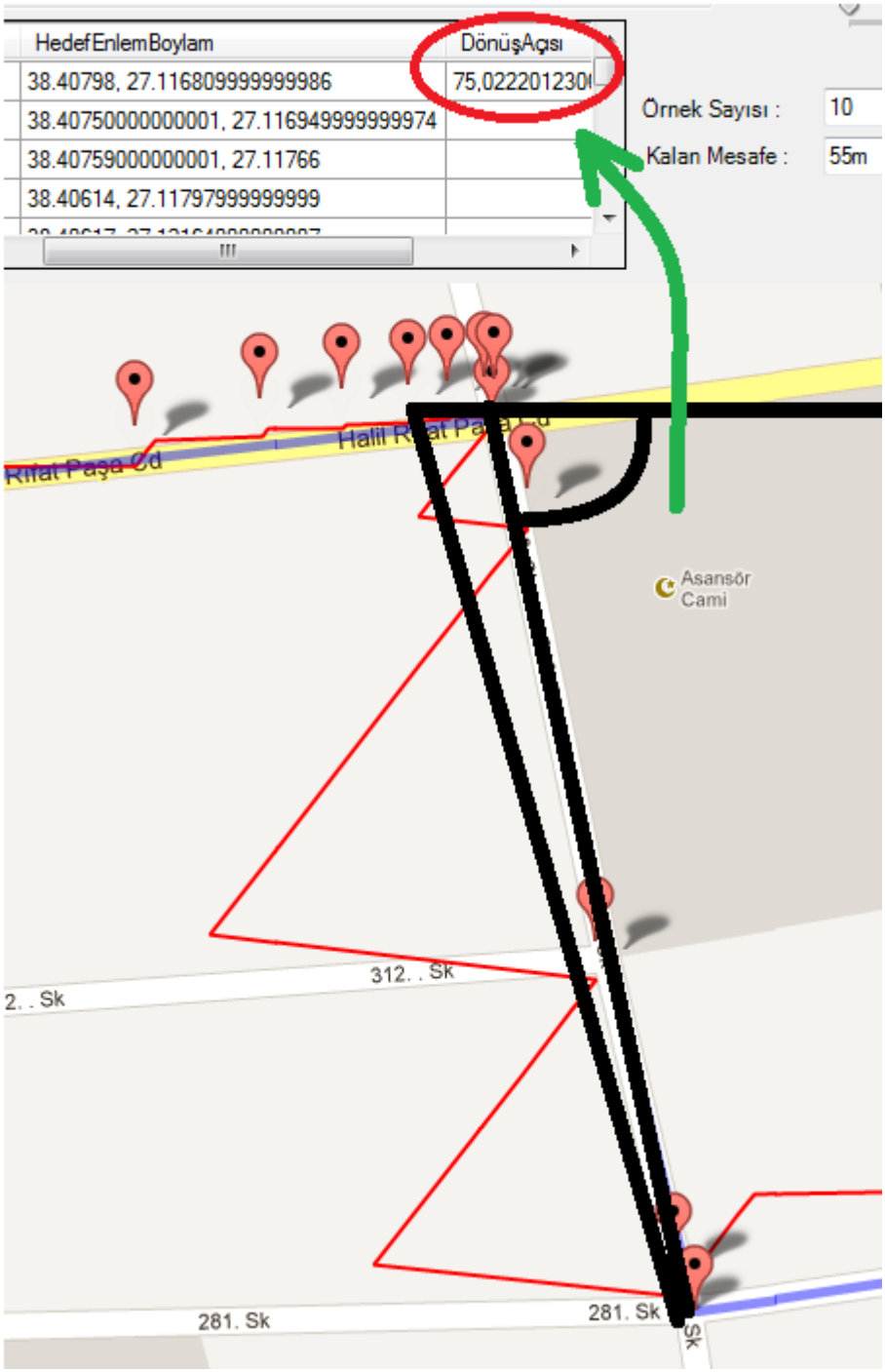
Bölüm 2.2'de bahsi geçen kosinüs teoremi dönüşlerdeki açıları hesaplamak için kullanılmıştır. Bir sonraki ara adıma belirlenen mesafe yaklaşan İHA; kendisi, dönüş noktası ve bir sonraki aşama ile bir üçgen oluşturmaktadır. İHA, Resim 2.2'deki C dış açısı kadar dönecektir. Bu açıyı hesaplayan kod parçacığı Şekil 3.20'de gösterilmiştir.

```
public double AsamaAciHesap()
{
    double aKenari;
    double bKenari;
    double cKenari;
    double DonusAcisi;

    aKenari = Mesafe(AnlikX, AsamaX[KacinciAsama], AnlikY, AsamaY[KacinciAsama]);
    bKenari = Mesafe(AsamaX[KacinciAsama], AsamaX[KacinciAsama + 1], AsamaY[KacinciAsama],
        AsamaY[KacinciAsama + 1]);
    cKenari = Mesafe(AnlikX, AsamaX[KacinciAsama + 1], AnlikY, AsamaY[KacinciAsama + 1]);
    DonusAcisi = Math.Acos((-(aKenari * aKenari) + (bKenari * bKenari) + (cKenari * cKenari))
    /(2 * bKenari * cKenari)) + Math.Acos(((aKenari * aKenari) - (bKenari * bKenari) +
        (cKenari * cKenari)) / (2 * aKenari * cKenari));
    double AlternatifAci = Math.Acos(((aKenari * aKenari) + (bKenari * bKenari) -
        (cKenari * cKenari)) / (2 * bKenari * aKenari));

    return DonusAcisi * (180.0 / Math.PI);
}
```

Şekil 3.20 Dönüş açısını hesaplayan C# kodu

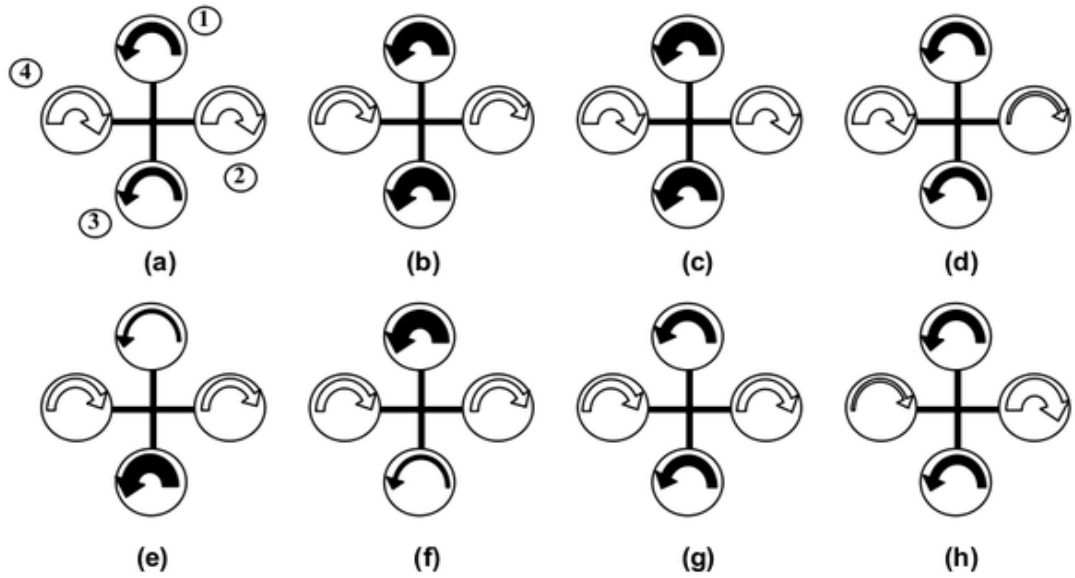


Şekil 3.21 Simulasyonda dönüş açısına bir örnek

BÖLÜM DÖRT
İNSANSIZ HAVA ARACININ MODELLENMESİ

Darbe genlikli modülasyon sinyalleri motorlara giriş olarak verildiğinde motorların açısal hızları çıkış olarak gözlemlenmektedir. Bu açısal hızlar ise pervaneler için bir giriş değeridir. Pervanelerin açısal hızlara bağlı olarak değişen kaldırma kuvveti ise pervanelerin çıkış değeri olarak ele alınmaktadır. Dört pervanenin değişen kaldırma kuvvetleri üç eksende açı değişimi ve yer değiştirme yaratmaktadır.

İnsansız hava aracının tork dengesi amacıyla karşılıklı pervaneleri aynı yönde, komşu pervaneleri ters yönde dönmektedir. Ön ve arka pervanelerin saat yönünün tersine, sağ ve sol pervanelerin saat yönünde döndüğünü kabul edersek İHA'nın yapacağı hareketler Şekil 4.1'de gösterilmektedir.

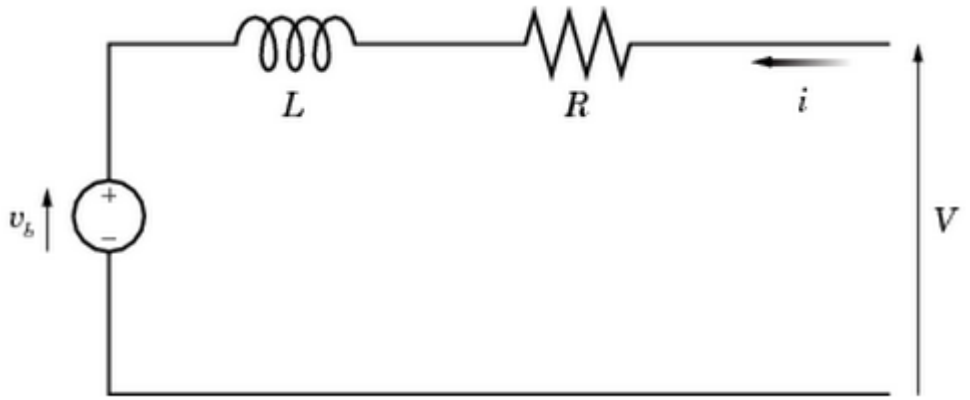


(a)	Sapma (saat tersi yönü)	(e)	Yunuslama (saat tersi yönü)
(b)	Sapma (saat yönü)	(f)	Yunuslama (saat yönü)
(c)	Havalanma	(g)	İniş
(d)	Yalpa (saat yönü)	(h)	Yalpa (saat yönü tersi)

Şekil 4.1 İHA'nın pervane açısal hızlarına göre hareket şekli

4.1 DC Motor Modeli

DC motorun benzetimi için Simulink DC Motor bloğu kullanılmıştır. Blok Şekil 4.2'deki eşdeğer devre modelinin elektriksel ve tork karakteristiklerini ifade etmektedir.



