T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

İNSANSIZ HAVA ARACI TASARIMI VE KONTROLÜ

ANIL SEZGİN

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ PROGRAMI

DANIŞMAN
DOÇ. DR. SIRMA ÇEKİRDEK YAVUZ

İSTANBUL, 2017

T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

İNSANSIZ HAVA ARACI TASARIMI ve KONTROLÜ

Anıl SEZGİN tarafından hazırlanan tez çalışması 10.11.2017 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Doç. Dr. Sırma ÇEKİRDEK YAVUZ

Yıldız Teknik Üniversitesi

Juri Üyeleri

Doç. Dr. Sırma ÇEKİRDEK YAVUZ

Yıldız Teknik Üniversitesi _____

Doç Dr. M. Fatih Amasyalı

Yıldız Teknik Üniversitesi _____

Yrd. Doç. Dr. Oğuz ATA

Altınbaş Üniversitesi _____

Bu çalışma, Yıldız Teknik Üniversitesi Bilimsel Araştırma Projeleri Koordinatörlüğü'nün 2015-04-01-YL01 numaralı projesi ile desteklenmiştir.

ÖNSÖZ

Bu tez konusunu seçmemde ve çalışmalarımda bana yön veren tez danışmanım Doç. Dr. Sırma ÇEKİRDEK YAVUZ'a teşekkürlerimi sunarım. Bu çalışmamı her an yanımda olan, canımdan çok sevdiğim aileme armağan ediyorum.

Ağustos, 2017

Anıl SEZGİN

İÇİNDEKİLER

Sayfa

SİMGE LİSTESİ

I	Atalet matrisi
R	Rotasyon

KISALTMA LİSTESİ

API	Application Programming Interface
HSV	Hue Saturation Value
İHA	İnsansız Hava Aracı
RGB	Red Green Blue
ROS	Robot Operating System
SDK	Software Development Kit

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

Sayfa

İNSANSIZ HAVA ARACI TASARIMI VE KONTROLÜ

Anıl SEZGİN

Bilgisayar Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Tez Danışmanı: Doç. Dr. Sırma ÇEKİRDEK YAVUZ

Dört motorlu araçlar dikey iniş kalkış (Vertical Take-off & Landing, VTOL) ve manevra kabiliyeti sayesinde klasik insansız hava araçlarına göre birçok avantaja sahip olmakla birlikte son yıllarda birçok araştırmacı tarafından çalışılan bir insansız hava araçlarıdır. VTOL özelliği sayesinde engebeli ve kısıtlı ortamlarda uzun piste ihtiyaç duymadan kullanılabilmektedir. Bu çalışmada 4 motorlu uçuş araçlarının tasarımları incelenerek ihtiyaçlar doğrultusunda özelleştirilecek olan tasarımın detaylı anlatımı yapılacak ve kontrolü için bir istasyon geliştirilecektir. Geliştirilecek olan yazılım ile insansız hava aracının kontrolü sağlanacak ve araç üzerinde bulunan sensörler ile ortamdan alınan veriler işlenecektir. Bu control istasyonu ile insansız hava aracını uzaktan kumanda ederek kullanmak veya otonom görevleri takip etmek, havada kaldıkları süre içinde ve görüş alanı dışında iken yer ve durumunu izlemek için kullanılacaktır. Böylece askeri amaçlar için geliştirilen insansız hava aracı teknolojisinin arama kurtarma alanında kullanımı sağlanacaktır.

Anahtar Kelimeler: İnsansız hava aracı, görüntü işleme, otonom uçuş sistemi, kontrol

ABSTRACT

DESIGN AND CONTROL OF UNMANNED AERIAL VEHICLE

Anıl SEZGİN

Department of Computer Engineering

MSc. Thesis

Adviser: Doç. Dr. Sırma ÇEKİRDEK YAVUZ

Quadrotor has a huge advantage to classical unmanned aerial vehicles (UAV) since it performs vertical take-off and landing (VTOL) with high maneuverability, which is why it has been worked on by many researchers in recent years. Thanks to VTOL ability, it can be used in rugged and limited terrain without the need of long runways for take-off and landing. In this study, the structure and dynamics of UAVs will be examined and a software will be developed to control a Quadrotor. With this control station, the UAV will be remote controlled, the autonomous missions will be tracked and the vehicles status when it is airborne or out of sight will be monitored. Thus the UAV technology developed for military intentions will be able to be used in the search and rescue field.

Keywords: Unmanned aerial vehicle, image processing, autonomous flight system, control

YILDIZ TECHNICAL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

BÖLÜM 1

GİRİŞ

Son yıllarda insansız hava araçları (İHA, UAV) askeri araştırmaların ve uzay çalışmalarının önemli bir bölümünü oluşturmaktadır. Bu araçlar insan hayatını tehlikeye atmadan birçok görevi başarıyla yerine getirme şansı sunar. Bu araçlar, insanlı sistemlere göre, özellikle askeri alanda tehlikenin yoğun olduğu bölgelerde insan kaybı riskinin bulunmaması ve hava aracı performasının insan zaaflarına bağlı olmaması gibi çok büyük avantajlara sahiptir. Bu nedenle günümüzde bir çok ülke bu alanda arge çalışması yapmaktadır. NATO kaynaklarında kabul gören insansız hava aracı tanımı şöyledir; İçinde insan olmayan, uzaktan kumanda ile yönlendirilen veya otonom olarak kendisini yönlendiren motorlu itki gücü olan, silah ya da faydalı yükleri ana gövdesine yüklenip çıkarılabilen, görev sonu geri dönerek iniş yapabilen veya hedefte silah olarak kendini imha edebilen araçlardır.

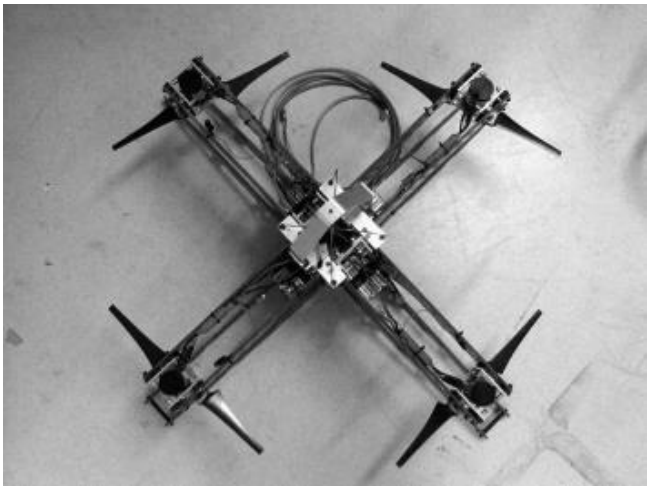
Dört motorlu araçlar dikey iniş kalkış (Vertical Take-off & Landing, VTOL) ve manevra kabiliyeti sayesinde klasik insansız hava araçlarına göre birçok avantaja sahip olmakla birlikte son yıllarda birçok araştırmacı tarafından çalışılan bir insansız hava araçlarıdır. VTOL özelliği sayesinde engebeli ve kısıtlı ortamlarda uzun piste ihtiyaç duymadan kullanılabilmektedir. Ayrıca 4 motorlu VTOL özelliği taşıyan diğer insansız hava araçlarına göre mekanik olarak daha basit yapıya sahiptir. Fakat bu araçlar, bir çok avantajına rağmen kararsız bir yapıya sahiptir ve yüksek derecede doğrusal olmayan ve birbiriyle ilişkili bir dinamiğe sahiptir. Bu nedenle 4 motorlu araçların kontrolü zorlukları da beraberinde getirmektedir. Bu çalışmada 4 motorlu araçların yapısı incelenip uygun bir tasarım bulunacak ve kontrolü için bir yazılım geliştirilecektir.

1.1 Literatür Taraması

EADS'nin (European Aeronautic Defense and Space Company) geliştirdiği 4 motorlu hava aracı 20 dakikalık bir uçuş süresine sahip, ağırlığı 0,5 kg, uçuş menzili 1km, 6-axis atalet sensörü, bir GPS ünitesi ve gaz sensörü bulunan bir insansız hava aracıdır [1].

Şekil 1. 1 Quattrocopter

Australian National University tarafından geliştirilen X4-Flyer dual HC-12 mikro işlemcili bir kontrol kartı, kendi geliştirdikleri Eimu adında 6-axis sensör, input PWM sinyalleri için R700 JR Slimline RC Receiver, JP 3810 radio transmitter mevcuttur. Hız kontollörü olarak MSC30BL kullanılmıştır [2].



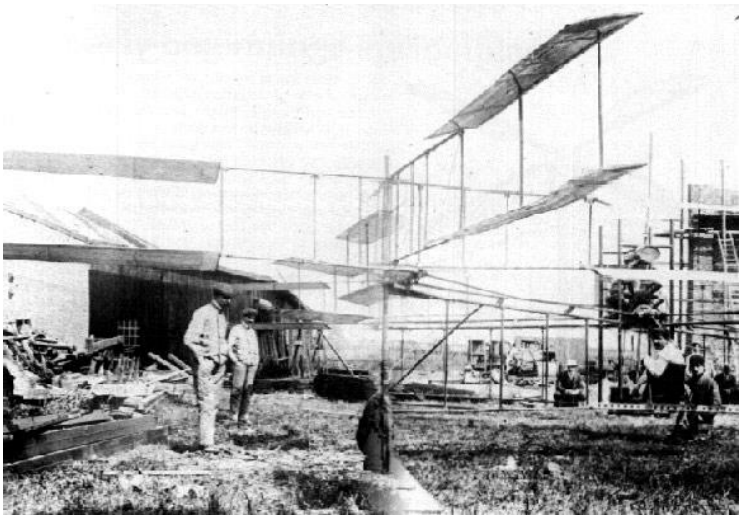
Şekil 1. 2 X4-Flyer

University of British Columbia quad-rotor insansız hava araçlarının nonlineer modellenmesi üzerine bir çalışma yapmıştır. DSP teknikleri, mikroişlemci ve kablosuz transmitter kullanılarak uçuş kontrolörü test edilmiştir. Nonlineer modelden yola çıkılarak bir H∞ loop shaping kontrolör tasarlanmıştır. Bu kontrolör tasarımında hedeflenen ise stabil çalışma, hız ve yaw kontrolü [3].

Stanford University tarafından geliştirilen STARMAC işaretlenen noktaya path planning yaparak ulaşmaktadır. Geliştirdikleri kontrol algoritmasının yükseklik bilgileri yüzünden sürekli olarak motor güçlerini değiştirmek zorunda kaldığını görmüşlerdir [4].

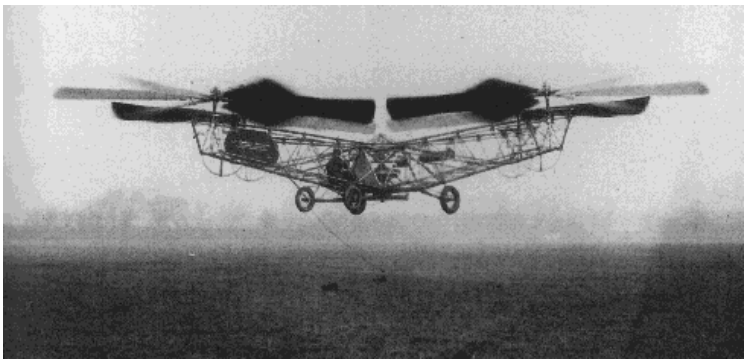
İnsansız hava aracı olarak 4 motorlu uçuş aracının tercih etmemizin sebepleri her türlü zeminde rahatlıkla dikey iniş – kalkış yapabilmesi (VTOL, vertical take-off and landing) ve manevra kabiliyetidir. Bu 2 özellik sayesinde indoor/outdoor olarak kullanılabilir, gözetleme ve araştırma görevlerinde büyük bir avantaj sağlar. Ancak manevra kabiliyeti bu araçların kontrolünü zorlaştırır [5].

20. yüzyılın başlarında Fransız bilim adamı ve akademisyen Charles Richet pilotsuz bir helikopter geliştirdi ancak bu çalışma başarılı sonuçlanmadı. Ancak bu çalışma Charles Richet'in öğrencisi olan Louis Breguet'e ilham verdi. 1906 yılında Louis ve Jacques Breguet kardeşler helikopter çalışmalarına başladı ve 1907 yılında Gyroplane No. 1 adını verdikleri ilk 4 motorlu aracı geliştirdiler [6].



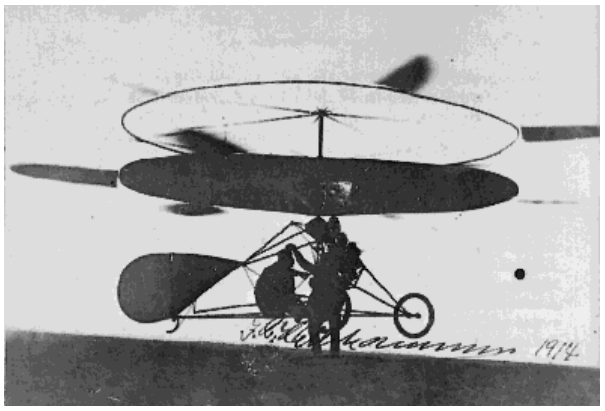
Şekil 1. 3 Gyroplane no. 1

1922 yılında Georges de Bothezat X şeklinde bir gövdeye sahip ve rotorların X'in uçlarına yerleştirildiği bir quadrotor geliştirdi [6].



Şekil 1. 4 Bothezat

1914 yılında Danimarkalı Jen C. Ellehammer bir coaxial rotor helikopter geliştirdi [6].



Şekil 1. 5 Coaxial rotor helikopter

1.2 Amaç

Bu çalışmada 4 motorlu uçuş araçlarının tasarımları incelenerek ihtiyaçlar doğrultusunda özelleştirilecek olan tasarımın detaylı anlatımı yapılacak ve kontrolü için bir istasyon geliştirilecektir. Geliştirilecek olan yazılım (Kontrol İstasyonu) ile insansız hava aracının kontrolü sağlanacak ve araç üzerinde bulunan sensörler ile ortamdan alınan veriler işlenecektir. Bu kontrol istasyonu ile insansız hava aracını uzaktan kumanda ederek kullanmak veya otonom görevleri takip etmek, havada kaldıkları süre içinde ve görüş alanı dışında iken yer ve durumunu izlemek için kullanılacaktır. Bu sayede insansız hava aracı

teknolojisi, fiziksel gözetlemenin mümkün olmadığı ücra bölgelerde sınır güvenliği ve güvenlik ihlalleri tespitinde kullanılacaktır.

