T.C.
İSTANBUL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

KARGO TAŞIMACILIĞINDA İNSANSIZ HAVA ARAÇLARININ (DRONE) HEDEFE ULAŞMASINA YÖNELİK YENI BİR YAKLAŞIM

Nauryzkhan DEBISHEV

Akıllı Ulaşım Sistemleri Anabilim Dalı

Akıllı Ulaşım Sistemleri Programı

DANIŞMAN
Prof. Dr. Abdullah OKUMUŞ

Aralık, 2021

İSTANBUL

Bu çalışma, 3.01.2022tarihinde aşağıdaki jüri tarafından Akıllı Ulaşım Sistemleri Anabilim Dalı, Akıllı Ulaşım Sistemleri Programında Yüksek Lisans tezi olarak kabul edilmiştir.

Tez Jürisi

Prof. Dr. Abdullah OKUMUŞ (Danışman)
İstanbul Üniversitesi
Mühendislik Fakültesi

Doç. Dr. Erkan ÇELİK
İstanbul Üniversitesi
Ulaştırma ve Lojistik Fakültesi

Doç. Dr. Muhammet GÜL
Munzur Üniversitesi
Mühendislik Fakültesi

ÖNSÖZ

Uzun süren Yüksek lisans eğitim yolculuğu süresince, maddi ve manevi desteklerini esirgemeyen tüm hocalarıma, tüm süreçte ve tez aşamasında gece-gündüz emeği geçen sayın Prof. Dr. Abdullah OKUMUŞ hocama, tüm teknik destekleri ile omuz omuza, deney ve teknik çalışmalarımızda yardım eden Doç. Dr. Erkan ÇELİK hocamıza şükranlarımız sonsuz. Son olarak veri toplama sürecinde tez çalışmama yaptıkları katkılarından dolayı Nurzhan AMANTAY' a minnetlerimi sunuyorum. Her zaman destekleyen sevgili dostlarımıza teşekkürlerimizi bir borç biliriz.

Aralık 2021 Nauryzkhan DERBISHEV

İÇİNDEKİLER

Sayfa No

ŞEKİL LİSTESİ

TABLO LİSTESİ

SİMGE VE KISALTMA LİSTESİ

Simgeler **Açıklama**

Kısaltmalar **Açıklama**

İHA : İnsansız Hava Aracı
FOV : Görüş Alanı
ESC : Elektronik Hız Kontrol Cihazı
PDB : Dron Güç Dönüştürücü Modülü
FC : Dron Uçuş Kontrol Kartı
NMPC : Doğrusal Olmayan Model Öngörücü Kontrol

ÖZET

YÜKSEK LİSANS TEZİ

KARGO TAŞIMACILIĞINDA İNSANSIZ HAVA ARAÇLARININ (DRONE) HEDEFE ULAŞMASINA YÖNELİK YENI BİR YAKLAŞIM

Nauryzkhan DERBİSHEV

İstanbul Üniversitesi

Lisansüstü Eğitim Enstitüsü

Akıllı Ulaşım Sistemleri Anabilim Dalı

Danışman : Prof. Dr. Abdullah OKUMUŞ

İnsansız hava araçları (IHA) azalan bedelinden dolayı birçok sektörde kullanılmaktadır. En mühim sektörlerinden biri kargo taşımacılığıdır. Son yıllarda teslimat ve lojistik sektöründe insansız hava araçları uygulaması çeşitli araştırma projelerinin odak noktası haline geldi. Ticari kullanım içeren, dron tasarlayan ve üreten yaklaşık 70'ye şirket var. Teknoloji ilerledikçe, ticari şirketlerin kargo taşımacılığı endüstrisinde İHA'ların kullandığı ortaya çıkmaktadır. Bekleme sürelerini azaltmayı ve insan gücü bedelini bertaraf etmeyi amaçlayan bu çözüm, yeni bir teslimat vasıta namına lojistik prosesinde yerini almaktadır. Otoyol yoğunluğu, siparişin alıcılara teslim edilmesini geciktiriyor ve artan CO^2 emisyonları havaya ve ekonomiye zararını vermektedir. Standart bir dron platformu yoktur, kurallar ve düzenlemeler henüz iyi planlanmamıştır, bu nedenle simülasyon ve test şiddetle tavsiye edilir. Şu anda çoğu insansız hava aracı yıldırım çarpması ve yağmur gibi kötü havalarda uçmuyor, çünkü bunu yapacak şekilde tasarlanmamışlar. Kargoların dron'lar ile iç mekânda taşınması geride kalıyor. Bu hem yasal kısıtlamalara hem de eksik teknik yeniliklere bağlanabilir. Ayrıca, hem bir dron'un uçuş sırasında temel takibi hem de kargoyu alması ve indirmesi için gereken hassas takibi için uygun bir takip sistemi sunulmalı. Bu aşamada, nakliye güzergâhında son aşaması olarak kabul edilen