Şekil 4.2 DC motor eşdeğer devre modeli

R direnci armatür direncini, endüktans L armatür endüktansını tanımlamaktadır. Motordaki doğal mıknatıs, armatürdeki zıt elektromotor kuvveti v_b 'ye sebep olacaktır.

$$v_b = k_v w \tag{12}$$

Burada k_v zıt elektromotor kuvveti sabiti, w açısal hızdır. Mototrun ürettiği tork motor akımıyla doğru orantılıdır.

$$T_E = k_t i \tag{13}$$

Burada k_t tork sabitidir. DC Motor bloğu elektromanyetik kayıpları ihmal etmiştir. Bu da armatürdeki zıt elektromotor kuvvetinin harcadığı elektriksel gücün mekanik güce eşdeğer olması demektir.

$$T_E w = v_b i \tag{14}$$

$$k_t i w = k_v w i \tag{15}$$

$$k_v = k_t \tag{16}$$

Sonuç olarak blokta k_v veya k_t değeri belirlenebilir. Bloğun 'Model parameterization' parametresi 'By stall torque & no-load speed' veya 'By rated power, rated speed & no-load speed' seçildiğinde blok eşdeğer devre parametrelerini aşağıdaki gibi çözecektir. Kararlı durum tork-hız ilişkisinde L değeri etkisizdir.

$$i = \frac{V - v_b}{R} = \frac{V - k_v w}{R} \tag{15}$$

Bulunan i değeri tork denkleminde yerine yazılırsa,

$$T_E = \frac{k_t}{R}(V - k_v w) \tag{16}$$

elde edilir. Motor eylemsizliği ve sönümlenmesi eklenirse,

$$T = \frac{k_t}{R}(V - k_v w) - J\dot{w} - \lambda w \tag{17}$$

Yüksüz koşulda elektriksel olarak üretilmiş mekanik tork, rotor sönümlenme torkuna eşit olmalıdır.

$$k_t i_{noload} = \lambda w_{noload} \tag{18}$$

Burada i_{noload} yüksüz durumdaki akımdır. 'By no-load current' seçeneği seçildiğinde bu denklem tork-hız denklemine ek olarak, λ ve diğer denklem sabitleri belirlemekte kullanılır.

4.1.1 Kontrollü Darbe Genlik Modülasyonu Sinyali ve H Köprüsü

Sistemde kumandadan gelen komutlar vasıtasıyla alıcı, motoru darbe genlik modülasyonu sinyalleriyle kontrol eder. Bu sinyaller elektronik hız devresinin

girişini oluşturmaktadır. Çıkışı ise motora bağlıdır. Benzetimde Simulink'in kontrollü darbe genlik modülasyonu ve H köprüsü bloğu kullanılmıştır.

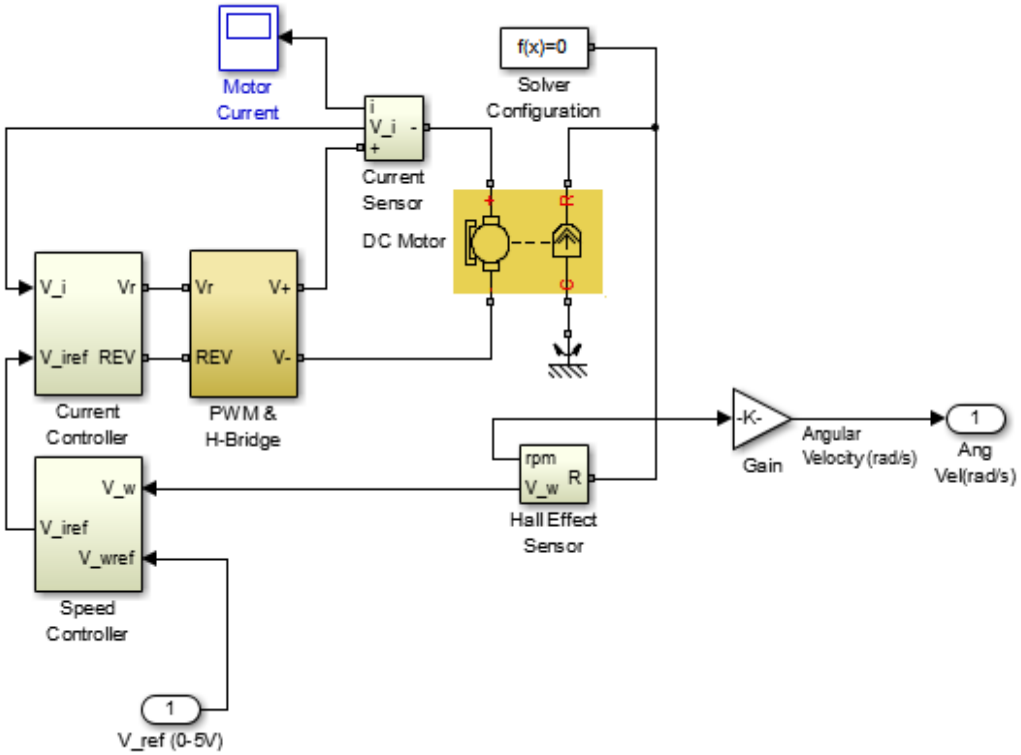
Kontrollü darbe genlik modülasyonu gerilim bloğu Vref referans gerilimne bağlı olarak gerilim üretir. Vmin en düşük referans gerilim, Vmax en yüksek referans gerilimidir.

$$100 * \frac{V_{ref} - V_{min}}{V_{max} - V_{min}} percent \quad (19)$$

H köprüsü bloğu H köprüsü motor sürücüsünü temsil etmektedir. Eğer giriş sinyali 'Enable threshold voltage' parametresinden büyükse istenilen genlikte gerilim çıkışı verir.

4.1.2 DC Motorun Hız Kontrolü

DC motordan alan etkili sensör vasıtasıyla alınan hız bilgisi bir alçak geçirgen filtreye tabi tutulmuş, doğrusal bir mantık ile gerilim değerine dönüştürülmüş akabinde istenilen gerilim değeri ile oran-integral kontrolüne tabi tutulmuştur. 'Linear Electric Actuator (System-Level Model)' Simulink modeli baz alınmıştır.



Şekil 4.3 DC motor, sürücü ve kontrol devresi

4.2 Pervane Modeli

Pervane modellenirken havanın sürtünme kuvveti gibi aerodinamik kuvvetler, kanatların savrulması ve yer etkisi ihmal edilmiştir. Herbir pervanenin açısal hızlarının karesiyle orantılı itki kuvveti F_i ve momenti M_i aşağıdaki denklemlerde gösterilmiştir.

$$F_i = K_T w_i^2 \tag{20}$$

$$M_i = K_M w_i^2 \tag{21}$$

Buradaki K_T ve K_M katsayıları aşağıda gösterildiği gibi bulunmaktadır.

$$K_T = C_T \frac{\rho D^4}{4\pi^2} \tag{22}$$

$$K_M = C_P \frac{\rho D^5}{8\pi^2} \tag{23}$$

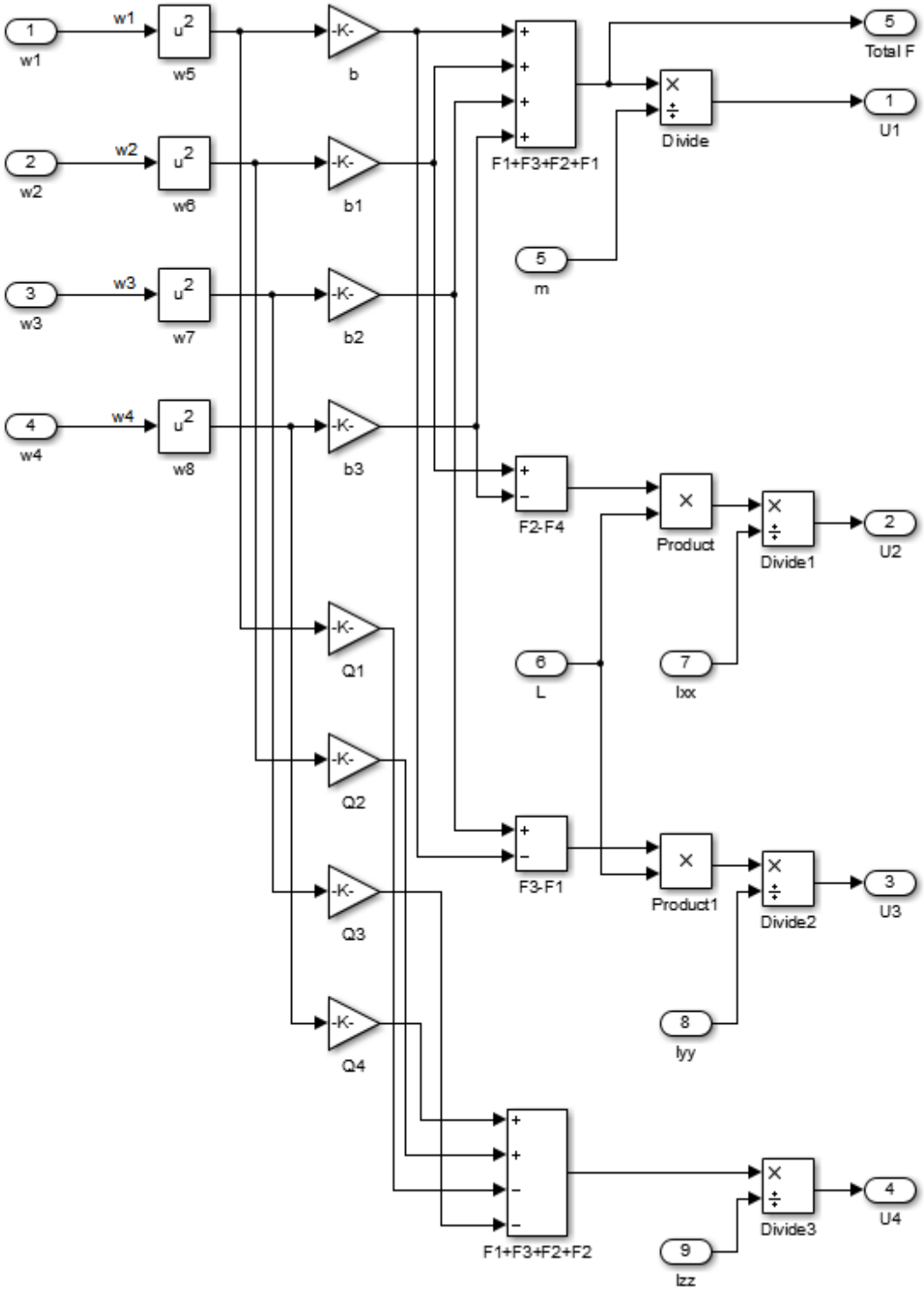
$$M_x = (w_2^2 - w_4^2)K_T\mathrm{d} = (F_2 - F_4)d \quad (24)$$

$$M_y = (w_1^2 - w_3^2)K_T d = (F_1 - F_3)d \quad (25)$$

$$M_z = (w_1^2 + w_3^2 - w_2^2 - w_4^2)K_M = (F_1 + F_3 - F_2 - F_4)K \quad (26)$$

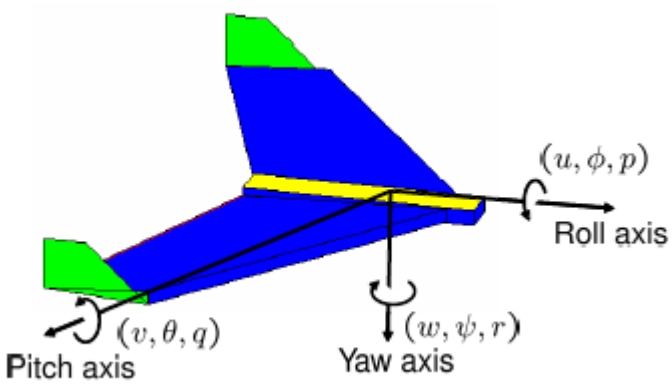
Burada d pervane merkezinin İHA merkez noktasına uzaklığı, ρ hava yoğunluğu, C_T itki kuvveti katsayısı ve C_P güç katsayısı, K = $\frac{K_M}{K_T}$ 'dir.