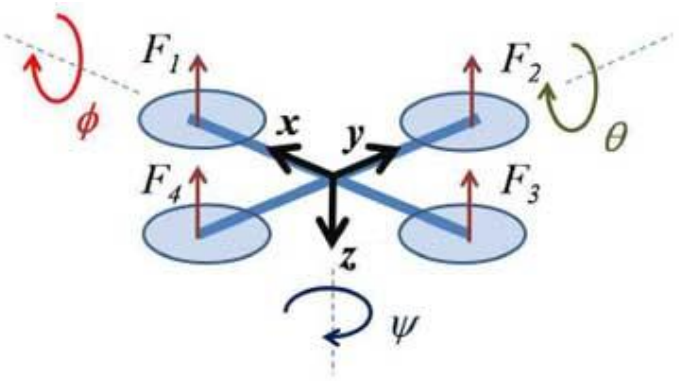
1.3 Hipotez

Teknolojik gelişmeler sınır güvenliğinin sağlanmasında yeni çözümler üretmiştir. Bu çözümlerde sınır güvenliği için personel ile gözetlemeyi minimuma indirmek amaçlanmıştır. Bu tez kapsamında geliştirilen uygulamalar belirli parametreleri göz önünde bulundurarak otomatik hedef tespiti yapmaktadır. Belirli bir alan içerisinde güvenliği tehdit edebilecek unsurlar mevcut hedef tespit algoritmalarıyla birleştirilerek kullanılacaktır. Geliştirilecek olan kontrol istasyonu ile aracın uzaktan kumanda edilmesinin yanı sıra belirli nesneleri takip ederek otonom görevleri yerine getirmesi sağlanacaktır.

BÖLÜM 2

MATEMATİKSEL MODEL

Bu çalışmada bir quad-rotor aracın durum ve yükseklik dinamikleri incelenmiş olup, roll, pitch, yaw hareketleri ve yükseklik değerleri kullanılarak sistemin matematiksel modeli çıkarılmıştır.



Şekil 2. 1 Motor hareket yönleri

F1, F2, F3 ve F4 kuvvetleri quad-rotor üzerinde bulunan pervaneler tarafından uygulanan itme kuvvetlerini temsil etmektedir.

Quad-rotor aracının yapısı simetrik olmalıdır. F1 kuvvetinin uygulandığı kısım ön, F2 sağ, F3 arka ve F4 sol tarafı göstermektedir.

Roll, pitch ve yaw hareketleri sırasıyla φ, θ, ve ψ sembolleriyle gösterilmektedir.

Eksenler ile yapılan açılar ise p, q, r ile gösterilmektedir.

Rotation: R, Şekil 2.2'de gösterilmektedir.

$$\psi \text{ Rotasyon } R_1 = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} x_b \\ y_b \\ z_b \end{bmatrix}$$

$$\theta \text{ Rotasyon } R_2 = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}$$

$$\varphi \text{ Rotasyon } R_3 = \begin{bmatrix} x_3 \\ y_3 \\ z_3 \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}$$

$$R_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix}, \; R_2 = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\varphi & 0 & \cos\theta \end{bmatrix}$$

$$R_3 = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Şekil 2. 2 Rotasyon

Son durumda R Şekil 2.3'de gösterilmektedir.

$$R = \begin{bmatrix} \cos\psi\cos\theta & \cos\psi\sin\theta\sin\varphi - \cos\varphi\sin\psi & \cos\psi\sin\theta\cos\varphi + \sin\psi\sin\varphi \\ \sin\psi\cos\theta & \sin\psi\sin\theta\sin\varphi + \cos\psi\cos\varphi & \sin\psi\sin\theta\cos\varphi - \sin\varphi\cos\psi \\ -\sin\theta & \sin\varphi\cos\theta & \cos\theta\cos\varphi \end{bmatrix}$$

Şekil 2. 3 Rotasyon matrisi

Açısal hız vektörü Şekil 2.4'te gösterilmektedir.

$$\omega = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

Şekil 2. 4 Açısal hız vektörü

Euler açılarının durum denklemleri Şekil 2.5'te gösterilmektedir.

$$\varphi = p + q \tan\theta \sin\varphi + r \tan\theta \cos\varphi$$

$$\theta = q \cos\varphi - r \sin\varphi$$

$$\psi = p + q \sec\theta \sin\varphi + r \sec\theta \cos\varphi$$

Şekil 2. 5 Durum denklemleri

Atalet matrisi Şekil 2.6'da gösterilmektedir.

$$I = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}$$

Şekil 2. 6 Atalet matrisi

Atalet matrisi bir diagonal matris olarak varsayılmalıdır, yani çarpımları 0 olmalıdır.

3 boyut momentum için Newton kanunu Şekil 2.7'de gösterilmektedir.

$$I\dot{\omega} + \omega \times (I\omega) = M$$

Şekil 2. 7 3 Boyut momentum için Newton kanunu

Newton kanunu açısal hızların üzerine uygulandığında son durum Şekil 2.8'de gösterilmektedir.

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = Ir^{-1}.\begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} - I^{-1}.\left\{\begin{bmatrix} p \\ q \\ r \end{bmatrix} \times I.\begin{bmatrix} p \\ q \\ r \end{bmatrix}\right\}$$

Şekil 2. 8 Açısal hızlar

Roll, pitch ve yaw eksenleri için momentum hesaplamaları Şekil 2.9'da gösterilmektedir.

$$\sum M_x = (F_3 - F_1)\frac{L}{2}$$

$$\sum M_y = (F_2 - F_4)\frac{L}{2}$$

$$\sum M_z = (M_1 + M_3) - (M_2 + M_4)$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{qr}{I_x}(I_y - I_z) - \frac{L}{2I_x}(F_1 - F_3) \\ \frac{pr}{I_y}(I_y - I_z) - \frac{L}{2I_2}(F_2 - F_4) \\ \frac{pq}{I_z}(I_x - I_y) + \frac{((F_2 + F_4) - (F_1 + F_3))d}{I_z b} \end{bmatrix}$$

Şekil 2. 9 Roll, pitch, yaw için momentum hesaplama

Durum vektörü ve durum denklemleri Şekil 2.10'da gösterilmektedir.

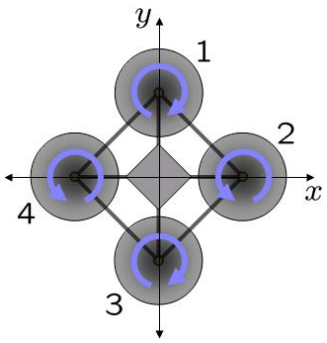
$$x = \begin{bmatrix} \varphi \\ \theta \\ \psi \\ p \\ q \\ r \end{bmatrix}, \quad \begin{bmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} p + q\tan\theta\sin\varphi + r\tan\theta\cos\varphi \\ q\cos\varphi - r\sin\varphi \\ p + q\sec\theta\sin\varphi + r\sec\theta\cos\varphi \\ \frac{qr}{I_x}(I_y - I_z) - \frac{L}{2I_x}(F_1 - F_3) \\ \frac{pr}{I_y}(I_z - I_x) - \frac{L}{2I_y}(F_2 - F_4) \\ \frac{pq}{I_z}(I_x - I_y) + \frac{((F_2 + F_4) - (F_1 + F_3))d}{I_z b} \end{bmatrix}$$

Şekil 2. 10 Durum vektörü ve durum denklemleri

BÖLÜM 3

DRONE YAPISI

Quad-Rotor 4 motorlu olarak modellenen bir hava aracıdır. Bu motorların 2 tanesi saat yönünde dönerken, diğer ikisi ters yönde dönmektedir, böylece pervanelerin oluşturduğu tork dengelenir. Motorların farklı hızlarda dönmesiyle boşlukta istenildiği gibi hareket edebilir. Komşu motorların pervane helisleri farklı olmalıdır, böylece komşu pervaneler birbirilerine zıt yönde dönerek yerçekimine zıt dikey kuvvet oluştururlar.



Şekil 3. 1 Motorların dönüş yönleri

Bu tip hava araçları 6-axis sensöre (6 Degrees of Freedom) sahip gözükseler de 4-axise sahiptir. Bunun sebebi x ekseni etrafında dönerken ile y eksenindeki ilerlemesiyle, y ekseni etrafında dönerken x ekseninde ilerlemesi birbirinden bağımsız değildir. Bu durumda hareketler x,y,z eksenleri etrafında dönme ve z ekseninde yükselmedir.

Drone'lar otonom ya da uzaktan kontrol edilebilen insansız araçlardır. Kara, hava ve deniz için olan farklı modelleri mevcuttur. İnsansız hava araçları birçok farklı görevi yerine

getirebilecek şekilde tasarlanabilir. Quadrotor küçük insansız hava araçlarına örnektir, bu tarz araçlar en çok hava fotoğrafçılığı için kullanılmaktadır.

3.1 Sabit Kanat vs Döner Kanat

İnsansız hava araçları genel olarak 2 formda bulunur: Sabit Kanat ve Döner Kanat (Rotorcraft). Sabit kanat insansız hava araçları çok daha uzun mesafe ve çok daha hızlı uçabilir. Ancak bu tarz araçların iniş kalkışları için çok uzun bir pist gereklidir. Döner kanat insansız hava araçları ise dikey bir şekilde iniş kalkış yapabilir ve daha fazla manevra kabiliyetine sahiptir. Dolayısıyla kullanılacak göreve göre drone seçilmelidir.

Döner kanat araçların en önemli özelliklerinden biri ise bulunduğu yerde sabit kalabilmesidir. Bu sayede istenilen hedefe daha rahat yaklaşılabilir ve görüntü alınabilir. Bu araçlar iniş yaptıktan sonra tüm motorları durdurup ses ve görüntü kaydedici gibi çevresel aygıt ya da sensörlerini çalışır halde tutabilir, bu işleme perch-and-stare denir [7]. Bu tip araçlar ile yük taşınması durumunda ise yük istenilen yere rahatlıkla bırakılabilir, sabit kanat araçlarla bu durum mümkün değildir.

Bu tip araçların bu kadar popüler olması başta pil, wireless iletişim ve solid state cihazlar gibi birçok alanda yenilikler yapılmasını teşvik etti. Bu düşük bütçeli insansız hava araçları birçok askeri ve sivil uygulamanın birer parçası haline geldi. Düşük bütçeli insansız hava araçları data toplama, doğa olaylarını izleme, güvenlik için gözetim gibi alanlarda kullanılmaya başlandı. Araştırmacılar bu küçük insansız hava araçlarının tasarımı üzerinde çalışırken başta uçuş platformlarının ağırlığı ve kullanılan parçaların harcadığı enerji gibi birçok farklı probleme daha uygun çözümler üretmek için uğraşmaktadır.

3.2 Drone Özellikleri

Drone seçimi yapılırken dikkat edilmesi gereken özellikler şunlardır: ağırlık, dayanıklılık, mesafe, irtifa, kanat yükü ve motor tipi.

3.2.1 Ağırlık

İnsansız hava araçlarının ağırlıkları çok geniş bir aralıktadır. 400gr'lık araçlar bulunmakta iken bunun yanında 10 tonluk RQ-4 Global Hawk gibi araçlar da mevcuttur.

Şekil 3. 2 RQ-4 Global Hawk

3.2.2 Dayanıklılık ve Mesafe

İnsansız hava araçlarının farklı görevlere göre sınıflandırılması gibi dayanıklılık ve mesafeye göre de sınıflandırılması önemlidir. Drone mesafesi hesaplanırken ne kadar sıklıkla şarj edilmesi ya da yakıt ikmali yapılması gerektiği göz önünde bulundurulur. Birçok drone şarj ya da yakıt için yere inmek zorundadır ve bu işlem zamanlarını etkiler. Global Hawk gibi büyük drone'lar iniş yapmadan havada yakıt ikmali yapabilir ancak bu bile işlem zamanını etkileyecektir. AR.Drone daha düşük bir dayanıklılığa sahip olmasıyla birlikte yaklaşık 20 dakika havada kalabilir, bu süre kısa mesafeli görevler için uygundur. Yüksek dayanıklılığa sahip araçlar ise 24 saatten fazla havada kalabilir ve yaklaşık 22000 km gidebilir.

3.2.3 İrtifa

Bu özelliğin askerin alanda drone seçiminde önemli olmasının sebebi araç yüksek irtifada iken daha zor görüleceği için yakalanma ihtimalinin düşük olmasıdır, bu aracın farkedilip ve yokedilmesini engeller. Ayrıca irtifa görüntü alımında da önemlidir, araç ne kadar yüksekten gidiyor ise aldığı arazi görüntüsü o kadar büyük olacaktır. AR.Drone gibi düşük

irtifalı araçlar 100 m gibi bir yüksekliğe çıkabilir. Darkstar ve Predator B gibi yüksek irtifa kabiliyetine sahip drone'lar ise 45.000 ft yüksekliğe çıkabilir.

Şekil 3. 3 Predator B

Şekil 3. 4 Darkstar

3.2.4 Kanat Yükü

Kanat yükü uçak ağırlığının kanat alanına oranı olarak tanımlanır. Bu oran ile bir insansız hava aracının ne kadar yük taşıyabileceği ve yükünü taşımak için hangi hızda hareket etmesi gerektiği hesaplanır. Bir hava aracı ne kadar hızlı uçarsa kanat birim alanında o kadar fazla kaldırma olur. Bu sayede küçük insansız hava araçları yüksek hızda neredeyse

kendi ağırlığı kadar yük taşıyabilmektedir. Ancak aracın fazla yük taşıması yüksek hızda iniş kalkış yapmasını zorlar ve manevra kabiliyetini azaltır [8].