ve umumiyetle depodan ya da satıcıdan son alıcıya taşımayı belirten son aşamadaki teslim, birçok nedenden dolayı planlama bakış açısıyla gerçekleştirilir. Siparişin şekil ve ağırlık farklılıkları, ayrımlı teslimat yerleri, uzak mesafeler ve siparişi alıcının istediği vakitte teslim etme ihtiyacı gibi kısıtlamalar olabildiğince zor ve pahalıdır. Gizlilik, güvenlik, çevre ve teknoloji konularının hala araştırılması gerekirken, dron'lar, hızlı teknolojik gelişme ve düzenleyici gelişme ile ürün teslimatının son aşamasında yaygın olarak kullanılıyor gibi görünüyor.

Aracın hemen önündeki engelleri tespit etme sorununun giderilmesi, dron'lar için dinamik ortamlarda havadan çarpışmadan kaçınmaya yönelik yeni bir yaklaşımın tasarlanması, statik veya hareketli engellerle dolu bir çalışma alanında güvenli bir şekilde gezinmek için yeni bir model yaklaşımı tasarlanması, birden fazla engelin dinamiklerini tahmin etme ve gerçek zamanlı olarak bunlardan kaçınma yeteneğine sahip olması beklenmektedir.

Bu tezin maksadı, dron kullanarak, önündeki engelleri kaldırılıp, hızlı ve doğa dostu teslimatı sağlayarak alıcı memnuniyetini artırmak ve nakliye maliyetlerini azaltmak ve yukarıda bahsi geçen sorunlara çözüm önerileri getirmektir.

Aralık 2021, 90 sayfa

Anahtar kelimeler: Kargo Taşımacılığı, İnsansız Hava Aracı, Simülasyon, Gazebo.

SUMMARY

M.Sc. THESIS

A NEW APPROACH FOR DRONES İN REACHİNG THE DESTİNATİON IN CARGO TRANSPORT

Nauryzkhan DERBISHEV

Istanbul University

Institute of Graduate Studies

Department of Intelligent Transport Systems

Supervisor : Prof. Dr. Abdullah OKUMUŞ

Unmanned aerial vehicles (UAV) are used in many sectors due to their decreasing cost. One of the most important sectors is cargo transportation. In recent years, the application of unmanned aerial vehicles in the delivery and logistics sector has become the focus of various research projects. There are about 70 companies that design and manufacture drones for commercial use. As technology advances, it appears that commercial companies are using UAV's in the cargo transportation industry.

This solution, which aims to reduce waiting times and eliminate the cost of manpower, takes its place in the logistics process on behalf of a new delivery vehicle. Highway congestion delays order delivery to buyers, and increased CO^2 emissions are damaging the air and the economy. There is no standard drone platform, the rules and regulations are not well planned yet, so simulation and testing are highly recommended. Currently, most drones don't fly in bad weather like lightning strikes and rain because they're not designed to do so transporting cargo indoors with drones is lagging behind. This can be attributed to both legal restrictions and missing technical innovations. In addition, a suitable tracking system should be introduced for both basic tracking of a drone in flight and the precise tracking required to pick up and unload cargo. At this stage, the final stage of delivery, which is considered as the last stage of the transportation route and generally refers to the transportation from the warehouse or the seller to the final buyer, is carried out from a planning perspective for many reasons. Constraints such