Pervane Simulink Modeli aşağıda gösterilmiştir.



Şekil 4.4 Pervane Simulink Modeli

4.3 Kinematik Ve Dinamik Hesaplar



Şekil 4.5 Eksen tanımları

Şekil 4.5'te durum değişkenleri gösterilmiştir. İHA'nın pozisyon değerleri pn, pe, h (sırasıyla kuzey atalet pozisyonu, doğu atalet pozisyonu, aracın yüksekliği) atalet çerçevesinde verilmiş ve h değeri negatif Z ekseninde tanılanmıştır. İHA'nın hızları (u,v,w) ve açısal hızları (p,q,r) gövde çerçevesine riayet edilerek verilmiştir.

4.3.1 Kinematik Hesaplar

u, v ve w gövde çerçevesi değerleri iken pn, pe ve –h atalet çerçeve değerleridir. Pozisyon ve hız arasındaki ilişki aşağıdaki gibi verilmiştir.

$$\frac{d}{dt}\begin{pmatrix} p_n \\ p_e \\ -h \end{pmatrix} = R_b^v \begin{pmatrix} u \\ v \\ w \end{pmatrix} = (R_v^b)^T \begin{pmatrix} u \\ v \\ w \end{pmatrix} =$$

$$\begin{pmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{pmatrix} \begin{pmatrix} u \\ v \\ w \end{pmatrix} \quad (27)$$

Açısal hızlar gövde çerçevesindeyken yalpalama açısı ϕ F_{v2}, yunuslama açısı θ F_{v1} ve yönelme açısı ψ F_v çerçevesindedir. p,q ve r ile $\dot{\phi}$, $\dot{\theta}$ ve $\dot{\psi}$ arasında ilişki kurulmalıdır. $\dot{\phi}$, $\dot{\theta}$ ve $\dot{\psi}$ küçük olmasını ve aşağıdaki durumu gözeterek,

$$R_{v2}^b(\dot{\phi}) = R_{v1}^{v2}(\dot{\theta}) = R_v^{v1}(\dot{\psi}) = I \quad (28)$$

$$\begin{pmatrix} p \\ q \\ r \end{pmatrix} = R_{v2}^{b}(\dot{\phi}) \begin{pmatrix} \dot{\phi} \\ 0 \\ 0 \end{pmatrix} + R_{v2}^{b}(\phi) R_{v1}^{v2}(\dot{\theta}) \begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix} +$$

$$R_{v2}^{b}(\phi) R_{v1}^{v2}(\theta) R_{v\to v1}(\dot{\psi}) \begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix} \quad (29)$$

$$= \begin{pmatrix} \dot{\phi} \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 \\ 0 & c\phi & s\phi \\ 0 & -s\phi & c\phi \end{pmatrix} \begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 \\ 0 & c\phi & s\phi \\ 0 & -s\phi & c\phi \end{pmatrix} \begin{pmatrix} c\theta & 0 & -s\theta \\ 0 & 1 & 0 \\ s\theta & 0 & c\theta \end{pmatrix} \quad (30)$$

$$= \begin{pmatrix} 1 & 0 & -s\theta \\ 0 & c\phi & s\phi c\theta \\ 0 & -s\phi & c\phi c\theta \end{pmatrix} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} \quad (31)$$

$$\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{pmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi)\sec(\theta) & \cos(\phi)\sec(\theta) \end{pmatrix} \begin{pmatrix} p \\ q \\ r \end{pmatrix} \quad (32)$$

elde edilir.

4.3.2 Rigid Body Dinamikleri

v İHA'nın hızı kabul edilsin. Newton yasalarına göre,

$$\mathrm{m}\frac{\mathrm{dv}}{\mathrm{dt_i}} = \mathrm{m}\left(\frac{\mathrm{dv}}{\mathrm{dt_b}} + \mathrm{w_{b/i}x\,v}\right) = \mathrm{f} \quad (33)$$

Burada $w_{b/i}$atalet çerçevesine bağlı hava çerçevesinin açısal hızıdır.

$$\mathrm{v^b} \underset{=}{\Delta} (\mathrm{u,v,w})^{\mathrm{T}} \text{ve } \mathrm{w_{b/i}^b} \underset{=}{\Delta} (\mathrm{p,q,r})^{\mathrm{T}} \quad (34)$$

kabul edilirse,

$$\begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix} = \begin{pmatrix} rv-qw \\ pw-ru \\ qu-pv \end{pmatrix} + \frac{1}{m} \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} \quad (35)$$

$$\mathbf{f^b} \underset{=}{\Delta} (\mathrm{f_x, f_y, f_z})^{\mathrm{T}} \quad (36)$$

Dönüş hareketi için Newton'un ikinci yasasından faydalanırsak,

$$\frac{dh^b}{dt_i} = m, \tag{37}$$

yazılabilir. Burada h açısal momentum ve m ise uygulanan torktur. Coriolis denklemi kullanarak

$$\frac{\mathrm{dh}}{\mathrm{dt_i}} = \frac{\mathrm{dh}}{\mathrm{dt_b}} + \mathrm{w_{b/i}\,x\,h = m} \tag{38}$$

$$\mathrm{h^b = Jw^b_{b/i}} \tag{39}$$

$$\mathrm{J} = \begin{pmatrix} \int(y^2+z^2)dm & -\int xydm & -\int xzdm \\ -\int xydm & \int(x^2+z^2)dm & -\int yzdm \\ -\int xzdm & -\int yzdm & \int(x^2+y^2)dm \end{pmatrix} \overset{\Delta}{=} \begin{pmatrix} J_x & -J_{xy} & J_{zz} \\ -J_{xy} & J_y & -J_{yz} \\ -J_{xz} & -J_{yz} & J_z \end{pmatrix} \tag{40}$$

Burada J sabit atalet matrisidir. Jxy = Jxz = Jyz = 0 olduğundan,

$$\mathrm{J} = \begin{pmatrix} J_x & 0 & 0 \\ 0 & J_y & 0 \\ 0 & 0 & J_z \end{pmatrix} \tag{41}$$

elde edilir.

$$\mathrm{J^{-1}} = \begin{pmatrix} 1/J_x & 0 & 0 \\ 0 & 1/J_y & 0 \\ 0 & 0 & 1/J_z \end{pmatrix} \tag{42}$$

Katı cisimlerin ataleti,

$$\mathrm{J} = 2MR^2/5 \tag{43}$$

olarak düşünülürse,

$$J_x = \frac{2MR^2}{5} + 2l^2m \tag{44}$$

$$J_y = \frac{2MR^2}{5} + 2l^2m \tag{45}$$

$$J_z = \frac{2MR^2}{5} + 4l^2m \tag{46}$$

elde edilir.

$$m^b \triangleq (\tau_\phi, \tau_\phi, \tau_\psi)^T \tag{47}$$

tanımıyla,

$$\begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} = \begin{pmatrix} \frac{1}{J_x} & 0 & 0 \\ 0 & \frac{1}{J_y} & 0 \\ 0 & 0 & \frac{1}{J_z} \end{pmatrix} \left[\begin{pmatrix} 0 & r & -q \\ -r & 0 & p \\ q & -p & 0 \end{pmatrix} \begin{pmatrix} J_x & 0 & 0 \\ 0 & J_y & 0 \\ 0 & 0 & J_z \end{pmatrix} \begin{pmatrix} p \\ q \\ r \end{pmatrix} + \begin{pmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{pmatrix} \right] \tag{48}$$

$$= \begin{pmatrix} \frac{J_y - J_z}{J_x} qr \\ \frac{J_z - J_x}{J_y} pr \\ \frac{J_x - J_y}{J_z} pq \end{pmatrix} + \begin{pmatrix} \frac{1}{J_x}\tau_\phi \\ \frac{1}{J_y}\tau_\theta \\ \frac{1}{J_z}\tau_\psi \end{pmatrix} \tag{49}$$

elde edilir. 6 serbestlik dereceli İHA'nın kinematik ve dinamik denklemleri şu şekilde özetlenebilir.

$$\begin{pmatrix} \dot{p}_n \\ \dot{p}_e \\ \dot{h} \end{pmatrix} = \begin{pmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ s\theta & -s\phi c\theta & -c\phi c\theta \end{pmatrix} \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{50}$$

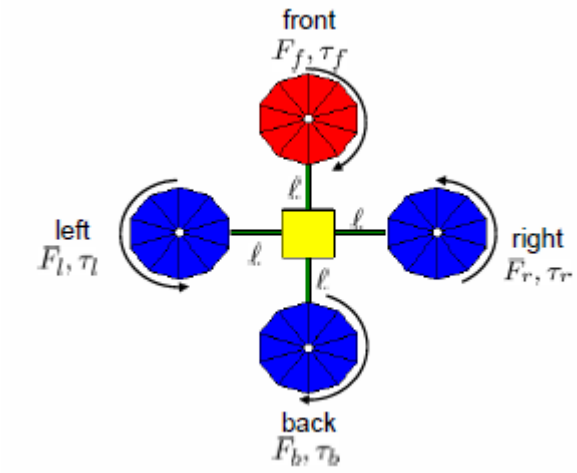
$$\begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix} = \begin{pmatrix} rv - qw \\ pw - ru \\ qu - pv \end{pmatrix} + \frac{1}{m} \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} \tag{51}$$

$$\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{pmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{pmatrix} \begin{pmatrix} p \\ q \\ r \end{pmatrix} \tag{52}$$

$$\begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} = \begin{pmatrix} \frac{J_y - J_z}{J_x} qr \\ \frac{J_z - J_x}{J_y} pr \\ \frac{J_x - J_y}{J_z} pq \end{pmatrix} + \begin{pmatrix} \frac{1}{J_x} \tau_\phi \\ \frac{1}{J_y} \tau_\theta \\ \frac{1}{J_z} \tau_\psi \end{pmatrix} \tag{53}$$

4.3.3 Kuvvetler ve Momentler

İHA'nın her motoru yukarı kuvvet F ve tork τ üretir. İHA'ya etkiyen toplam kuvvetler aşağıdaki gibidir.