3.2.5 Motor Tipi

İnsansız hava araçları motor tiplerine göre de sınıflandırılabilir. Aracın ağırlığı motor tipine bağlıdır, araç ne kadar büyük ise kalkış için daha fazla güç gerekecektir. Küçük insansız hava araçlarında daha çok elektrik motorları kullanılırken endüstriyel araçlarda ise piston motorları kullanılmaktadır. Ayrıca motor tiplerinin drone'larda mesafe ve dayanıklılıkta etkisi vardır.

3.3 Drone Seçimi

Belirtilen özelliklere bakılarak bu çalışma kapsamında seçilmesi gereken İHA hafif olmalı, yeterli mesafe gidebilmesi için 20 dakika havada kalabilmeli, yaklaşık 20 metre yüksekliğe çıkabilmeli, dikey iniş kalkış (Vertical Take-Off and Landing) yapabilmeli, görüntü alabileceğimiz kameraları olmalıdır. Bu özellikler göz önünde bulundurulduğunda Parrot firmasına ait olan AR.Drone 2 bu çalışmada kullanılacak olan İHA olmalıdır.

3.3.1 AR Drone Quadrotor

Parrot AR Drone video oyun sektörü için geliştirilmiş bir mikro insansız hava aracıdır. 2010 yılında piyasaya sürülen drone hem askeri hem sivil uygulamalarda kullanılmaktadır. Drone'un hafif olması, düşük maliyetli olması ve manevra kabiliyeti sayesinde akademik araştırmalarda da kullanılmaktadır. Bu modelin seçilmesinde etkili olan 3 neden ürünün sağlamlığı, modifiye edilebilmesi ve fiyat olarak benzerlerinden çok daha uygun olmasıdır. AR Drone birçok mobil cihaz ya da Wi-Fi arayüzü olan bir cihaz ile uzaktan kumanda edilen ilk 4 pervaneli helikopterdir. Cihazın üzerinde sensörler, bir ön ve bir alt kamera ve ultrasonik yükseklikölçer bulunmaktadır. Bu hava aracı Wi-Fi bağlantı üzerinden gönderilen komutlarla çalışmaktadır.

Şekil 3. 5 AR.Drone 2.0

AR.Drone Parrot SA tarafından geliştirilmiş bir araçtır. Pervaneleri korumak için Expanded Polypropylene (EPP) kullanılarak geliştirilmiş iç mekan koruyucusu buunmaktadır, bu sayede dayanıklı, hafif ve geri dönüşümü olan bir parçadır. Pervaneler 4 fırçasız motor ile çalıştırılmaktadır (28500 RPM ve 14,5W). Araçta 1500mAh Li-po pil bulunmaktadır bu da araca yaklaşık 10 dakikalık bir uçuş sağlamaktadır.

Ar.Drone aracında 1GHz 32bit ARM Cortex A8 işlemci ve 1GB DDR2 Ram'e sahip bir bilgisayar bulunmaktadır. Bu bilgisayar üzerinde Linux işletim sistemi çalıştırılmaktadır. Üzerinde bulunan USB bağlantılar sayesinde cihaz yazılımı üzerindeki değişiklikler ve GPS sensör gibi yeni donanımları kolaylıkla sisteme dahil etmek mümkündür. AR.Drone üzerinde 6 axis atalet sensörü bulunmaktadır (6-DOF, degrees of freedom), böylelikle pitch, yaw ve roll ölçümleriyle aracın kontrolü sağlanır. Ölçüm sisteminde 3 eksenli accelerometer, 2 eksenli roll ve pitch gyrometer ve 1 eksen yaw gyrometer bulunmaktadır. Accelerometer olarak Bosch Sensortec tarafından geliştirilmiş olan BMA150 kullanılmaktadır.

Bu hava aracı karbon fiber ve kuvvetli PA66 plastikten yapılmıştır. MEMS (Mikro-Elektro-Mekanik Sistem) ve video işleme sayesinde uzaktan kumandalı bir objenin sezgisel pilotluğu mümkün kılınır. Wi-Fi arayüzü bulunan tüm cihazlara anlık video aktarımı yapılarak ve bu video üzerinde görüntü işleme yazılımları kullanılarak cihazın istenilen

amaç için çalışması sağlanır. Bu hava aracında kullanılan pervaneler bu araca özgü üretilmiştir ve araç karbon fiber tüp gövdeye sahiptir.

Şekil 3. 6 AR Drone 2.0 parçaları

3.3.2 Motor

AR Drone'da bir microcontroller tarafından kontrol edilen fırçasız motorlar bulunmaktadır. Araç tüm motorların çalışıp çalışmadığını kontrol edebilir. Bu sayede araç kontrolünün kaybedilmesinin önüne geçilir. Herhangi bir pervanenin dönmediği tespit edildiğinde, araç tüm motorları durdurur [9].

3.3.3 LiPo Pil

AR Drone 2.0 aracında 1000mAh, 11.1V LiPo pil bulunmaktadır. Araç havadayken pil voltajı tam şarjdan (12.5V) düşük şarja (9V) doğru azalır. AR Drone pil voltajını ölçerek onu pil yüzdesi olarak çevirmektedir (Tam dolu ise %100, düşük ise %0). Araç düşük pil tespit ettiğinde, kullanıcıya bir uyarı mesajı gönderir ve otomatik iniş yapar. Eğer voltaj kritik bir seviyedeyse olası zararlardan korunmak için tüm sistem kapatılır [9].

3.3.4 Sensör

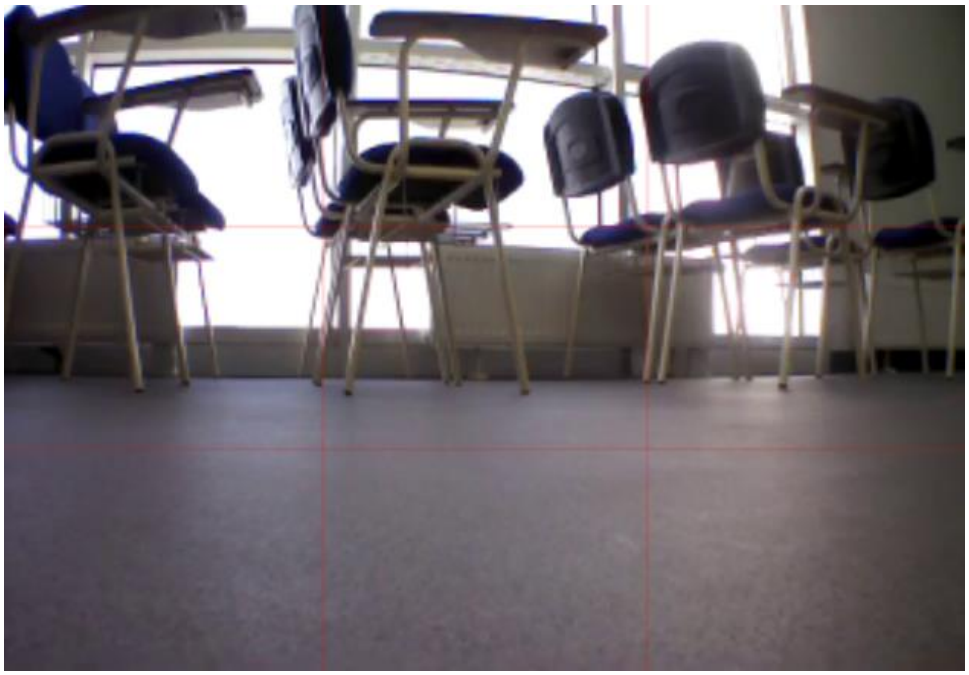
Aracın gövdesinde birçok sensör bulunmaktadır. AR Drone 1.0 modelinde 6-DOF mikro elektro-mekanik system tabanlı (MEMS) atalet ölçüm cihazı vardır, bu cihaz sayesinde aracın pitch, roll ve yaw değerleri alınır. Ayrıca cihaz üzerinde 3 eksenli accelerometer, 2 eksenli roll ve pitch gyrometer ve tek eksenli yaw gyrometer bulunmaktadır [10]. AR Drone 2.0 üzerinde ise 3 eksenli magnetometer vardır. Araç üzerinde 2 kamera ve üzerinde 2 işlemci ve sonar olan 1 anakart mevcuttur. Bu işlemcilerden biri çevre birimlerden veri toplamak için diğeri ise aracın uçuş stabilitesi ve görüntü işleme algoritmaları içindir [11].

3.3.5 Anakart

AR Drone üzerindeki anakartta 1GHz ARM Cortex A8 işlemci, 1Gb 200MHz DDR2 RAM, Atheros Wi-Fi chipset ve USB port mevcuttur. Anakarta bir basınç sensörü ve dikeyde görüntü alınabilen bir kamera yerleştirilmiştir. Bu basınç sensörü ve kamera araç üzerinde bulunan navigasyon kartının daha stabil çalışmasını sağlar.

3.3.6 Kamera

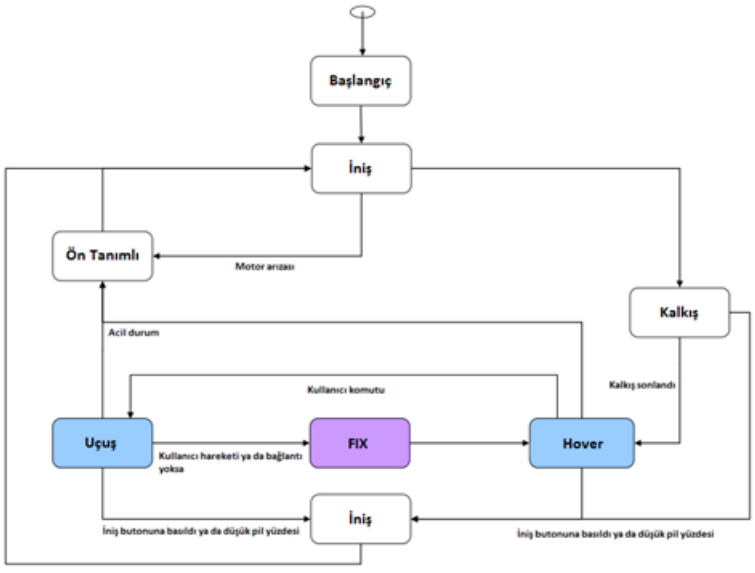
AR Drone üzerinde 2 tane kamera bulunmaktadır. Ön kamerada 90 derece açılı CMOS sensörü vardır. AR Drone görüntü verilerini otomatik encode eder ve karşı tarafa yollar. AR Drone 1.0 sürümünde QCIF (176x144) ya da QVGA (320x240) çözünürlükleri vardır, video stream frame rate ise 15 FPS'dir. AR Drone 2.0 ise her iki kamera 360p (640x360) ya da 720p (1280x720) görüntü çözünürlüğüne sahiptir, video stream frame rate ise 15 ile 30 arasında değişmektedir.



Şekil 3. 7 Ön kamera

3.3.7 Gömülü Yazılım

AR Drone içinde Linux tabanlı gerçek zamanlı bir işletim sistemi bulunmaktadır. İşletim sistemi içinde Wi-Fi bağlantı, video data sampling, video sıkıştırma, görüntü işleme, sensör bilgilerini işleme gibi iş parçacıkları yönetilir [11].



Şekil 3. 8 Uçuş algoritması

3.3.8 Wi-Fi Network ve Bağlantı

AR Drone 2.0 Wi-Fi destekleyen tüm Client cihazlar ile kontrol edilebilir.

Bağlantı için şu adımlar izlenir:

1) AR Drone, ardrone2_xxx ESSID ile bir Wi-Fi network kurar ve IP adresi atanır (192.168.1.1).
2) Kullanıcı Client cihaz ile bu ESSID ağa bağlanır.
3) Client cihaz drone DHCP Serverdan bir IP adresi talep eder.
4) AR Drone DHCP Server Client için IP atar.
5) Client cihaz AR Drone IP adres ve servis portlarına istek gönderir.

3.4 Uçuş Otomasyonu

Uçuş otomasyonları hedef tespit, tanıma ve takip sistemlerinde önemli bir faktördür. Otomasyon aracın insan kontrolünde olmadığı durumlarda çalışması için gerekli komutları veren sistemdir. AR Drone kullanmak için geliştirilen uçuş otomasyonu kameralardan alınan görüntülerle 3D model oluşturabilir ve sensörlerden alınan ölçümlerle aracın kontrolü için gerekli işlemlere karar verir. Eğer araç dış mekanlarda kullanılacaksa GPS verilerinden de yararlanılabilir, ancak iç mekanlarda bu mümkün değildir.

3.4.1 Üç Boyutlu Modelleme

3 boyutlu (3B, 3D) modelleme 3 boyutlu bir nesnenin ya da yüzeyin matematiksel gösterimini içerir. Derinlik olduğu için 3D modeller 2D modellerden daha fazla ayrıntı içerir. 3D modeller için görme teknolojisi ve lazer mesafe ölçümü ile birlikte kullanılarak iki boyutlu haritalamada mümkün olmayan nesneleri (merdiven, pencere gibi) haritalandırabilir [12].

3.4.2 SLAM

Otonom araçlar kullanarak dış ya da iç mekan haritalanması probleminin çözümünde eş zamanlı konum belirleme ve haritalama algoritmaları kullanılır. Bu algoritmalar ile

araçlarda hedef noktaya erişim, ortamın belirlenmesi ve gözlemlenmesi, engellerden sakınma gibi stratejik yol planlaması ile planlanan bu yolun izlenmesi gibi navigasyon fonksiyonları gerçeklenebilir. Bu durumda AR Drone bilinmeyen bir noktadan harekete başlayarak bulunduğu ortamın haritasını çıkarması ayrıca ortamda bulunan nesneleri etkileyebilmesi gerekmektedir. AR Drone üzerindeki kameraları kullanarak görüntüye dayalı eş zamanlı konum belirleme ve haritalama (SLAM) yapılabilir böylece GPS bağımlılığı ortadan kalkmış olur. SLAM yöntemindeki amaç bir otonom aracın engellerden sakınarak yol planlama amacıyla bulunduğu ortamın haritasını oluşturmak hem de aracın bulunduğu konumu ve durumunu eş zamanlı olarak belirlemektir. SLAM teknikleri otonom araçların belirli bir görevi gerçekleştirebilmesi için gerekli temel unsurlardandır. Bu teknikleri kullanılarak otonom araçların dinamik bir ortamdaki navigasyon kabiliyetleri arttırılabilir.