Şekil 4.6 İHA'ya yukardan bakış, tork ve kuvvetler

$$F = F_f + F_r + F_b + F_l. \tag{54}$$

Sağ ve sol motorun oluşturduğu kuvvetlerden oluşan yalpalama torku ise aşağıdaki gibidir.

$$\tau_\phi = \ell(F_l - F_r). \tag{55}$$

Benzer şekilde arka ve ön motorların oluşturduğu kuvvetlerden oluşan yunuslama torku aşağıda belirtilmektedir.

$$\tau_\theta = \ell(F_f - F_b). \tag{56}$$

Toplam yönelme torku ise aşağıdaki gibidir.

$$\tau_\psi = \tau_r + \tau_l - \tau_f - \tau_b. \tag{57}$$

Akabinde İHA'nın kuvvet ve torkları matris formunda aşağıdaki gibi yazılmaktadır.

$$\begin{pmatrix} F \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{pmatrix} = \begin{pmatrix} k_1 & k_1 & k_1 & k_1 \\ 0 & -\ell k_1 & 0 & \ell k_1 \\ \ell k_1 & 0 & -\ell k_1 & 0 \\ -k_2 & k_2 & -k_2 & k_2 \end{pmatrix} \begin{pmatrix} \delta_f \\ \delta_r \\ \delta_b \\ \delta_l \end{pmatrix} \triangleq \mathcal{M} \begin{pmatrix} \delta_f \\ \delta_r \\ \delta_b \\ \delta_l \end{pmatrix} \tag{58}$$

Araç çerçevesi F^v'de yer çekiminin kütle merkezine uyguladığı kuvvet,

$$\mathbf{f}_g^v = \begin{pmatrix} 0 \\ 0 \\ mg \end{pmatrix}. \tag{59}$$

şeklindedir. Bunu gövde çerçevesine dönüştürüldüğünde,

$$\begin{aligned} \mathbf{f}_g^b &= R_v^b \begin{pmatrix} 0 \\ 0 \\ mg \end{pmatrix} \\ &= \begin{pmatrix} -mg\sin\theta \\ mg\cos\theta\sin\phi \\ mg\cos\theta\cos\phi \end{pmatrix}. \end{aligned} \tag{60}$$

elde edilmektedir. Sonuç olarak,

$$\begin{pmatrix} \dot{p}_n \\ \dot{p}_e \\ \dot{h} \end{pmatrix} = \begin{pmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ s\theta & -s\phi c\theta & -c\phi c\theta \end{pmatrix} \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{61}$$

$$\begin{pmatrix}\dot{u}\\ \dot{v}\\ \dot{w}\end{pmatrix} = \begin{pmatrix}rv-qw\\ pw-ru\\ qu-pv\end{pmatrix} + \begin{pmatrix}-g\sin\theta\\ g\cos\theta\sin\phi\\ g\cos\theta\cos\phi\end{pmatrix} + \frac{1}{m}\begin{pmatrix}0\\ 0\\ -F\end{pmatrix}, \tag{62}$$

$$\begin{pmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{pmatrix} = \begin{pmatrix}1 & \sin\phi\tan\theta & \cos\phi\tan\theta\\ 0 & \cos\phi & -\sin\phi\\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta}\end{pmatrix}\begin{pmatrix}p\\ q\\ r\end{pmatrix}, \tag{63}$$

$$\begin{pmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{pmatrix} = \begin{pmatrix}\frac{J_y-J_z}{J_x}qr\\ \frac{J_z-J_x}{J_y}pr\\ \frac{J_x-J_y}{J_z}pq\end{pmatrix} + \begin{pmatrix}\frac{1}{J_x}\tau_\phi\\ \frac{1}{J_y}\tau_\theta\\ \frac{1}{J_z}\tau_\psi\end{pmatrix}. \tag{64}$$

denklemleri elde edilmektedir.

4.3.4 Basitleştirilmiş Model

px,py ve pz hedef ve araç arasındaki v1 çerçevesine göre çözülmüş bağıl pozisyon durumudur.

$$\begin{pmatrix}\dot{p}_x\\ \dot{p}_y\\ \dot{p}_z\end{pmatrix} = \begin{pmatrix}c\theta & s\phi s\theta & c\phi s\theta\\ 0 & c\phi & -s\phi\\ -s\theta & s\phi c\theta & c\phi c\theta\end{pmatrix}\begin{pmatrix}u\\ v\\ w\end{pmatrix}. \tag{65}$$

ϕ ve θ değerleri çok küçük olduğundan,

$$\begin{pmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{pmatrix} = \begin{pmatrix}p\\ q\\ r\end{pmatrix}. \tag{66}$$

ifadesi yazılabilir.Aynı şekilde denklem 64 de,

$$\begin{pmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{pmatrix} = \begin{pmatrix}\frac{1}{J_x}\tau_\phi\\ \frac{1}{J_y}\tau_\theta\\ \frac{1}{J_z}\tau_\psi\end{pmatrix}. \tag{67}$$

Şeklinde ifade edilebilir. Bu iki denklem birleştirilirse,

$$\begin{pmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{pmatrix} = \begin{pmatrix} \frac{1}{J_x}\tau_\phi \\ \frac{1}{J_y}\tau_\theta \\ \frac{1}{J_z}\tau_\psi \end{pmatrix}. \tag{68}$$

ortaya çıkar. Denklem 61 $\dot{R}_b^v$ ihmal edilerek çözülürse,

$$\begin{pmatrix} \ddot{p}_n \\ \ddot{p}_e \\ \ddot{p}_d \end{pmatrix} = \begin{pmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{pmatrix} \begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix}. \tag{69}$$

ortaya çıkar. Nihayetinde basitleştirilmiş model aşağıdaki gibidir.

$$\ddot{p}_n = (-\cos\phi \sin\theta \cos\psi - \sin\phi \sin\psi)\frac{F}{m} \tag{70}$$

$$\ddot{p}_e = (-\cos\phi \sin\theta \sin\psi + \sin\phi \cos\psi)\frac{F}{m} \tag{71}$$

$$\ddot{p}_d = g - (\cos\phi \cos\theta)\frac{F}{m} \tag{72}$$

$$\ddot{\phi} = \frac{1}{J_x}\tau_\phi \tag{73}$$

$$\ddot{\theta} = \frac{1}{J_y}\tau_\theta \tag{74}$$

$$\ddot{\psi} = \frac{1}{J_z}\tau_\psi. \tag{75}$$

Denklem 65 R_b^{v1} ihmal edilerek çözülürse,

$$\begin{pmatrix} \ddot{p}_x \\ \ddot{p}_y \\ \ddot{p}_z \end{pmatrix} = \begin{pmatrix} c\theta & s\phi s\theta & c\phi s\theta \\ 0 & c\phi & -s\phi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{pmatrix} \begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix} \tag{76}$$

ortaya çıkmaktadır. Coriolis ifadeleri ihmal edilerek denklem 62, denklem 76'ya dahil edilirse

$$\begin{pmatrix} \ddot{p}_x \\ \ddot{p}_y \\ \ddot{p}_z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} + \begin{pmatrix} -c\phi s\theta \\ s\phi \\ -c\phi c\theta \end{pmatrix} \frac{F}{m}. \tag{77}$$

ortaya çıkmaktadır. Sistemin nihai durum denklemleri ise aşağıdaki gibidir.

$$\ddot{p}_x = -\cos\phi \sin\theta \frac{F}{m} \tag{78}$$

$$\ddot{p}_y = \sin\phi \frac{F}{m} \tag{79}$$

$$\ddot{p}_z = g - \cos\phi \cos\theta \frac{F}{m} \tag{80}$$

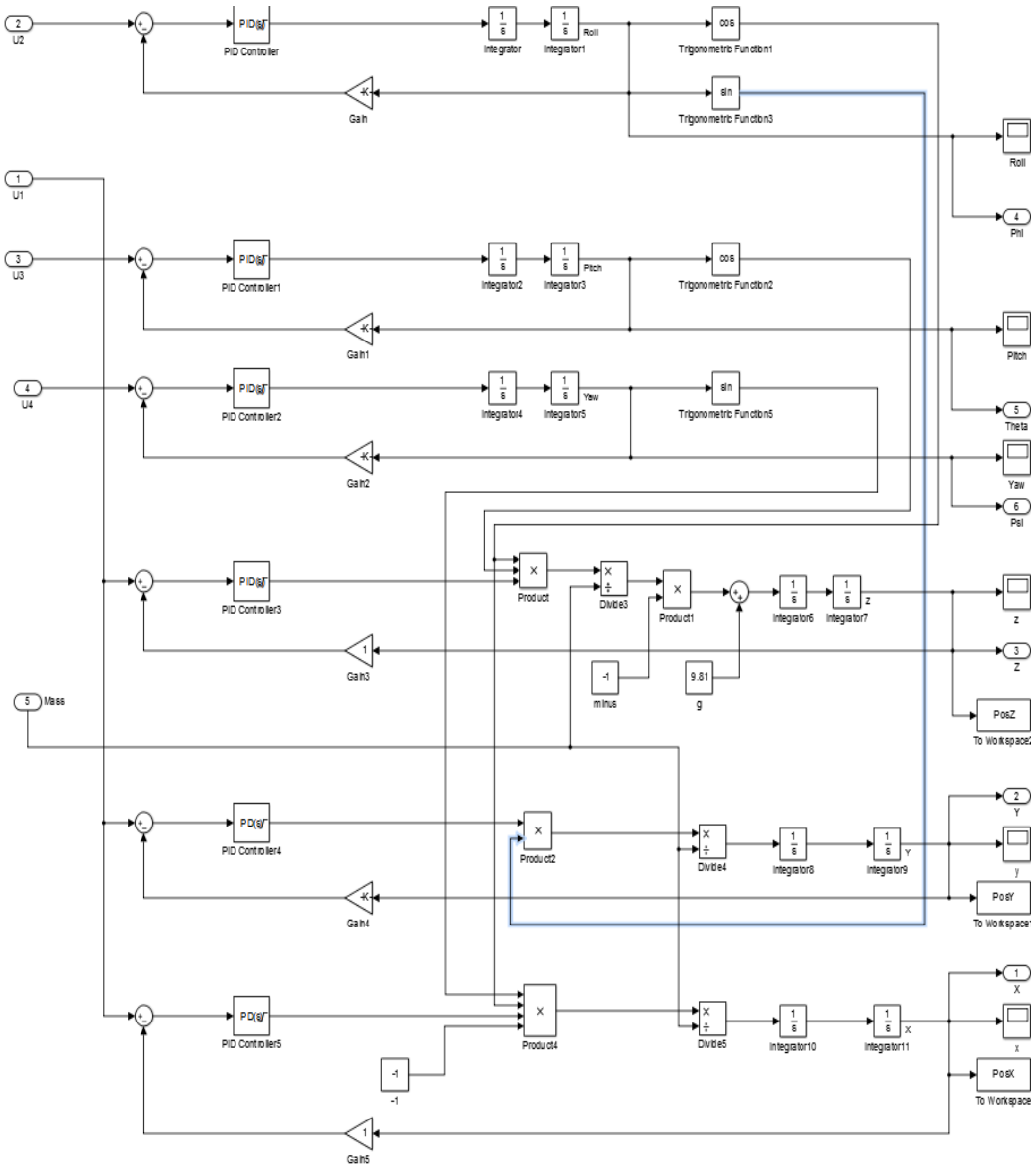
$$\ddot{\phi} = \frac{1}{J_x} \tau_\phi \tag{81}$$

$$\ddot{\theta} = \frac{1}{J_y} \tau_\theta \tag{82}$$

$$\ddot{\psi} = \frac{1}{J_z} \tau_\psi \tag{83}$$

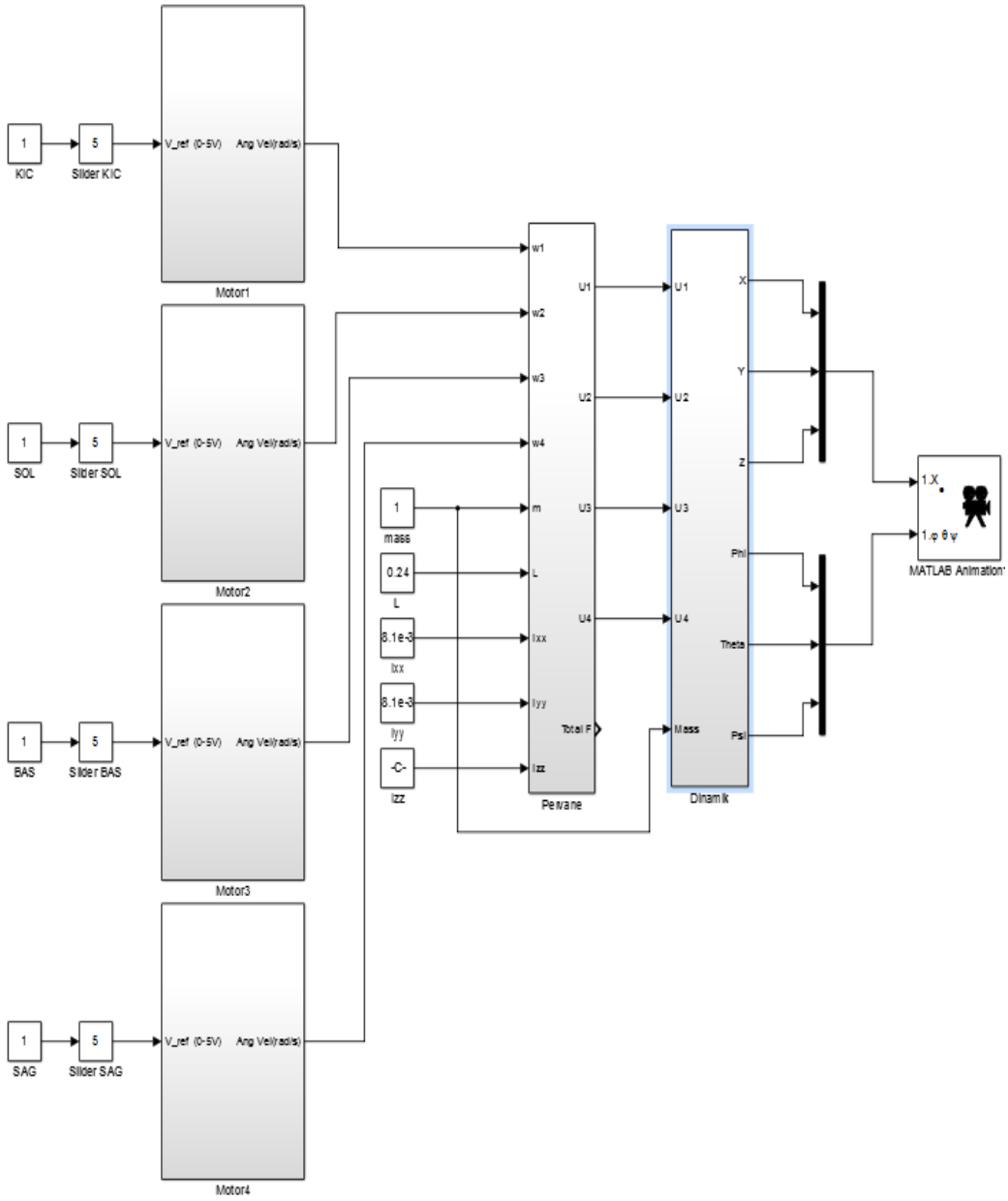
4.4 Modellemenin Benzetimi

Sistemin Simulink'te dinamik benzetimi aşağıdaki gibidir. Yalpalama, yunuslama, yönelme ve Z ekseninde PID kontrol denenmiştir.



Şekil 4.7 Sistemin dinamik kısmının Simulink benzetimi

Tüm sistemin Simulink benzetimi ise Şekil 4.8'de gösterilmektedir.



Şekil 4.8 Tüm sisteminin Simulink modeli

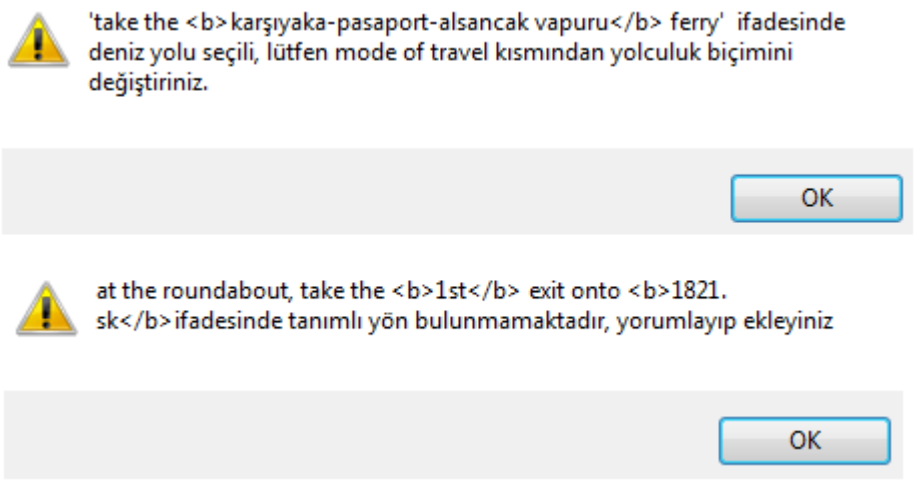
BÖLÜM BEŞ
SONUÇLAR

5.1 Simulasyon Sonuçlarının Değerlendirilmesi

Simülasyonun başarıya ulaşması için gerekli kıstasları belirtmek gerekirse, konum bilgilerinin doğruluğu, yön tarifinin doğruluğu, motor davranışının doğruluğu ve sayısal ölçümlerin doğruluğu sayılabilir.

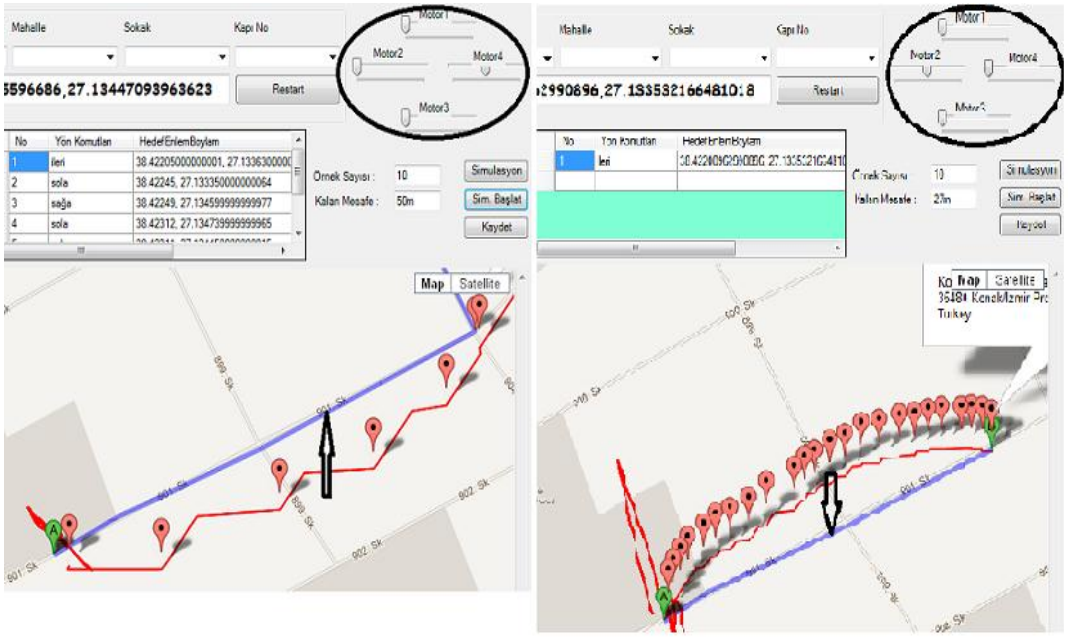
Belirlenen başlangıç ve bitiş noktaları "Google Maps" servisine göre (tercihen) yürünebilir, araç ile gidilebilir, kısaca ulaşılabilir olmadığında, servis bu noktaya en yakın ulaşılabilir noktayı atamaktadır.

Yön tarifi açıklamalarında yön tanımı bulunmayan ifadelerle ya da vapur, feribot gibi ulaşım alternatiflerinin öne sürüldüğü durumlarla karşılaşılabilmektedir. Şekil 3.25'te bu duruma iki örnek gösterilmektedir.