Otonom araçların konumunun belirlenmesi için içinde bulunduğu ortamın haritasına ihtiyaç vardır ve aracın içinde bulunduğu ortamın haritasını oluşturmak için konum bilgisine ihtiyaç vardır. Bu 2 problemin çözümü için genel olarak lineer kestirim teorisi kullanılmakta ve bu çerçevede özellikle olasılıksal SLAM yöntemleri kullanılmaktadır. Burada ortam içinde hareketini oluşturan kontrol girişi ile ortamdaki cisimlerden elde edilen çeşitli mesafe ölçüm bilgileri, bilinen büyüklükler olarak kabul edilir. Bu olasılıksal yaklaşımda aracın ortam ile yaptığı herbir ölçüm etkileşiminde ölçüm değerlerine ek olarak belirsizlik bilgileri de kapsanır. Olasılıksa SLAM yaklaşımında temel 2 yaklaşım vardır. Bunlar Genişletilmiş Kalman Filtreleri ile SLAM metodu ve Rao-Blackwellized parçacık filtreleri ile SLAM metodudur.

SLAM ile ilgili çalışmalara 1980'li yıllarda başlanmıştır [13]. Haritalama için en basit yaklaşım aracın konumunu mesafe sayacından konum hesabı bilgisine dayanarak kestirimi üzerine kuruludur. SLAM konusunda en iyi bilinen çalışmalara örnek olarak Cheesman ve Smith'in 1986 yılında navigasyonu yapılacak ortamın yapısının ayrık-zamanlı durum uzay gösterimiyle ifade edilmesi timeline dayanan istatistiksel haritalama çalışması verilebilir [14].

3.4.3 Görüntü Tabanlı Haritalama

Görüntü tabanlı haritalama 3D haritalama tekniklerine bir alternatif olarak gösterilebilir. Bu teknik bir yeri haritalamak için geometrik gösterimi kullanır. Photo-Realistic grafikler ve gerçek zamanlı sahne animasyonlar yaratarak yer haritalanır [12]. Panoramik görüntü ve sanal gerçeklik bunun en güzel örneklerindendir, Google Street View ise Photo-Realistic grafiklere örnektir.

3.4.4 Görsel Tabanlı Navigasyon

Otonom araç çalışmalarındaki görevlerden biri aracın çevresini görebilmesidir. Eğer navigasyon ortamında çevresinin görüntü tabanlı algılama elde edilebilirse, bu görüntü verileri sensör verileriyle birleştirilebilir. Görüntü tabanlı navigasyon etkili bir navigasyon tekniğidir çünkü bu yöntem sayesinde daha az güç harcayarak uzak mesafeden toplanan görsel veriler işlenir. Görüntü sistemlerinde sürekli bir hareketin elde edilebilmesi için örnekleme frekansı sürekliliği bozmayacak şekilde ardışıl bir görüntü sırasının yakalanabilmesi ve gözlenebilmesi gerekmektedir. Görüyle hareketin kavranabilmesi 2 açıdan önem taşımaktadır. Bunlardan birincisi, görüntü düzlemi üzerindeki izdüşümlerinden cisimlerin şekillerinin, konumlarının ve 3 boyutlu hareketlerinin kestirilebilmesidir. Diğer önemli nokta ise, algılayıcının içinde bulunduğu ortam ile ilgili bir ön bilgi sunabilmesidir [19].

3.4.5 Görsel Olmayan Navigasyon

Bu navigasyon tekniğinde ise araç üzerinde bulunan lazer mesafe ölçer, sonar ve infra-red sensörler kullanılmaktadır. Roberts et al. 2009 yılındaki çalışmada ultra-sonic ve infra-red sensör verileri kullanılarak quadrotor kontrolü sağlanmıştır [15]. Son yıllarda hızlı bir gelişim gösteren atalet sensörleri aracılığıla geliştirilen ataletsel navigasyon sistemleri hali hazırda füzelerin güdüm ve kontrolünde, yer ve hava araçlarına yönelik olarak geliştirilen uygulamalarda yaygın olarak kullanılmaktadır. Bu sistemler araçlara uygulanarak otonom sistemler alanında çalışmalar yapılmaktadır. Ataletsel navigasyon sistemleri otonom araçlarda konum belirleme, hedef izleme ve navigasyon uygulamalarında GPS donanımına

ek olarak kullanılan sistemlerdir. Sisteme ait jiroskop ve ivmeölçerler sistemin ivmelenmesini, açısal hızlanmasını ölçerek bu verileri sistemin anlık konum verilerine dönüştürmektedir [16]. Ataletsel navigasyon sistemlerinin çok maliyetli olması karşısında düşük maliyette bir çözüm GPS entegrasyonuyla oluşturulabilmektedir. Bu 2 sistemin birleştirilmesi ANS üzerindeki hataların azaltılmasına ve düşük maliyetli IMU sistemlerinin kullanılmasını sağlamaktadır. Bu sayede düşük maliyetli entegre sistemleri, yüksek maliyette gerçekleştirilen uzaktan kontrollü araçlar, otonom araçlar için geliştirilebilmektedir [16].

3.4.6 Stabilite ve Kontrol

Görüntü tabanlı algoritmalar pozisyon tahmini ve uçuş stabilizasyonu için kullanılmaktadır. Bu stabilizasyon algoritmalarının bazı uygulamalarında derinlik ölçümü için kullanılan kameralar mevcuttur [17], bu sayede araç etrafındaki nesnelere istenilen uzaklıkta durabilir. Johnson (2008) tarafından yapılan çalışmada aracın irtifa stabilizasyonu ve pozisyon kontrolü için görüntü tabanlı algoritmalar kullanılmıştır [18]. Bu alanda yapılan bir diğer çalışmada ise düşük çözünürlüklü bir on board kamera ve atalet ölçüm cihazı kullanılarak araç hareketi ve derinlik bilgisi hesaplanarak haritalama yapılmıştır [19].

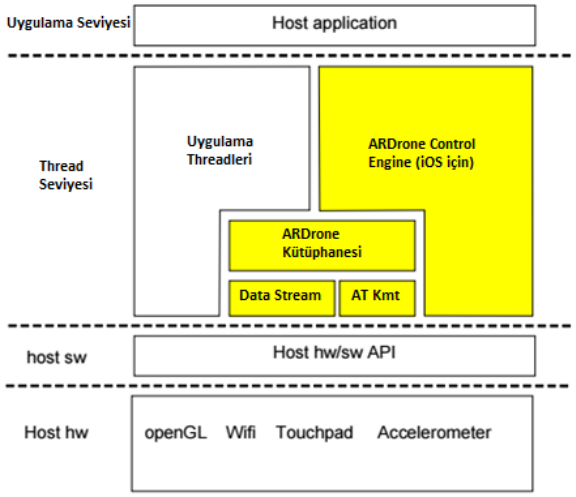
BÖLÜM 4

DRONE İLE HABERLEŞME

Bu çalışmada geliştirilecek olan yazılımların temelinde API'ler (Application Programming Interface) ve sürücüler bulunmaktadır.

4.1 Parrot AR.Drone SDK

AR Drone'larda uygulama geliştirmek için açık kaynak API mevcuttur. API içinde C ile yazılmış iOS, Android ve Linux platformlarında çalışabilen SDK (Software Development Kit) vardır.



Şekil 4. 1 Katmanlı mimari

4.2 ROS (Robot Operating System)

Robot işletim sistemi (ROS) robot uygulamaları geliştirmek için kütüphaneler ve paketler barındıran bir platformdur. Bu system açık kaynak olarak sunulmuştur. ROS ile yapılan çalışmaların sayısı gün geçtikçe artmakta ve kütüphane genişlemektedir. ROS üzerinde çalışılması için önerilen işletim sistemi Ubuntu işletim sistemidir. Bağımsız bir mimariye sahip olduğu için aynı robot üzerinde birçok farklı dil kullanılabilir.

4.3 AR.Drone Autonomy

AR.Drone Autonomy Parrot AR.Drone 1.0 ve 2.0 sürümleri için geliştirilmiş bir ROS sürücüsüdür. Bu sürücü resmi AR.Drone SDK 2.0.1 baz alınarak geliştirilmiştir [20].

4.3.1 Navigasyon Verileri

Drone üzerinden alınan veriler *ardrone/navdata* topic ile yayınlanmaktadır. Mesaj türü ise *ardrone_autonomy::Navdata* şeklinde olup şu bilgileri taşımaktadır;

- **header**: ROS mesaj başlığı
- **batteryPercent**: Drone üzerindeki pilin kalan yüzdesi (%)
- **state**: Drone'nun o anki durumu
    - **0**: Unknown
    - **1**: Inited
    - **2**: Landed
    - **3**: Flying
    - **4**: Hovering
    - **5**: Test
    - **6**: Taking off
    - **7**: Flying
    - **8**: Landing
    - **9**: Looping
- **rotX**: X ekseni etrafındaki dönüş (Sol/Sağ)
- **rotY**: Y ekseni etrafındaki dönüş (İleri/Geri)

- **rotZ**: Z ekseni etrafındaki dönüş (Derece yönelimi)
- **magX, magY, magZ**: Magnetometer çıktısı
- **pressure**: Drone üzerindeki barometer ile ölçülen basınç değeri
- **temp**: Drone üzerindeki sensör ile ölçülen sıcaklık
- **wind_speed**: Tahmini rüzgar hızı
- **wind_angle**: Tahmini rüzgar açısı
- **altd**: Tahmini yükseklik (mm)
- **vx, vy, vz**: Lineer hız (mm/s)
- **ax, ay, az**: Lineer ivme
- **tm**: Zaman bilgisi (Timestamp)

4.3.2 Kamera

AR.Drone 1.0 ve 2.0 sürümlerinde 2 kamera bulunmaktadır. Aracın ön kısmını gösteren bir ön kamera ve alt kamera. AR.Drone Autonomy sürücüsü ile Drone için 3 topic oluşturulmaktadır.

ardrone/image_raw
ardrone/front/image_raw
ardrone/bottom/image_raw

Tüm bunlar standart ROS kamera arayüzünü kullanmaktadır.

ardrone/togglecam çağrılarak o an aktif olan kamerayı değiştirebiliriz.

rosservice call /ardrone/togglecam

4.3.3 AR.Drone Komut Gönderilmesi

Drone 3 temel işlemi standart ROS mesajları (std_msgs/Empty) ile yapmaktadır. Bunlar kalkış (takeoff), iniş (land) ve reset (emergency stop) işlemleridir.

ardrone/takeoff
ardrone/land
ardrone/reset

Kalkış sonrasında aracın kontrol edilmesi için geometry_msgs::Twist mesajları kullanılmaktadır.

-linear.x: Geri

+linear.x: İleri

-linear.y: Sağ

+linear.y: Sol

-linear.z: Aşağı

+linear.z: Yukarı

-angular.z: Sola dön

+angular.z: Sağa dön

Buradaki değerler -1.0 ile 1.0 arasında olmalıdır.

4.3.4 Hover Mod

Geometry_msgs::Twist angular.x ve angular.y olmak üzere 2 değişkene daha sahiptir. Bu değişkenler auto-hover modun açılıp kapatılması için kullanılmaktadır. Bunun yanı sıra yukarıda bahsedilen 6 değişkenin hepsini 0'a eşitleyerek de auto-hover mod açılabilir. Eğer aracın auto-hover moduna geçmesi istenmiyorsa, bu durumda yukarıdaki ilk 4 değişken 0'a eşitlenerek, angular.x ve angular.y değişkenlerine 0'dan farklı bir değer verilmelidir.

4.3.5 Led Animasyon

ardrone/setledanimation çağrılarak 14 ön-tanımlı LED animasyon çalıştırılabilir.

LED animasyon parametreleri;

- uint8 type: Animasyon tipini seçmek için kullanılır [0-13] aralığında olmalıdır.
- Float32 freq: Animasyonun frekansı (Hz)
- Uint8 duration: Animasyonun devam edeceği süre (sn)

rosservice call /ardrone/setledanimation yazılarak istenilen animasyon çağrılır.

4.3.6 Flat Trim

ardrone/flattrim servisi AR.Drone aracına “Düz Zemin” komutunu gönderecektir. Bu komut ile araç düz bir zeminde olduğunu varsayarak rotasyon tahminini tekrar kalibre edecektir. Bu servis araç havadayken ya da düz bir zeminde değilken kullanılmamalıdır.

4.3.7 Terminal Üzerinden Kontrol

AR.Drone sürücüleri başarılı bir şekilde yüklendikten sonra araç terminal üzerinden komutlarla kontrol edilir [21].

Temel hareketler için komutlar;

- Kalkış

  rostopic pub -1 /ardrone/takeoff std_msgs/Empty

- İniş

  rostopic pub -1 /ardrone/land std_msgs/Empty

- Kamera Değiştirme

  rosservice call /ardrone/togglecam

- İleri Hareket

  rostopic pub -r 10 /cmd_vel geometry_msgs/Twist '{linear: {x: 1.0, y: 0.0, z: 0.0}, angular: {x: 0.0,y: 0.0,z: 0.0}}'

- Geri Hareket

  rostopic pub -r 10 /cmd_vel geometry_msgs/Twist '{linear: {x: -1.0, y: 0.0, z: 0.0}, angular: {x: 0.0,y: 0.0,z: 0.0}}'

- Sola Hareket

  rostopic pub -r 10 /cmd_vel geometry_msgs/Twist '{linear: {x: 0.0, y: 1.0, z: 0.0}, angular: {x: 0.0,y: 0.0,z: 0.0}}'

- Sağa Hareket

  rostopic pub -r 10 /cmd_vel geometry_msgs/Twist '{linear: {x: 0.0, y: -1.0, z: 0.0}, angular: {x: 0.0,y: 0.0,z: 0.0}}'

- Yükselme

  rostopic pub -r 10 /cmd_vel geometry_msgs/Twist '{linear: {x: 0.0, y: 0.0, z: 1.0}, angular: {x: 0.0,y: 0.0,z: 0.0}}'

- Alçalma

  rostopic pub -r 10 /cmd_vel geometry_msgs/Twist '{linear: {x: 0.0, y: 0.0, z: -1.0}, angular: {x: 0.0,y: 0.0,z: 0.0}}'

- Saat Yönü Dönüş

  rostopic pub -r 10 /cmd_vel geometry_msgs/Twist '{linear: {x: 0.0, y: 0.0, z: 0.0}, angular: {x: 0.0,y: 0.0,z: -1.0}}'

- Saat Yönü Tersi Dönüş

  rostopic pub -r 10 /cmd_vel geometry_msgs/Twist '{linear: {x: 0.0, y: 0.0, z: 0.0}, angular: {x: 0.0,y: 0.0,z: 1.0}}'

- Stop

  rostopic pub -r 10 /cmd_vel geometry_msgs/Twist '{linear: {x: 0.0, y: 0.0, z: 0.0}, angular: {x: 0.0,y: 0.0,z: 0.0}}'

- Kamera

  rosrun image_view image_view image:=/ardrone/image_raw

- Ön Kamera

  rosrun image_view image_view image:=/ardrone/front/image_raw

- Alt Kamera

  rosrun image_view image_view image:=/ardrone/bottom/image_raw

- Yükseklik Sensörü

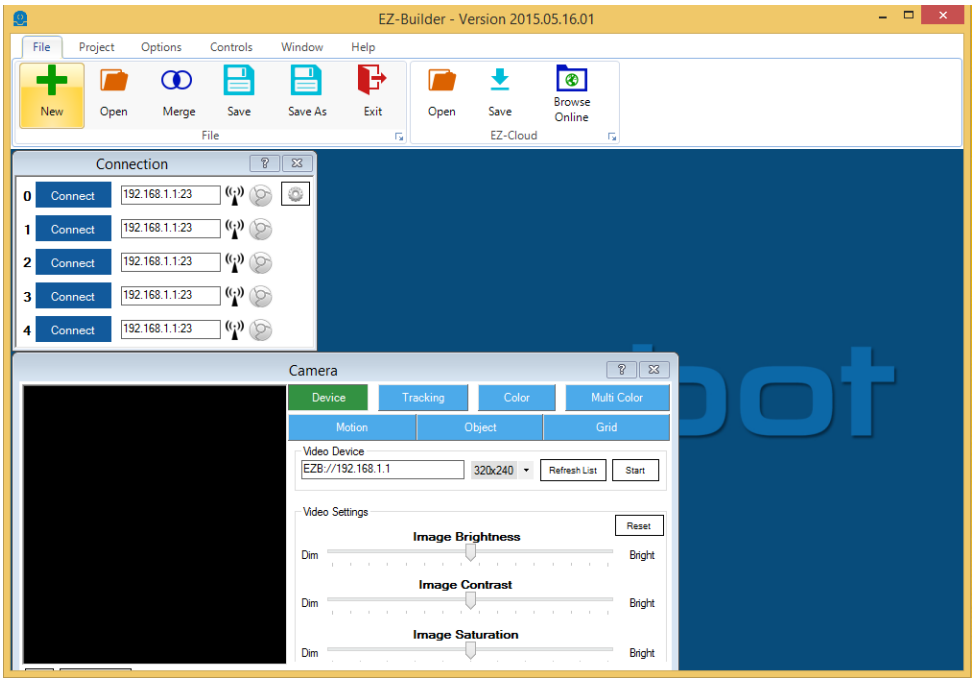
  rostopic echo /sonar_height

- Navigasyon Bilgisi

  rostopic echo /ardrone/navdata

4.3 EZ-Builder SDK

EZ-Builder robotlar için geliştirilmiş bir kontrol yazılımıdır. Birçok robot ve donanım ile uyumlu çalışan bir arayüze sahip olmasının yanı sıra farklı programlama dilleri için geliştirilmiş kütüphanelere sahiptir. Servo motor kontrolünden, görüntü işlemeye kadar birçok farklı alanda çalışmayı sağlayan bu sistem üzerinde script geliştirilmektedir. Böylece diğer programlama dillerinden bağımsız olarak da robot ve donanım kontrolü sağlanmaktadır. EZ-B script için kendine özel bir syntax kullanmaktadır, ayrıca script editörü üzerinde debugging arayüzü olduğu için test sırasında rahatlıkla izlenebilir.



Şekil 4. 2 EZ-Builder

4.3 AT Komutları

AT komutları aracın wifi üzerinden kontrol edilmesini sağlayan bir protokoldür. AT komutlarını araca gönderebilmek için AR.Drone ile kablosuz bağlantı kurulmalıdır. AT komutları 8-bit ASCII olarak kodlanmıştır, yeni satır için <CR> carriage return mevcuttur. Tüm AT komutları AT ile başlamakta ve peşinden komut adı gelmektedir, komut adı belirtildikten sonra o komuta özgü parametreler girilmektedir.

- AT*REF (input) - Kalkış, iniş, acil stop komutları
- AT*PCMD (flag, roll, pitch, gaz, yaw) - Drone hareket komutları (İstenilen değerler argüman olarak gönderilmektedir.)

$$\textbf{Args:}\quad (\quad \begin{array}{ll} \textbf{float } \textit{phi}: & \text{Left/right angle} \in [-1.0; +1.0] \\ \textbf{float } \textit{theta}: & \text{Front/back angle} \in [-1.0; +1.0] \\ \textbf{float } \textit{gaz}: & \text{Vertical speed} \in [-1.0; +1.0] \\ \textbf{float } \textit{yaw}: & \text{Angular speed} \in [-1.0; +1.0] \end{array}$$

Şekil 4. 3 Drone hareket komut argümanları

- AT*FTRIM - Flat trim için kullanılmaktadır. Bu komut yatay zemini ayarlamak için kullanılmaktadır.
- AT*CONFIG (key, value) - Drone konfigürasyonu için kullanılmaktadır.
- AT*CONFIG_IDS - Oturum, kullanıcı adı ve uygulama id ayarlamak için kullanılmaktadır.
- AT*LED – Drone üzerindeki ledleri çalıştırmak için kullanılmaktadır. İstenilen bir animasyon yapılabilmektedir (Sequence sayısı, animasyon numarası, frekans ve süre değerleri argüman olarak gönderilmektedir.)

  Drone üzerinde ön-tanımlı olan bir script bulunmaktadır ve bu script içerisinde kayıtlı led animasyonları vardır. Bu animasyonlarda led renkleri, hangi ledlerin yanacağı ve ne kadar süre yanacağı gibi bilgiler bulunmaktadır. Bu animasyonlara örnek olarak BLINK_GREEN_RED verilebilir. Bu animasyon için cycle 2 saniyedir. Dolayısıyla aşağıdaki komutu içeren paket araca gönderildiğinde, 4 saniyelik bir animasyon başlatılacaktır ve 2 kez tekrar edecektir.

Led animasyonu için kullanılan argümanlar:

1. Animasyon Adı
2. Cycle Sayısı - Eğer bu sayı 0 ise default tanımlı olan cycle değeri kullanılmaktadır.
3. Led renk ve pozisyonları - Bu argüman bit olarak ifade edilmektedir. Her bir bit ledin rengini ve pozisyonunu ifade etmektedir. (Sol Ön Yeşil | Sol Ön Kırmızı | Sağ Ön Yeşil | Sağ Ön Kırmızı | Sağ Arka Yeşil | Sağ Arka Kırmızı | Sol Arka Yeşil | Sol Arka Kırmızı)

Tanımlı led animasyonları:

0 - BLINK_GREEN_LED

1 - BLINK_GREEN

2 - BLINK_RED

3 - BLINK_ORANGE

4 - SNAKE_GREEN_RED

5 - FIRE

6 - STANDART

7 - RED

8 - GREEN

9 - RED_SNAKE

10 - BLANK

11 - RIGHT_MISSILE

12 - LEFT_MISSILE

13 - DOUBLE_MISSILE

14 - FRONT_LEFT_GREEN_OTHERS_RED

15 - FRONT_RIGHT_GREEN_OTHERS_RED

16 - REAR_RIGHT_GREEN_OTHERS_RED.

17 - REAR_LEFT_GREEN_OTHERS_RED

18 - LEFT_GREEN_RIGHT_RED

19 - LEFT_RED_RIGHT_GREEN

20 - BLINK_STANDARD

BÖLÜM 5

NESNE TAKİBİ

Bu çalışmanın ana konularından biri drone ile nesnelerin takip edilebilmesidir. Bu aşamada 2 farklı yaklaşım ile nesneleri tespit edip drone ile takip etmeyi amaçladık. Aracın ön kamerasından alınan görüntüler üzerinde görüntü işleme teknikleri kullanılarak renk tespiti ve şekil tespiti yapılabilmektedir.

5.1 Renk ile Nesne Takibi

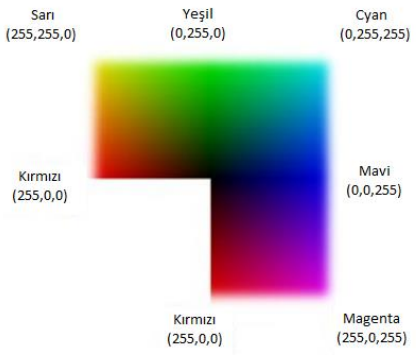
HSV renk uzayı, renkleri RGB değerlerine göre değil Hue (Renk Özü), Saturation (Doygunluk) ve Value (Parlaklık) değerlerine göre belirten bir renk uzayı türüdür. Renk özü, rengin baskın dalga uzunluğunu belirleyen açısal bir değerdir. Doygunluk rengin canlılığını belirler. Yüksek doygunluk canlı renklere neden olurken, düşük doygunluk rengin gri tonlara yaklaşmasına neden olur ve 0-100 arası değişir. Parlaklık rengin aydınlığını yani içindeki beyaz oranını belirler. 0-100 arasında bir değer alır. HSV renk uzayında ışık değişimleri kolaylıkla tanınabilmekte ve görüntü işleme sırasındaki zorluklar ortadan kaldırılabilmektedir.

HSV renk uzayı 1978 yılında Alvy Ray Smith tarafından tanımlanmıştır. Amacı RGB uzayına göre insan görü düzeneğine daha yakın bir yapı oluşturmaktadır. HSV, RGB renk uzayından doğrusal olmayan bir dönüşüm ile elde edilir. HSV uzayı ilk tanımlandığı zamanlarda konik bir biçime sahipti ancak sonraki yıllarda gerçek zamanlı geçerli koordinat denetimi için zamanın bilgisayarları yeterli olmadığından silindir biçimine dönüştürülmüştür. Konik biçimde iken aydın düzeyi azaldıkça koninin genişliği azalır, dolayısıyla insan görüsüne

uygun olarak, düşük aydınlıkta algılanabilen farklı doygunluk düzeyleri de azalırlar. Diğer yandan, silindir biçimi ile 0 aydınlık düzeyinde bile yüksek doygunluk düzeyleri tanımlanabilir ve böylece geçersiz renkler elde edilebilir. Dolayısıyla görüntü işleme uygulamalarında konik biçimi tercih edilirken, renk seçimi görevlerinde silindir biçimi kullanma eğilimini gösterir.

Şekil 5. 1 HSV renk uzayının konik ve silindirik görünümleri

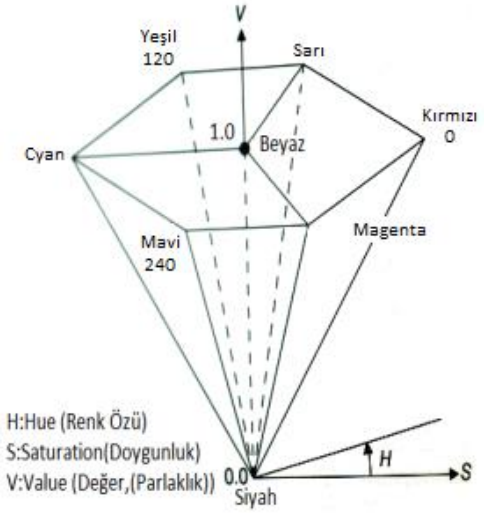
RGB görüntüsünü meydana getiren üç boyutlu matris de üçüncü boyuttaki her renk (kırmızı, yeşil, mavi) 8 bit ile temsil edilir. Buradan yola çıkarak $(2^8)^3$ = 16,877,216 renk oluşturulabilir. Dolayısıyla bir piksel 24 bit ile temsil edilir. RGB uzayında büyün renkler teorik olarak bir küpün içerisinde ya da üzerinde yer alır. Küpün köşegen notlarında gri tonlar birikir.



Şekil 5. 2 RGB köşegen noktaları

Çizelge 5. 1 RGB köşegen renk karşılıkları

	Siyah	Beyaz	Kırmızı	Sarı	Yeşil	Cyan	Mavi	Magenta
R	0	255	255	255	0	0	0	255
G	0	255	0	255	255	255	0	0
B	0	255	0	0	0	255	255	255



Şekil 5. 3 HSV değerlerinde değişimler ve etkileri

HSV uzayının konik biçimi her ne kadar silindirik haline göre bazı olumlu yanlara sahip olsa da, aydınlık ölçüsü olarak RGB değerlerinin basitçe en büyüğünün kullanılıyor olması insan görüsünün dalga uzunluğu hassasiyetlerinin dikkate alınmamasına neden olabilmektedir.

Şekil 19'da konik şeklin alt kısmı V=0 ve S=0 ise siyah, tepe noktası V=1 ve S=0 ise beyaz rengi göstermektedir. Şekilde S ile gösterilen ve 0 ile 1 arasındaki değere oran denilmektedir. S=0 iken H değerinin bir önemi yoktur. V değerini değiştirmeden S değeri azaltılırsa oluşan renge beyaz, S değeri değiştirmeden V değeri azaltılırsa renge siyah eklenebilir. Tonlama ise S ve V değerleri ile oynanarak elde edilebilir [22].