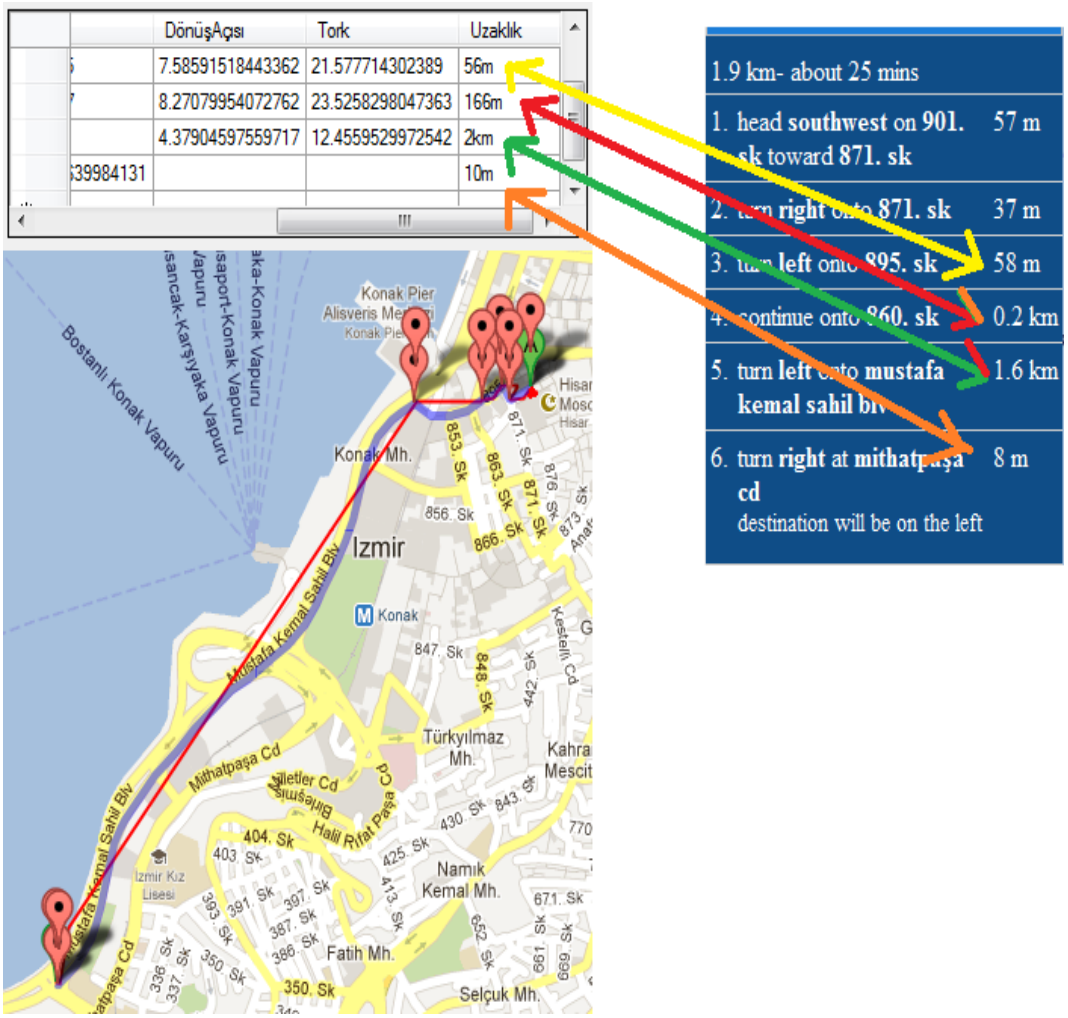
Şekil 5.1 Google Maps servisinden alınan yol tariflerinde karşılaşılan bazı durumlara örnekler

Yol tarifine göre motorların davranışları incelendiğinde iki ara adım arasındaki doğru bulma esasına riayet ettiği gözlemlenmiştir. Şekil 3.26'da simülasyondan örnek kareler verilmiştir.



Şekil 5.2 İki ara adım arası doğruyu bulmada motor davranışı

Bu projedeki sayısal ölçümler; açı, tork ve mesafedir. Mesafe hususunda "Google Maps" servisi temel alındığında tutarlı sonuçlar elde edilmektedir. Fakat sokakların, caddelerin eğimli olması gibi durumlarda hata büyümektedir. Şekil 3.27'de bu duruma bir örnek gösterilmiştir. Diğer ölçümlerde kararlı sonuçlar gözlemlenmektedir.



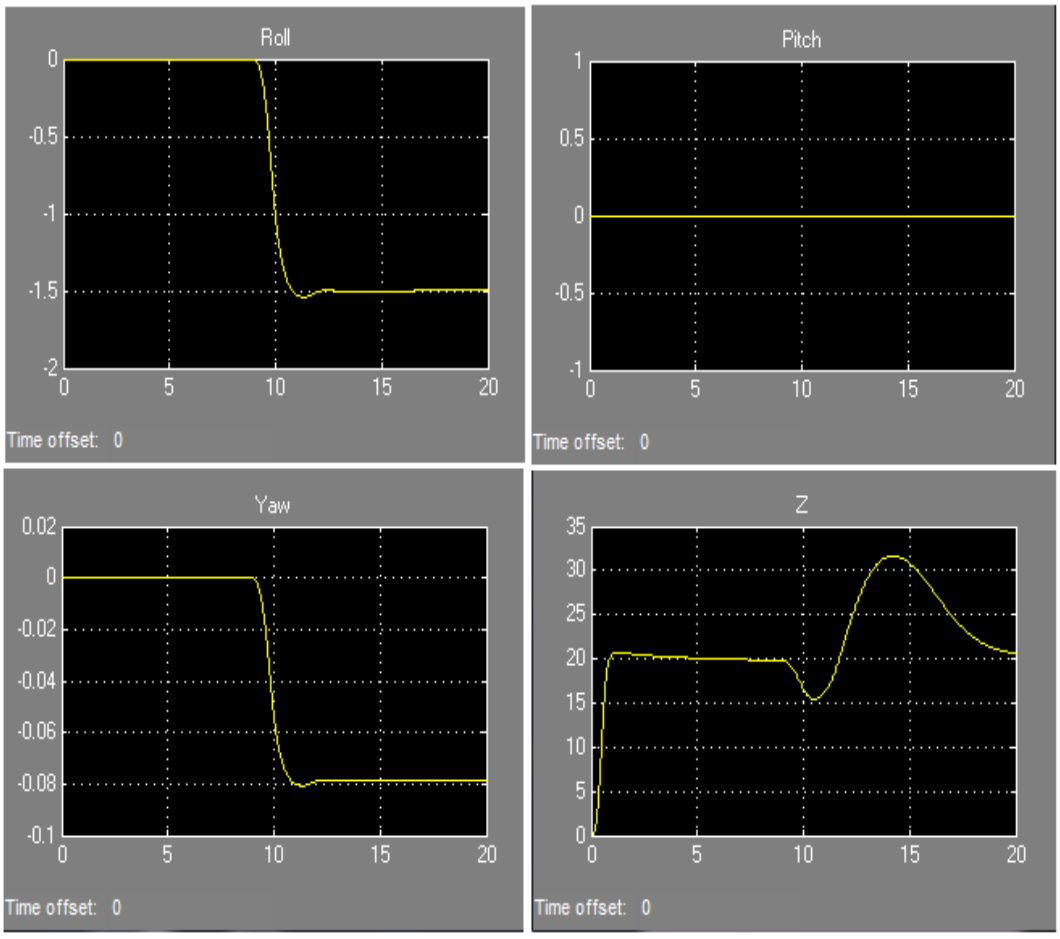
Şekil 5.3 “Google Maps” servisi uzaklık ölçümü ile “Mesafe” fonksiyonu sonucu karşılaştırmasına bir örnek

5.2 Modelleme Sonuçlarının Değerlendirilmesi

Modellemede ‘Slider Gain’ bloğu kullanılarak kumanda temsil edilmektedir. 0-5 volt arası gerilim vermektedir. Gerilim değeri motorun hız kontrol bloğuna gerilim referansı olarak giriş yapmaktadır. Darbe genlik modülasyon bloğu 50 Hertz’e ayarlanmıştır ve 5 volt gerilim çıkışı sağlamaktadır. H köprü çıkışına 24 volt değeri atanmıştır. DC motorun armatür endüktansı 1,6 mH, yüksüz hızı 7000 rpm, nominal hızı 5000 rpm, nominal yükü 19 W, nominal besleme gerilimi 24 volt, rotor ataleti 2,5x10^-5 kgm^2 ve rotor sönümleme sabiti 10^-8 Nm/(rad/s) kabul edilmiştir. Kütle 1 kg, pervane merkezinin kütle merkezine uzaklığı 0,24 m., gövdenin x eksenindeki eylemsizlik momenti 8,1x10^-3 Nms^2, gövdenin y eksenindeki eylemsizlik

momenti 8,1x10^-3 Nms^2 ve gövdenin z eksenindeki eylemsizlik momenti 14,2x10^-3 Nms^2 kabul edilmiştir.

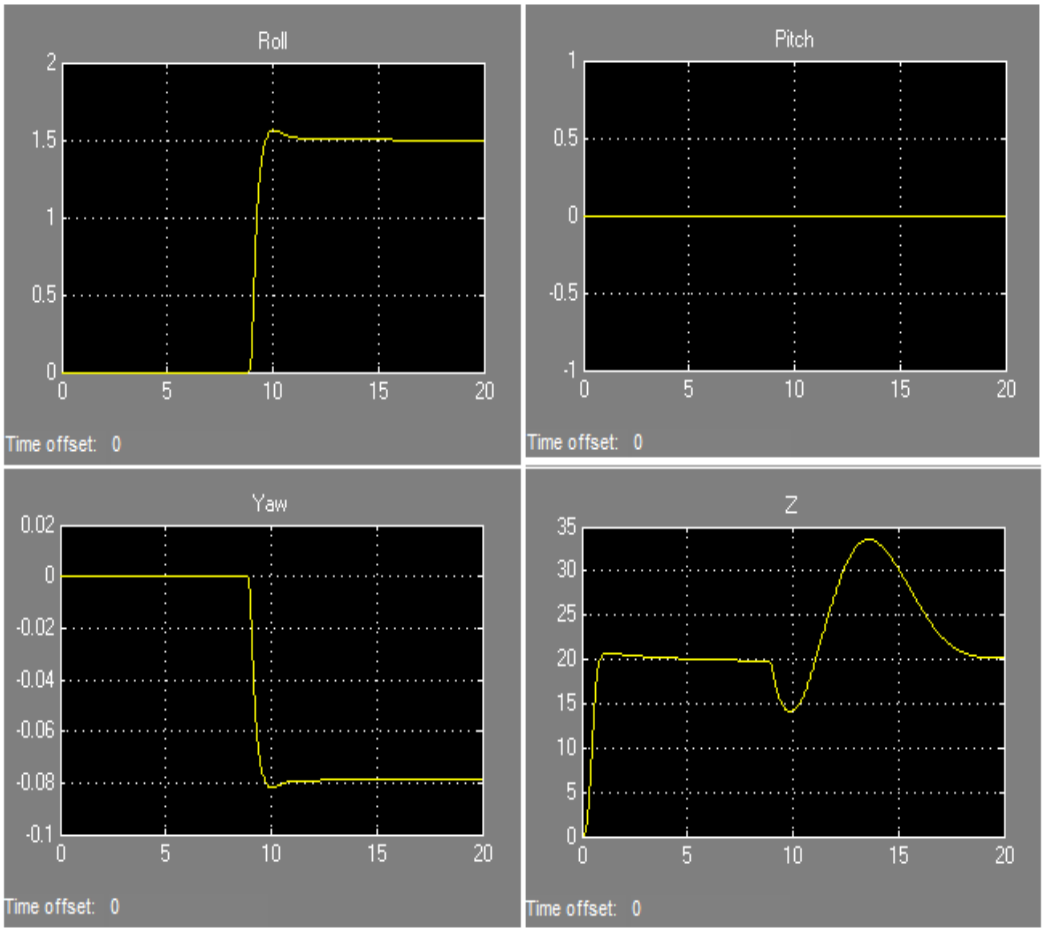
Aşağıda tüm motorlara tam gaz verilirken sol motor referans gerilimi 0'a getirilmiştir.



Şekil 5.4 Sol motor referans gerilimi 0 iken açılar ve yükseklik

Sol motorun referrans gerilimi 0 verildiğinde araç saat yönünün tersi istikametinde $-\pi/2$ kadar yalpalama hareketi yapacaktır.