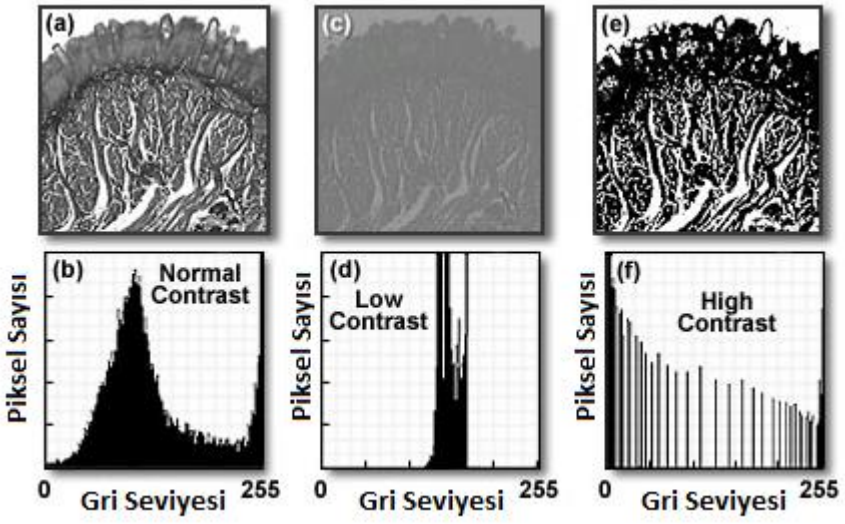
5.2 Şekil Tanıma ile Nesne Takibi

Şekil tanıma görüntü işlemenin önemli bir kısmını oluşturmakta ve birçok uygulama alanı bulunmaktadır. Plaka tanıma, trafik işaretlerini tanıma ve iris tanıma bu alanlara örnek olarak verilebilir. Şekil tanımaya yönelik olarak farklı uzaylarda çeşitli uygulamalar yapılabilmektedir. En-boy oranı ya da dairesellik gibi farklı metotlar mevcuttur. En yaygın kullanılan metot ise nesnenin uzunluğu ve genişliği arasındaki en-boy oranının hesaplandığı metottur. Bu çalışma kapsamında dairesel bir cisim takip edileceği için dairesellik değerleri kullanılarak şekil tanıma ve nesne takibi yapılmaktadır.

Sayısal görüntü, görüntünün ikili kodlama ile gösterimi olarak kısaca tariff edilebilir. Bu ikili kodlamada her hücre bir piksel olarak adlandırılır ve bir matris içinde geometric konuma karşılık gelen ton değerlerini ifade eder [23].

Sayısal görüntü gri seviye değeri olarak yalnız bir veya 0 değerini alıyorsa bu sayısal görüntüye ikili görüntü denir. Gri tonlu görüntülerde görüntü farklı gri ton değerlerinden oluşur. 0-255 arasında olmak üzere toplam 256 farklı gri ton değeri bulunmaktador. Burada 256 gri değer 1 byte olarak tanımlanır [24]. 0 değeri siyaha, 255 değeri ise beyaza karşılık gelmektedir. Bu değerler arasında gri tonlar oluşmaktadır.

- Histogram: Bir görüntüdeki her renk yoğunluğu düzeyindeki piksellerin dağılımının grafiksel gösterimine histogram adı verilmektedir. Bir görüntüde ortalama değer, standart sapma görüntüdeki gri değerlerin dağılımına ilişkin basit ölçütlerdir. Aynı şekilde görüntü histogramı da görüntüde piksellerin gri değerlerine ilişkin bağıl sıklık ölçütünü oluşturur. Histogramda yatay eksen gri değer aralığını, düşey eksen de her bir aralıktaki piksel sayısını göstermektedir.



Şekil 5. 4 Gri ton histogram

- Çevre: Çevre şeklin etrafını gösteren kapalı eğrinin uzunluğudur. Sayısal görüntülerde kullanılan komşuluk çevre için önemli bir faktördür zira dörtlü komşulukta şeklin çevresi eğri üzerinde piksellerin sayısına eşitken, sekizli komşulukta tek ve çift piksellerle aşağıdaki şekilde ifade edilir [25].

$$P = N_E + N_O \sqrt{2}$$

- Alan: Sayısal şekil için çevreye kıyasla elde edilmesi daha kolay bir veridir. Alan, belirtilen şekil içindeki toplam piksel sayısı olarak kabul edilir.
- Ağırlık Merkezi: Şekil matrisindeki her bir boyuta ait elemanların o boyuttaki konumları toplamının Alana bölünmesiyle elde edilen noktadır.
- Ağırlık Merkezine Maksimum ve Minimum Uzaklık: Şeklin ağırlık merkeziyle kendisini çevreleyen noktalar arasındaki maksimum ve minimum uzaklıkdır. Bu değerlerin oranı 1'e yaklaştıkça şekil daireselleşmeye başlamaktadır.
- Şeklin Çevresine Ortalama Uzaklığı: Şekli oluşturan her bir pikselin şeklin çevresine olan uzaklıkları toplamının toplam piksel sayısına bölünmesiyle elde edilir.
- Çap: Şeklin birbirine en uzak iki noktası arasındaki uzaklık olarak tanımlanabilir.

5.2.1 Hough Daire Dönüşümü

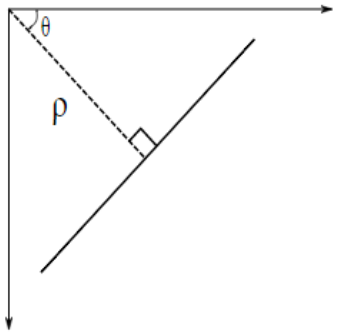
Hough dönüşümü, görüntü üzerinde yer alan doğruların ya da daha genel olarak (v,c) koordinat ve katsayı vektörü olmak üzere g(v,c) = 0 eşitliğini sağlayan şekillerin tespiti için kullanılan bir görüntü analiz yöntemidir [26]. Hough dönüşümü ilk olarak binary görüntü üzerindeki noktaların kompleks örüntülerini tespit etmek için kullanıldı [27].

Eğer bir şekil matematiksel formda ifade edilebiliyorsa, bu yöntem kullanılarak şekil görüntü içinde tespit edilebilir.

Analitik düzlemde doğru denklemi $y = mx + c$ olarak gösterilir. Doğrunun parametrik denklemi ise $p = x.cos\theta + y.sin\theta$

p: Orijin noktasından doğruya olan dikey uzaklık.

θ: Dikey çizginin yatay eksen (x-ekseni) ile yaptığı açı.



Şekil 5. 5 Analitik düzlemde doğru

Bu durumda eğer doğru orijin altından geçiyor ise p değeri pozitif bir değer olacaktır ve açı ise 180'den küçük olacaktır. Eğer Orijin üstünde kalıyor ise açı yine 180'den küçük iken bu kez p değeri negatif olacaktır.

Hough dönüşümü kenarların olası geometrik şekilleri oylaması mantığı ile çalışmaktadır. Şekilde tespiti için aşağıdaki adımlar izlenir:

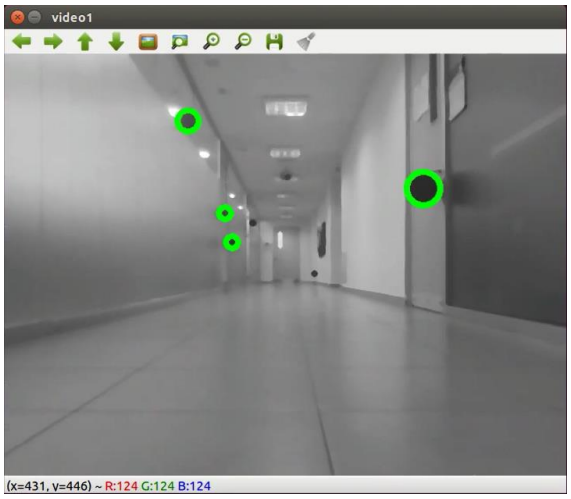
- Görüntü üzerindeki kenarlar belirlenir.
- Görüntü gri tonlara çevrilir (gray-scale).
- Her bir (p, θ) çifti için akümülatör içindeki değer 1 artırılır.

- En yüksek akümülatör değerine ulaşan şekil en çok oyu almış olacaktır.

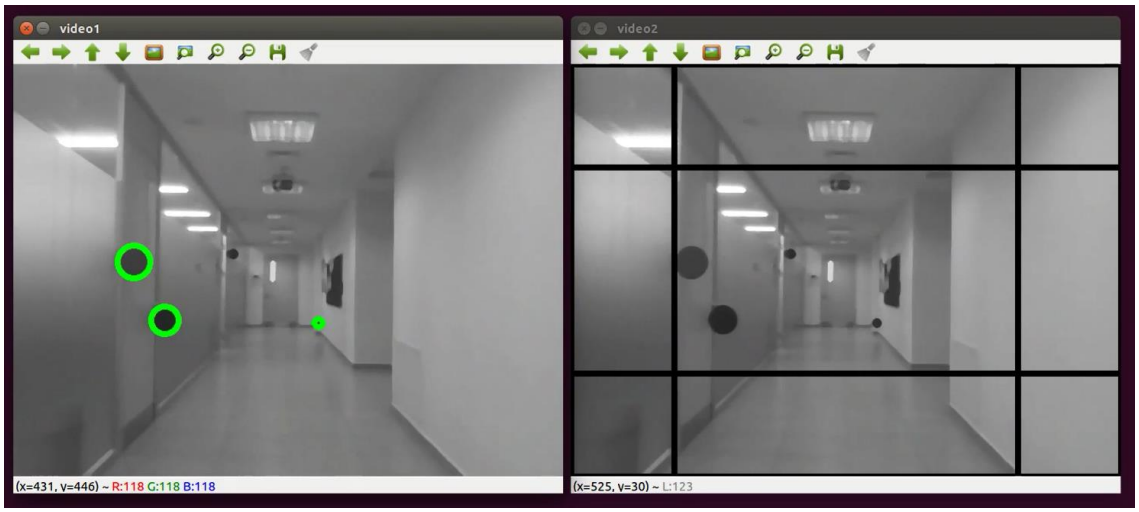
Yukarıdaki tanımlamalardan yola çıkarak bir doğru 2 değişken ile ifade edilebilir; p ve θ. Öncelikle bu değişkenler 2 boyutlu bir dizide tutulur ve başlangıç değerleri 0 yapılır. Dizideki satılar p için, sütunlar ise θ için kullanılır. Dizinin büyüklüğü ihtiyaç duyulan doğruluğa (accuracy) bağlıdır. Örneğin açı için accuracy 1 isteniyorsa, 180 adet sütuna ihtiyaç vardır. P için maksimum uzaklık görüntünün diyagonal uzunluğudur. Bu durumda 1 piksellik accuracy alınırsa, satır sayısı görüntünün diyagonal uzunluğuna eşit olacaktır.

100x100'lük bir görüntü ortasındayatay bir çizgi olduğunu varsayalım. Bu çizginin ilk noktasını alalım, bu noktaya ait olan x ve y değerlerini biliniyor. Doğru denkleminde θ yerine sırayla 0,1,2,3…180 değerlerini verip p değerlerini elde edelim. Her bir (p, θ) çifti ile akümülatör içindeki değer 1 artmaktadır. Oy alan şekillerin akümülatör değerleri de artmaya devam edecektir ve en yüksek akümülatör değerlerine ulaşan şekiller en çok oyu almış olacaktır.

Drone için hazırlanan ortama daire şeklinde levhalar yerleştirilmiştir. Drone kamerasından anlık olarak alınan görüntüler işlenerek daire şeklinde olan cisimler tespit edilip bulunan daireler yine anlık olarak yeşil bir çember ile işaretlenmektedir.



Şekil 5. 6 Drone kalkış anında tespit edilen daireler



Şekil 5. 7 Uçuş anında tespit edilen daireler

5.2.2 Kenar Algılama

Kenar algılama görüntü içindeki kenarları elde etmek için kullanılan ve görüntü işlemede öneme sahip konulardan birisidir. Bir görüntüdeki kenar, aydınlatma ya da yüzey yansımaları gibi bir görüntünün fiziksel görünüşünde oluşan önemli bir değişime karşı düşer ki bu değişim kendisini parlaklık, renk ve doku olarak gösterir. Görüntünün gri seviyelerinde ani değişikliklerin olduğu bölgeler de kenar olarak düşünülmektedir.

Prewitt [28], Sobel [29], Canny [30] ve Hilldreth [31] literatürdeki temel kenar bulma algoritmaları olarak kabul edilmektedir. Gri seviyeli görüntüler üzerinde bu algoritmalar kullanılarak yapılmış çok sayıda çalışma bulunmaktadır [32].

BÖLÜM 6

YAZILIM

Drone, gerekli kütüphaneler ile ya da AT komutları ile bilgisayar üzerinden kontrol edilebilir. Bu çalışmada ROS üzerinde klavye ile kontrol edilmesi ve araçtan gönderilen sensör verilerinin, kamera görüntülerinin aktarılması işlemi gerçekleştirilecek bunun yanı sıra Windows üzerinde C# uygulaması geliştirilecektir. Bu uygulama içinde 2 farklı yapı olacaktır, bunlardan biri API ile çalışacak diğeri ise AT komutlarını araca göndererek kontrol işlemi gerçekleştirilecektir. Linux üzerinde çalışan uygulamalar Python programlama dili kullanılarak geliştirilmiştir.

6.1 Python ile Kontrol

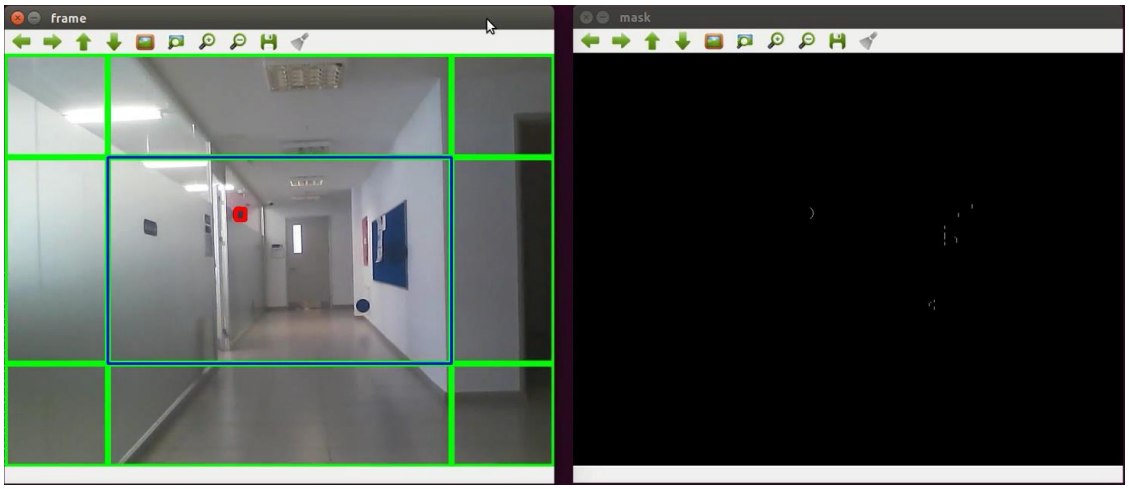
Bu aşamada linux üzerinde geliştirdiğimiz Python scriptler, terminale komut göndererek aracın klavye ile kontrol edilmesini sağlamaktadır. Terminalden komut gönderimi bölümünde anlatılan hareket ve kamera komutları burada Python scriptler ile çağrılmakta ve ekranda ön kamera ve alt kamera görüntüsü kullanıcıya gösterilmektedir. Bunun yanı sıra açılacak olan yeni terminaller ile farklı araç komutları eş zamanlı çalıştırılabilir ve çıktıları terminal üzerinde görülebilir. 2 farklı python script geliştirilmiştir, biri renk ile nesne takibi için diğeri ise şekil ile nesne takibi için.

Bu çalışmada dronenun hem klavyeden kontrol edilmesi hem de nesne tespiti ve takibi ile hareket etmesi için Python scriptler geliştirilmiştir. Script çalıştırıldığında drone kalkış işlemi gerçekleştirilmekte ve 2 kamera görüntüsü de kontrol edilmektedir. Eğer o anki konumunda kamera görüntüsünde takip edilecek nesne bulunamadıysa drone kendi

etrafında dönerek takip edilecek nesneyi aramaya başlamaktadır. İstenirse bu otonom çalışma sonlandırılarak, klavye ile kullanıcı kontrolüne geçilebilir. Bu scriptin esas kullanım amacı bir uçuş planlayıcısı kullanarak belirli bir rota üzerinde ilerlerken, tanımlı nesnenin tespit edilmesi ve bu nesnenin hangi koordinatta görüldüğü bilgisinin yer istasyonuna iletilmesidir. Bu işlemler yapılırken script içinde batarya bilgisi kontrol edilmektedir ve belli bir değerin altına düştüğünde şarj olabileceği bir platform inmesi amaçlanmaktadır. Çevre ya da sınır güvenliği oluşturmak için drone uçuş planı hazırlanarak ve batarya bilgisi kontrol edilerek daha efektik bir gözetleme sistemi kurulması amaçlanmaktadır.

6.1.1 Renk ile Nesne Takibi

Python programlama dilinde geliştirdiğimiz script ile araç üzerinde bulunan kameradan alınan görüntüler işlenerek aracın hareket etmesi sağlanmıştır, belirlenen renkteki cismin araç tarafından takip edilmesi amaçlanmıştır. Bu işlemler için kameradan alınan görüntüler HSV renk uzayına çevrilmiştir.



Şekil 6. 1 RGB - HSV

Yukarıdaki şekilde görüldüğü üzere kameradan aldığımız görüntüleri 9 kareye bölüyoruz. Video feed içinde tespit edilen kırmızı renkli nesnenin etrafına çizilen dikdörtgenin bu parçalar üzerine geçmesiyle dronenun hareket ettirilmesi sağlanmaktadır. Araç üzerinde 2

kamera bulunmaktadır, ön kameradan alınan video feedi kullanarak dronenun yatay (ileri, geri, sağ, sol) ve dikey (yukarı, aşağı) hareket etmesi sağlanmaktadır.

Şekil 6. 2 Uçuş anında renk tespiti

Bu hareketleri inceleyecek olursak, aracın ön kamera görüntülerini kullanarak hareket etmesi için tespit edilen nesnenin etrafına çizilen dikdörtgenin dikey kenar uzunluğunun yarısından fazlası 2 numaralı kutu içerisindeyse bu durumda araca yukarı çıkması için gerekli komut verilmekte ve bu işlem nesne etrafına çizilen dikdörtgenin dikey kenar uzunluğunun yarısından fazlası 5. kutu içinde olana kadar devam ettirilmektedir. Böylece nesnenin aracın yukarısında olduğu tespit edilerek drone yukarı hareket edecektir. Eğer aynı tespit edilen nesnenin etrafına çizilen dikdörtgenin dikey kenar uzunluğunun yarısından fazlası 8 numaralı kutu içerisindeyse bu durumda araca aşağı inmesi için gerekli komut verilmekte ve bu işlem nesne etrafına çizilen dikdörtgenin dikey kenar uzunluğunun yarısından fazlası 5. kutu içinde olana kadar devam ettirilmektedir. Sağa ve sola dönüş hareketlerinin kontrolü için ise 4 ve 6 numaralı kutular kontrol edilmektedir. Takip edilen nesne 5. kutu içerisinde ise bu aşamada nesne ile araç arasındaki mesafeye göre ileri-geri yapması amaçlanmıştır. Eğer takip edilen nesne etrafına çizilen alan 5. kutu alanının ¾'ünden küçük ise ileri gitme büyük ise geri gitme fonksiyonları çağrılmaktadır.

Aracın nesneyi takip etmesinin yanısıra belirli koordinatlar üzerinde uçarken, o güzergah üzerinde nesnenin tespit edildiği bilgisi kullanıcıya verilmektedir. Araç nesneyi tespit ettiği

an yer istasyonunda tarih ve konum bilgisi ile birlikte o anki kamera görüntüsü JPG formatında kaydedilmektedir. Bu işlem yakalanan tüm nesneler için uçuş boyunca tekrarlanmaktadır.

6.1.2 Şekil Tanıma ile Nesne Takibi

Python programlama dilinde geliştirdiğimiz script ile araç üzerinde bulunan kameradan alınan görüntüler işlenerek aracın hareket etmesi sağlanmıştır, belirlenen şekildeki (daire) cismin araç tarafından takip edilmesi amaçlanmıştır.

Şekil 6. 3 Uçuş anında dairesel cisimlerin yakalanması ve aracın yönlendirilmesi

Buradaki hareket işlemleri renk takibindeki işlemler ile aynıdır. Yapılan işlemlerin logları tarih bilgisiyle birlikte tutulmakta ve ayrıca konsol üzerinde anlık olarak kullanıcıya gösterilmektedir. Renk takibi scriptinde olduğu gibi araç belirli koordinatlar üzerinde uçarken, o güzergah üzerinde nesne tespit edilirse, kullanıcıya bilgi verilmekte ve tarih-konum bilgileriyle birlikte o anki kamera görüntüsü kaydedilmektedir.

6.1.3 Şekil Tanıma ile Renk Takibi Karşılaştırma

2 farklı görüntü işleme yöntemini karşılaştırmak için hazırlanan ortamda hem takip edilecek renklere hem de takip edilecek şekillere sahip nesneler bulunmaktadır.

Nesnelerin boyutları ve renk tonları aynıdır. Bu yöntemlerin karşılaştırılmasında önemli rol oynayan parametrelerin başında süre gelmektedir. Hazırlanan ortamda hava aracı aynı mesafeye yaklaşık olarak aynı sürede gideceği için bu deney tekrarlanarak elde edilen verilerden ortalama değerler bulunmakta ve çıkarım yapılmaktadır. Elde edilen verilerin büyük bir kısmı scriptlerin çalışma süresi boyunca yazdığı log dosyalarından oluşmaktadır. Kameradan alınan her görüntü hakkındaki bilgi log dosyasına zaman bilgisi ile birlikte yazılmaktadır. Bu bilginlerin içinde o an istenilen renk aralığının ya da şeklin yakalanıp yakalanmadığı bilgisi bulunmaktadır. Test ortamı araç çalışmaya başladığı andan itibaren iniş yaptığı noktaya kadar her an kamerada bir nesne görebilecek şekilde tasarlandı. Dolayısıyla loglara nesne bulunamadığına dair bir veri eklendiğinde aslında bu durum görüntü işleme algoritmasının o an o nesne için başarısız sonuçlandığını göstermektedir.

Şekil 6. 4 Test uçuş ortamı

Test ortamı hazırlandıktan sonra bu çalışma kapsamında toplam 12 uçuş yapılmıştır. Bu uçuşların her birinin süresi yaklaşık 90 saniye, uçuş mesafesi ise yaklaşık 50 metredir. 2 farklı görüntü işleme algoritmasının karşılaştırmasını yapmayı amaçladığımız bu testte 12 uçuşun 6 tanesi renk tanıma algoritması çalıştırılarak geri kalan 6 uçuş ise şekil tanıma

algoritması çalıştırılarak yapılmıştır. Her bir algoritma için yapılan 6 uçuşun 3 tanesi aydınlık ortamda, 3 tanesi ise daha karanlık bir ortamda gerçekleştirilmiştir. Böylece bu algoritmalardan hangisinin ışığın az olduğu ortamlarda daha iyi sonuç vereceği görülecektir.

Aşağıdaki tabloda (Çizelge 6.1) yapılan uçuşlar ve şekil tanıma algoritmasının ölçümleri yer almaktadır.

Çizelge 6. 1 Şekil tanıma ile yapılan testler

Test Uçuş Numarası	**Ortam**	**Toplam Ölçüm Sayısı**	**Şeklin Tespit Edildiği Ölçüm Sayısı**
1.Uçuş	Karanlık	1490	1007
2.Uçuş	Karanlık	1939	1377
3.Uçuş	Karanlık	976	700
4.Uçuş	Aydınlık	1119	913
5.Uçuş	Aydınlık	825	692
6.Uçuş	Aydınlık	807	662

Yukarıdaki tabloda (Çizelge 6.1) elde edilen ölçümlerden yola çıkarak bu yönteme ait olan başarı yüzdeleri aşağıdaki tabloda yer almaktadır (Çizelge 6.2) Başarı oranları şeklin tespit edildiği ölçüm sayısı, toplam ölçüm sayısına bölerek hesaplanmaktadır. 6 test uçuşu için ayrı ayrı hesaplanan başarı oranlarında en düşük değer 67.58 iken en yüksek değer 83.37 olarak hesaplanmıştır.

Çizelge 6. 2 Şekil tanıma başarı oranları

Test Uçuş Numarası	Başarı Oranı (%)
1.Uçuş	67.58
2.Uçuş	71.01
3.Uçuş	75.02
4.Uçuş	81.59
5.Uçuş	83.37
6.Uçuş	82.03

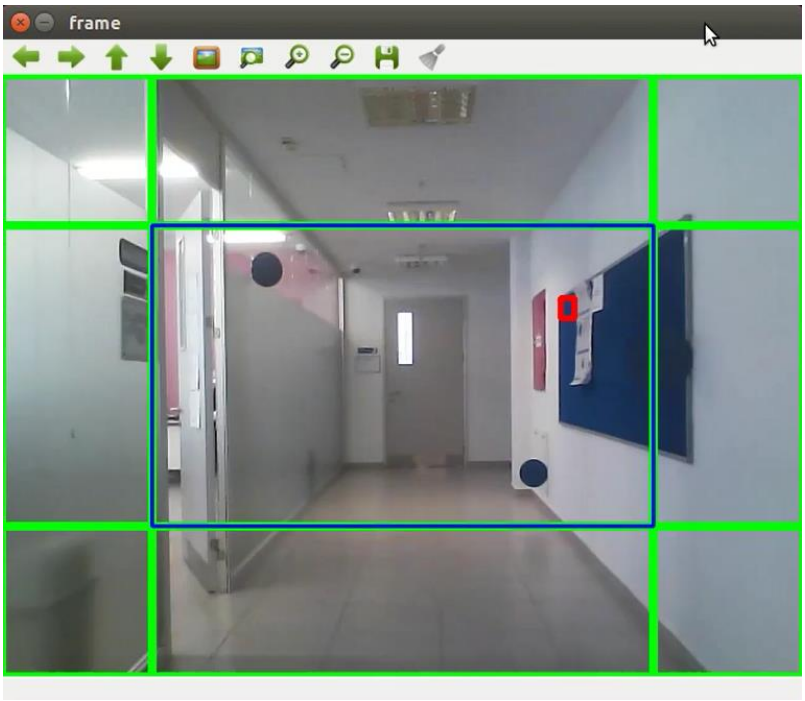
Aşağıdaki tabloda (Çizelge 6.3) yapılan uçuşlar ve renk tanıma algortmasının ölçümleri yer almaktadır.

Çizelge 6. 3 Renk tanıma ile yapılan testler

Test Uçuş Numarası	Ortam	Toplam Ölçüm Sayısı	Rengin Tespit Edildiği Ölçüm Sayısı
7.Uçuş	Karanlık	436	239
8.Uçuş	Karanlık	678	407
9.Uçuş	Karanlık	1093	643
10.Uçuş	Aydınlık	342	299
11.Uçuş	Aydınlık	334	322
12.Uçuş	Aydınlık	468	384

Yukarıdaki tabloda (Çizelge 6.3) elde edilen ölçümlerden yola çıkarak bu yönteme ait olan başarı yüzdeleri Çizelge 6.4'te yer almaktadır.

Renk tanıma algoritması ile yapılan ölçümlerde aydınlık ortamda daha başarılı sonuçlar elde edilmiş gibi gözükse de ortama aynı renkten farklı nesneler yerleştirilmiştir. Bu durumda istediğimiz rengi yakalamakta ancak istediğimiz nesnenin dışındaki nesneleri de yakalamaktadır. Yani bu algoritmadaki sonuçlarda false-positive ölçümler de bulunmaktadır.



Şekil 6. 5 False-positive ölçüm

Yukarıdaki görsel (Şekil 6.5) renk tanıma algoritması çalıştırılırken elde edilen false-positive ölçümlerden birini içermektedir. Sağ tarafta bulunan pano tespit etmek istediğimiz nesne ile aynı renk aralığında bulunduğu için algoritma panoyu nesne olarak yakalamaktadır.

Çizelge 6. 4 Renk tanıma başarı oranları

Test Uçuş Numarası	Başarı Oranı (%)
7.Uçuş	54.81
8.Uçuş	60.02
9.Uçuş	58.82
10.Uçuş	87.42
11.Uçuş	96.40
12.Uçuş	82.05

Çizelge 6.5'te iki yöntemin başarı oranlarının ortalamaları alınarak karşılaştırılması yapılmaktadır.