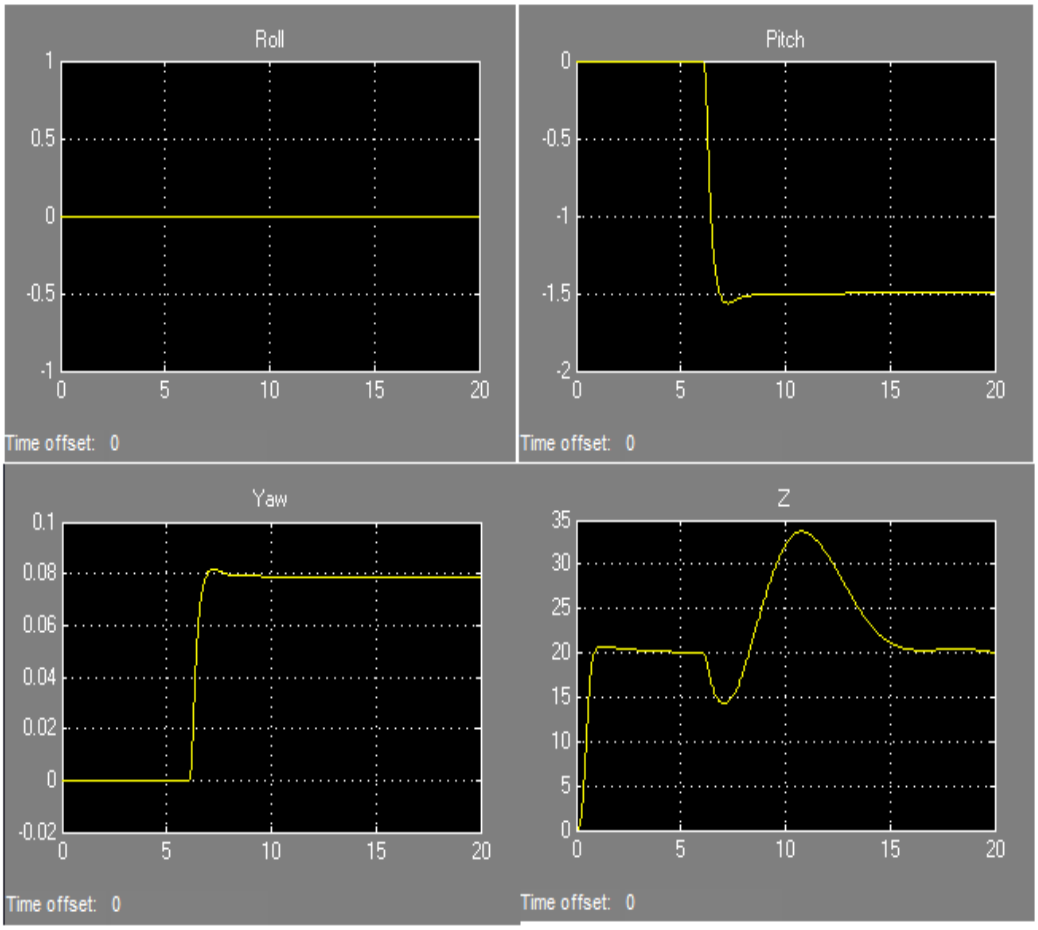
Sağ motor referans gerilimi 0'a getirildiğinde açılar aşağıdaki gibi tespit edilmiştir.



Şekil 5.5 Sağ motor referans gerilimi 0 iken açılar ve yükseklik

Sağ motor referans gerilimi 0'a düşürüldüğünde araç saat yönünde $\pi/2$ kadar yalpalama hareketi yapacaktır.

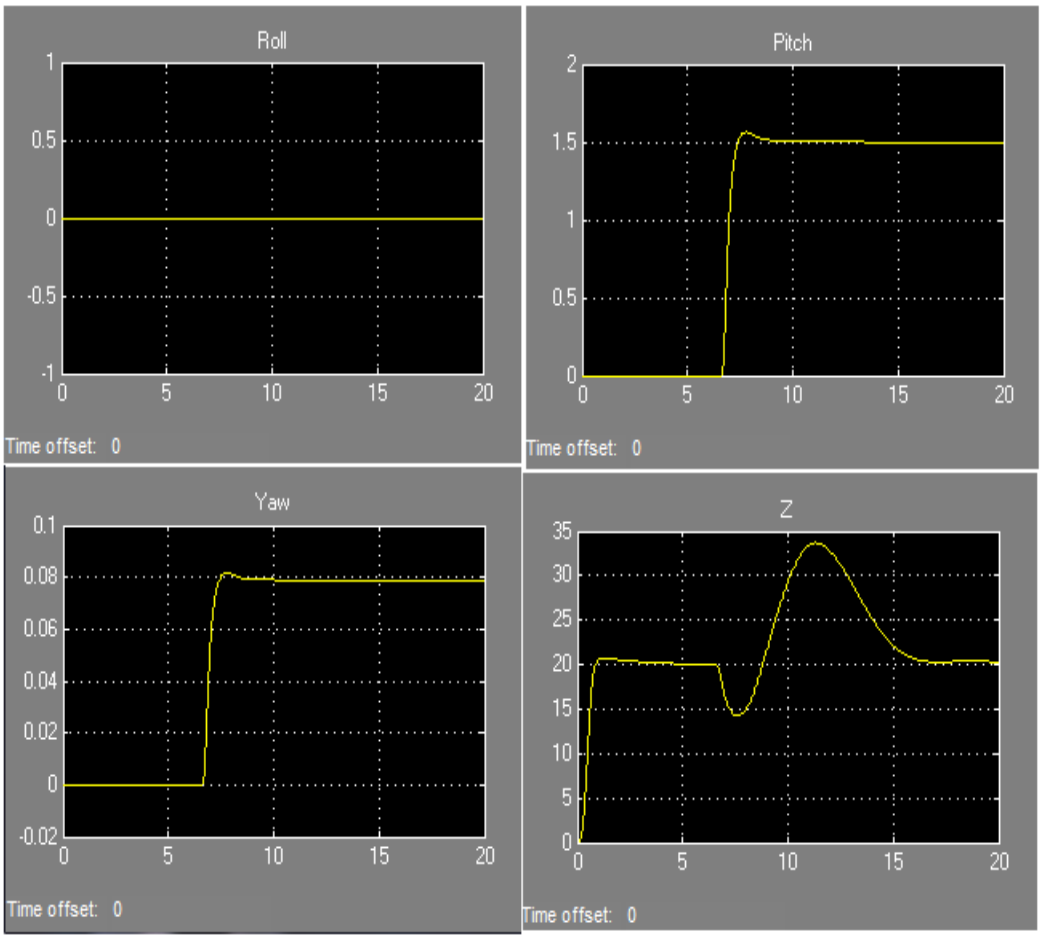
Ön motor referans gerilimi 0'a getirildiğinde tespit edilen açı değerleri aşağıda gösterilmiştir.



Şekil 5.6 Ön motor referans gerilimi 0 iken açılar ve yükseklik

Ön motor referans gerilimi 0'a getilirdiğinde araç $-\pi/2$ kadar yunuslama hareketi yapacaktır.

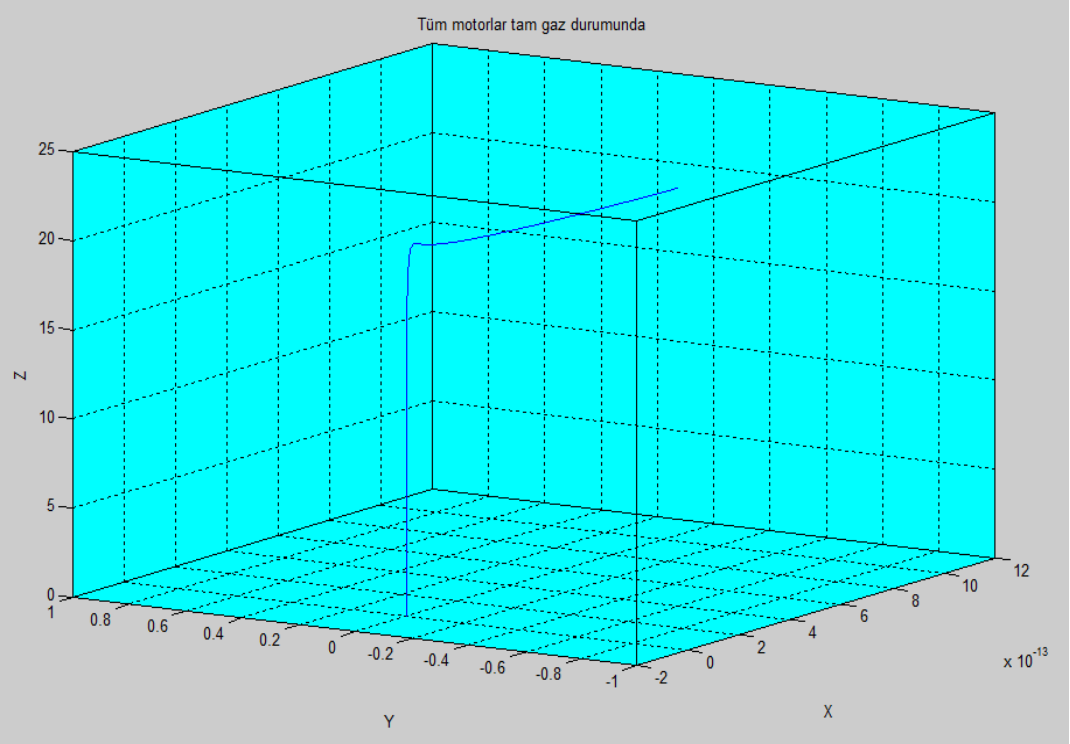
Arka motor referans gerilimi 0'a getirildiğinde,



Şekil 5.7 Arka motor referans gerilimi 0 iken açılar ve yükseklik

sonuçları ortaya çıkmaktadır. Araç bu durumda $\pi/2$ kadar yunuslama hareketi yapacaktır.

Bütün motorlara tam gaz verilirse Şekil 5.8'de görüleceği üzere 3 boyutlu grafiği elde ederiz.