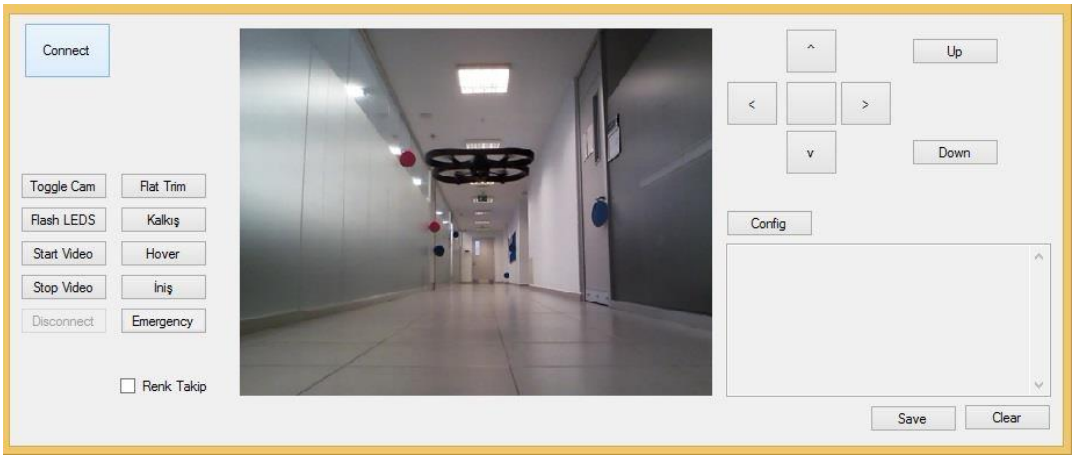
Çizelge 6. 5 Başarı oranlarının karşılaştırılması

Algoritma	Ortam	Ortalama
Şekil Tanıma	Karanlık	71.20
Şekil Tanıma	Aydınlık	82.33
Renk Tanıma	Karanlık	57.88
Renk Tanıma	Aydınlık	88.62

6.2 C# ile Windows Üzerinde Kontrol

Python programlama dili bir çok platform üzerinde (OSx, Linux, Windows) sorunsuz çalışmaktadır. Scriptler, geliştirme ortamı ve gerekli kütüphaneler kurularak çalıştırabilir. Bu çalışmanın Windows işletim sistemi üzerinde çalışması için yani aracın kontrolü ve

görüntü işleme algoritmaları için C# programlama dili kullanılarak bir yazılım geliştirilmiştir. Bu yazılım tıpkı Linux üzerinde olduğu gibi 2 farklı şekilde geliştirilmiştir. Bunlardan biri kütüphane kullanılarak (EZ-Builder) aracın fonksiyonlarının kullanılabilmesi bir diğeri ise kablosuz ağ üzerinden bir bağlantı oluşturularak AT komutlarının araca gönderilmesidir.



Şekil 6. 6 Windows ortamında geliştirilen uygulama

Aşağıdaki görselde AT komutlarının araca gönderilmesini gösteren bir kod parçacığı bulunmaktadır. Bu kod parçacığı incelenirse aracın ip adresine 5556 port üzerinden bağlanarak,

*AT*CONFIG=1,\"control:altitude_max\",\"200\rAT*REF=101,290718208\r"*

Komutu gönderilmektedir. Bu komut ile aracın kalkış yüksekliği 200 cm olarak ayarlanıp kalkış yapılmaktadır.

```
Socket sending_socket = new Socket(AddressFamily.InterNetwork, SocketType.Dgram, ProtocolType.Udp);
IPAddress send_to = IPAddress.Parse("192.168.1.1");
IPEndPoint sending_end_point = new IPEndPoint(send_to, 5556);
string buff1 = "AT*CONFIG=1,\"control:altitude_max\",\"200\rAT*REF=101,290718208\r";
byte[] buff2;

buff2 = Encoding.ASCII.GetBytes(buff1);
sending_socket.SendTo(buff2, sending_end_point);
```

Şekil 6. 7 AT komutları

EZ-Builder yazılımının C# programlama dili için geliştirdiği SDK kullanılarak da aracı control etmek mümkündür. Şekil 6.7'de bulunan kod parçacığı ile aracın hareket etmesi ve ardından havada asılı kalması (hover) sağlanmaktadır.

```
ezB_Connect1.EZB.ARDrone.SetProgressiveInputValues(0, 0, 0, moveSensitivivivity);
System.Threading.Thread.Sleep(moveSleepTime);
ezB_Connect1.EZB.ARDrone.Hover();
```

Şekil 6. 8 EZ-Builder ile kontrol

BÖLÜM 7

SONUÇ VE ÖNERİLER

Drone, uygun fiyatlara satılan ve herkesin erişimine açık olan bir teknolojidir. Araçlar, geliştirilen yazılımlarla farklı görevleri yerine getirmesi sağlanabilmektedir. Bu çalışma kapsamında geliştirilen kontrol istasyonuna farklı görüntü işleme teknikleri uygulanarak bulunmak istenen nesneler çeşitlendirilebilir ve başarı yüzdeleri arttırılabilir. Aracın konrol işlemleri resmi ya da resmi olmayan sürücü ve SDK ile yapılmasının yanı sıra belirli kontrol parametrelerinin araca gönderilerek yapılması da mümkündür. Bunun en büyük artısı 3.parti yazılımların kullanılmasınının önüne geçilmesidir.

Bu çalışma kapsamında geliştirilen yazılımlarda kullanılan görüntü işleme teknikleri farklı araçların görüntüleri üzerinde de kullanılabilmektedir. Bu sayede diğer araçların mevcut kontrol yöntemleri kullanılırken bir yandan bu yazılımlar ile nesne takibi yapılabilmesi mümkündür.

Yazılımın, ilerleyen aşamalarda geliştirilerek çoklu araçların kontrolünde kullanılması planlanmaktadır. Böylece araçların nesneleri farklı kamera ve sensörler ile takip edebilmeleri ve birbirileri arasında haberleşebilmeleri amaçlanmaktadır.

Mevcut araçların üzerine farklı sensör ve kameralar yerleştirilerek farklı tiplerde verileri almak mümkün. Örneğin AR.Drone üzerine bir termal kamera yerleştirilerek termal görüntüler elde edilebilir. Ancak bu tip fazla donanımların eklenmesi için aracın kaldırabileceği maksimum ağırlık hesaplanmalıdır ve gerekiyorsa aracın pil ve motor gibi mevcut donanımlarında da değişiklik yapılmalıdır. Yazılımın ilerletilmesi ve çalışmanın kapsamının genişletilmesi için ülkemizde akademik çalışmalar ile geliştirilen araçların kullanılması planlanmaktadır. Bu sayede hem yerli bir ürün ortaya çıkarılacak hem de geliştireceğimiz farklı araçların için bir standart haberleşme protokolü geliştirilebilir.

KAYNAKLAR

[1] Sassen, S. ve Uhleman, P., (2003). Quattrocopter Unique Micro-Aerial Vehicle, European Aeronautic Defence and Space Company Corporate Research Centre.

[2] Pounds, P., Mahony, R. ve Gresham, J., (2004). Towards Dynamically-Favourable Quad-Rotor Aerial Robots, Australian National University, Canberra, Australia.

[3] Chen, M. ve Huzmezan, M., (2003). A Combined MBPC/2 DOF H∞ Controller for a Quad-Rotor UAV, Department of Electrical and Computer Engineering University of British Columbia, Vancouver, Canada.

[4] Hoffman, G., Dostal, D., Waslander, S., Jang, J. ve Tomlin, C., (2004). "Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control (STARMAC)", Digital Avionics Systems Conference, Salt Lake City, UT.

[5] Altuğ, E., (2003). "Vision Based Control of Unmanned Aerial Vehicles with Applications to an Autonomous Quadrotor", Ph.D. Thesis, University of Pennsylvania.

[6] Leishman, J., (2000). Principles of Helicopter Aeordynamics, Cambridge University Press.

[7] Danko, T.W., Kellas, A. ve Oh, P.Y., (2005). "Robotic Rotorcraft and Perch-and-stare: Sensing Landing Zones and Handling Obscurants", Proceedings of the 12th International Conference on Advanced Robotics.

[8] Morris, S.J., (1997). "Design and Flight Test Results for Micro-sized Fixed-wing and VTOL Aircraft", Proceedings of the First International Conference for Emerging Technologies of Micro Air Vehicles, Atlanta.

[9] Piskorski, S., Brulez, N., Eline, P. ve D'Haeyer, F., (2012). AR Drone Developer Guide SDK 2.0, Parrot.

[10] Dijkshoorn, N., (2012). "Simultaneous Localization and Mapping with AR Drone", Universiteit van Amsterdam.

[11] Bristeau, P.J., Callou, F., Vissière, D. ve Petit, N., (2011). "The Navigation and Control Technology Inside The AR.Drone Micro UAV", 18th IFAC World Congress, Milano.

[12] Biber, P., Andreasson, H., Duckett, T. ve Schilling, A., (2004). "3D Modelling of Indoor Environments by a Mobile Robot with a Laser Scanner and Panoramic Camera", Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2004).

[13] Grimson, W.L. ve Lozano-Perez, T., (1985). "Recognition and Localization of Overlapping Parts from Sparse Data in Two or Three Dimensions", IEEE International Conference on Robotics and Automation, Massachusetts.

[14] Smith, R.C. ve Cheeseman, P., (1986). "On the Representation and Estimation of Spatial Uncertainty", The International Journal of Robotics Research, 5(4): 56-68.

[15] Roberts, J.F., Stirling, T.S., Zufferey, J.C. ve Floreano, D., (2009). "2.5D Infrared Range and Bearing System for Collective Robotics", Proceedings of the International Conference on Intelligent Robots and Systems.

[16] Walchko, K.J., (2002). "Low Cost Inertial Navigation: Learning to Integrate Noise and Find Your Way", Msc Thesis, University of Florida, Florida.

[17] Bills, C., Chen, J. ve Saxena, A., (2011). "Autonomous MAV Flight in Indoor Environments Using Single Image Perspective Cues", International Conference on Robotics and Automation.

[18] Johnson, N.G., (2008). "Vision-assisted Control of a Hovering Air Vehicle in an Indoor Setting", Brigham Young University, Department of Mechanical Engineering.

[19] Kendoul, F., Fantoni, I. ve Nonami, K., (2009). "Optic Flow-based Vision System for Autonomous 3D Localization and Control of Small Aerial Vehicles", Robotic and Autonomous System, 57: 591-602.

[20] Monajjemi, M., ARDrone Autonomy, http://wiki.ros.org/ardrone_autonomy, 23 Nisan 2015.

[21] Huang, H. ve Sturm, J., tum_simulator Package, wiki.ros.org/tum_simulator, 12 Mayıs 2014.

[22] Jack, K., (2007). Video Demystified: A Handbook for The Digital Engineer, Burlington, MA: Elsevier.

[23] Lu, G. ve Sajjanhar, A., (1999). "Region-based Shape Respresentation and Similarity Measure Suitable for Content-based Image Retrieval", Multimedia Systems, 7: 165-174.

[24] Gonzalez, R.C., Woods, R.E. ve Eddins, S.L., (2009). Digital Image Processing Using MATLAB, II. Baskı, Gatesmark Publishing.

[25] Jain, A.K., (1989). Fundamentals of Digital Image Processing, I. Baskı, N.J.: Prentice Hall.

[26] Gonzalez, R.C. ve Woods, R.E., (2002). Digital Image Processing, II. Baskı, Upper Saddle River, NJ: Prentice Hall.

[27] Hough, P.V., US Patent 3069654, 1962.

[28] Lipkin, B.S. ve Rosenfeld, A., (1970). Picture Processing and Psychopictorics, Orlando, FL, Academic Press.

[29] Sobel, E.I., (1970). Camera Models and Machine Perception, Stanford, CA: Stanford University.

[30] Canny, J., (1986). "A Computational Approach to Edge Detection", IEEE Transactions on Pattern Analysis and Machine Intelligence, 8: 679-700.

[31] Marr, D. ve Hildreth, E., (1980). "Theory of Edge Detection", Proceedings of the Royal Society of London, Series B, Biological Sciences.

[32] Basu, M., (2002). "Gaussian-Based Edge-Detection Methods," IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews, 32(3): 252-260.

[33] ABD Savunma Bakanlığı, (2005). Unmanned Aircraft Systems Roadmap 2005-2030, Office of the Secretary of Defense.

[34] Xu, R. ve Özgüner, U., (2006). "Sliding Mode Control of a Quadrotor Helicopter", Proceedings of the IEEE 45th Conference on Decision & Control, San Diego, USA.

[35] Trucco, E. ve Verri, A., (1998). Introductory Techniques for 3-D Computer Vision, Prentice Hall.

[36] Gonzalez, R.C. ve Woods, R.E., (2007). Digital Image Processing, III. Baskı, Upper Saddle River, NJ, Prentice Hall.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Anıl SEZGİN

Doğum Tarihi ve Yeri : 29.07.1985 İstanbul

Yabancı Dili : İngilizce

E-posta : anil.sezgin@yahoo.com

ÖĞRENİM DURUMU

Derece	**Alan**	**Okul/Üniversite**	**Mezuniyet Yılı**
Lisans	Bilgisayar Müh.	İstanbul Arel Üni.	2013
Lise	Fen	Büyükşehir Anadolu L.	2003

İŞ TECRÜBESİ

Yıl	**Firma/Kurum**	**Görevi**
2017	Utis Güvenlik Sistemleri (Sensormatic)	Yazılım Mühendisi
2016	Proline Bilişim Sistemleri	Yazılım Uzmanı
2014	İstanbul Esenyurt Üniversitesi	Araştırma Görevlisi
2013	Karash Yazılım	Yazılım Uzmanı

YAYINLARI

Makale

1. Yılmaz, R., Sezgin, A., Kurnaz, S. ve Arslan, Y.Z., (2017). “Object-Oriented Programming in Computer Science”, Encyclopedia of Information Science and Technology, 4: 7470-7480.
2. Sezgin, A., Yavuz, S., (2015). “Color-Based Object Tracking for Unmanned Aerial Vehicles”, International Journal of Electronics, Mechanical and Mechatronics Engineering (IJEMME), 5(1): 881-890.