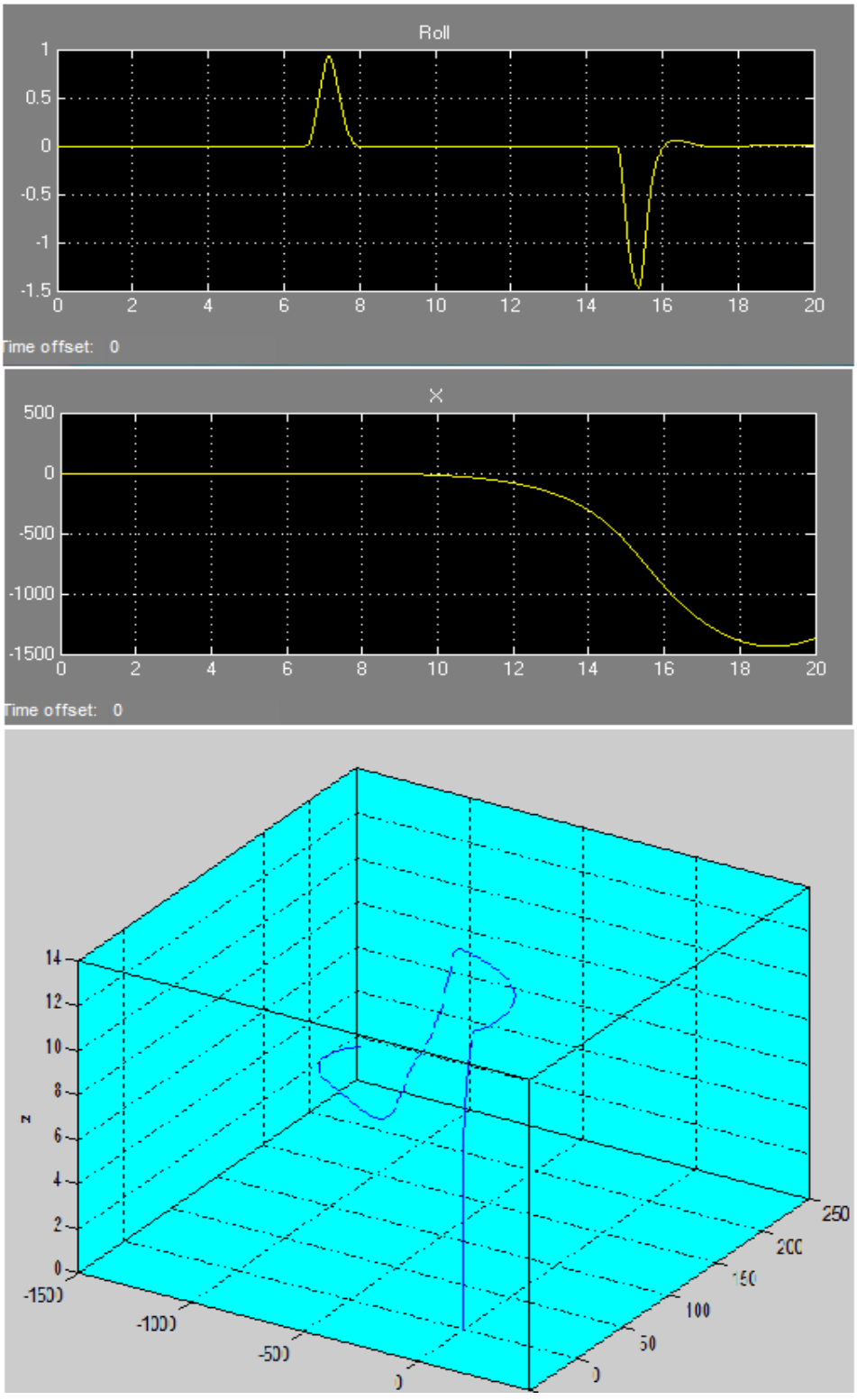


Şekil 5.8 Tüm motorlar tam iken 3 boyutlu gösterim.

Burada X ekseninin değeri çok çok küçüktür ve ihmal edilebilir. Y ekseninde de bir hareket olmamıştır. Araç Z ekseninde bir kalkış gerçekleştirmiştir.

Aracın sağ ve sol motorlarına 7,2. saniyede full gaz verilmiş, 15,4. saniyede referans gerilimi sıfırlanmıştır. Bu durumda araçtan iki kere yön değiştirme beklenir. Değerler Şekil 5.9'da gösterilmiştir.

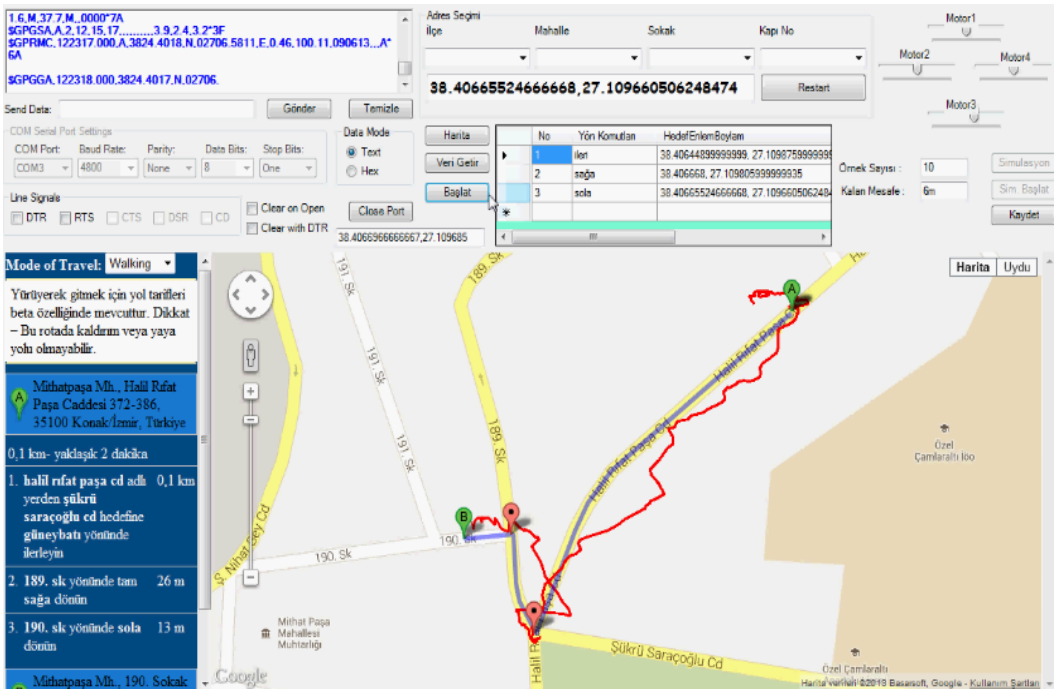
Simulasyon aracın davranışı hususunda genel olarak bize doğru yanıtları vermektedir. Yalpalama, yunuslama, yönelme açılarında ve Z ekseninde PID kontrol uygulanmıştır. Fakat X ve Y eksenlerinde bir kontrol söz konusu değildir. Bu anlamda sistemin davranışı konusunda genel olarak yeterli bir simülasyondur fakat X ve Y eksenlerinde kontrol olmadığından bu eksenlerin değerleri de kararlı değildir.



Şekil 5.9 Yan motorlara önce tam sonra 0 referans gerilimi verilmesi durumunda

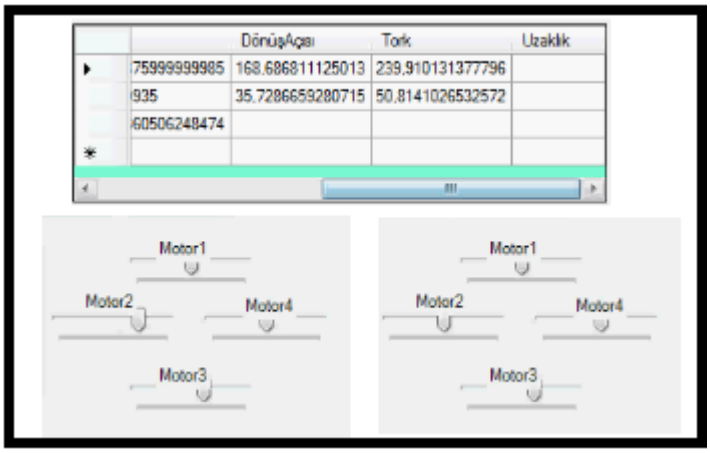
5.3 Deneysel Sonuçların Değerlendirilmesi

Bu deney GPS modülünün seyyar bir şekilde yer değiştirilmesi ile gerçekleştirilmiştir. Motorların dönüş hızlarının modülün konumuna göre nasıl olması gerektiği sliderlar vasıtasıyla gösterilmiştir. Başlangıçta sliderlara, 1024 sayısal değeri baz alınarak, arka motor hariç 512 değeri atanmıştır. İleri gitmesi için arka motora 640 değeri atanmıştır.



Şekil 5.10 Deneyin tamamlanmış hali

İHA'nın tam dönüşü 5 saniyede tamamladığı kabul edilmiştir. Dönüş için ilgili motora kalan 512'lik sayısal değer dönüş açısıyla orantılı olarak ilave edilmiştir. Kısaca dönüş anında sağ ya da sol motora dönüş açısıyla orantılı sayısal fark 5 saniye boyunca uygulanmıştır.



Şekil 5.11 Sağa dönüşte sol motora 512+239, sola dönüşte sağ motora 512+50 değerlerinin 5 saniye boyunca atanması

Daha önce mesafe ölçümünde değinildiği üzere Haversine teoremi ile uzaklık ölçülmüştür. Dönüş ve varış noktalarına 0,00003 enlem ve boylam farkı kala işlem bitirilmiştir. Gene aynı formül üzerinden gidersek bu fark 4 metreye tekabül etmektedir.

Kaydet butonu sayesinde GPS modülünden okunan tüm değerler uzantısı ".ubx" olacak şekilde kaydedilmektedir. Bu sayede U-center isimli program kullanılarak bu dosya okutulmuş ve anbean gidilen yol Google Earth üzerinde gözlemlenmiştir.

Şekil 5.12 Deney kaydının U-center programında Google Earth üzerinde gösterimi

Sonuç olarak iki nokta arasındaki doğrunun konumuna göre motor hareketlerinin doğruluğu gözlemlenmiştir. Bilgi alınan uyduların değişimi söz konusu olduğunda konum bilgisinin doğruluğu azalmaktadır.

KAYNAKLAR

Beard, R. W. (2008). *Quadrotor dynamics and control.* Şubat 2013, http://quad08topgun.groups.et.byu.net/documents/quadrotor_2_20_2008.pdf

Bouabdallah, S., Becker, M., Perrot, V., ve Siegwart, R. (2007). Toward obstacle avoidance on quadrotors. *Proceedings of the XII International Symposium on DynamicProblems of Mechanics (DINAME 2007), 2.*

Güllü R. (2012). *İnsansız hava araçlarının sınıflandırılması.* Ocak 2012, http://e-atolye.net/2012/01/02/insansiz-hava-araclarinin-siniflandirilmasi/.

Henriques, B. S. M. (Haziran, 2011). *Estimation and control of a quadrotor attitude.* Şubat 2013, https://dspace.ist.utl.pt/bitstream/2295/976052/1/Thesis.pdf

MathWorks, (b.t). Documentation Center, *Controlled PWM voltage*, Şubat 2013, http://www.mathworks.com/help/physmod/elec/ref/controlledpwmvoltage.html

MathWorks, (b.t). Documentation Center, *DC motor,* Şubat 2013, http://www.mathworks.com/help/physmod/elec/ref/dcmotor.html

MathWorks, (b.t). Documentation Center, *H-Bridge,* Şubat 2013, http://www.mathworks.com/help/physmod/elec/ref/hbridge.html

MSDN, (b.t). *Introduction to the C# language and the .NET framework,* Ocak 2013, http://msdn.microsoft.com/en-us/library/z1zx9t92%28v=vs.100%29.aspx

Taylor, A. J. P. (2004). *Jane's book of remotely piloted vehicles.*