as differences in shape and weight of the order, separate delivery locations, long distances, and the need to deliver the order on time, are as difficult and expensive as can be. While privacy, security, environment and technology issues still need to be explored, drones seem to be widely used in the final phase of product delivery with rapid technological and regulatory development. Addressing the problem of detecting obstacles just in front of the vehicle, devising a new approach for drones to avoid aerial collisions in dynamic environments, designing a new model approach for safely navigating a workspace filled with static or moving obstacles, predicting the dynamics of multiple obstacles and It is expected to be capable of avoiding them in real time.

The aim of this thesis is to increase buyer satisfaction, reduce transportation costs and offer solutions to the above-mentioned problems by using drones, removing the obstacles in front of them and providing fast and environmentally friendly delivery.

April 2021, 90 pages.

Keywords: Cargo Transportation, Unmanned aerial vehicle, Simulator, Gazebo

1. GİRİŞ

Devrimizde teknolojiye karşı olan rağbetin artması yatırımcıları ve şirketleri yeni arayışlar içine sokmuştur. Yeniliklere ayak uydurabilen şirketler diğerlerine göre bir adım daha öndedir. Lojistik sektörü de bu değişimden en çok etkilenen sektörlerden biridir. Son zamanlarda dünyadaki talep de bu alana kaymaktadır. Maliyeti düşük lakin hızda ve güvenilirlikte yeterli düzeye gelebilmek, kargo şirketlerinin birinci önceliğidir. Burada karşımıza çıkan bu özelliklere uygun teknolojik cihaz dron'dur.

İnsansız Hava Araçları; GPS ile kontrol edilen ve pilotsuz otomat namına seyir yapabilen bir hava aracı namına tarif edilmektedir. Bu günlerde dron'lar, yerdeki bir pilot aracılığı ile kontrol edilen ve uzaktan kumandalı görevler gerçekleştiren ya da evvelden hazırlanmış bir uçuş programını otomatik namına çıkaran hava araçlarıdır. Dron'lar ilk olarak askeri alanlarda ortaya çıkmalarına rağmen günümüzde reklamcılık, sağlık, tarım, gözlem ve kargo gibi birçok alanda kullanılmaktadır. Dron pazarı dünya genelinde hızlı bir şekilde büyümektedir. Her yıl yaklaşık %7 büyüme gösteren dron pazarının 2021 yılında 12 milyar dolara ulaşması bekleniyor. Dron pazarında kargo sektörü ise 3. sırada yer alıyor.

Dron taşımacılığı sayesinde motorlu araçlarla kargo taşımacılığının azalacağı düşünülmektedir. Kargo şirketiyle aynı şehirde olan bir müşterinin kargosunun ulaşımı ortalama 3 günü bulmaktadır. Dron ise bu süreyi 2 saate kadar indirip müşteri memnuniyetini getirmesi bekleniyor. Bunun diğer faydalarından biri de trafiği azaltması ve maliyeti azaltmasıdır.

İnsansız Hava Araçları; uçağın baz istasyonu, kontrol istasyonu ve uçağın kendisi olarak esas üç komponent'ten oluşmaktadır. Baz istasyonu olarak; radar, lazer, kameralar, termal görüntüleme sistemi, meteorolojik sensörleri olmaktadır. Uçağın kendisi; motor, kontrol sistemi, uçak gövdesi, uçuş planlama vs oluşmaktadır(Kahveci ve Can, 2017). Araçların motor özellikleri, gövde özellikleri ve kanat tasarımları izin verilen faydalı yüke göre belirlenir.

1.1. AMAÇ

Bu tezin amacı, Gazebo simülatör programını kullanarak dron'un engelden kaçınmasını sağlamak ve doğru kaynaklardan literatür taraması yaparak bunu desteklemektir. Tez konusu olarak Kargo taşımacılığında dron'un hedefe ulaşması incelenecektir. Öncellikle dron'un neden önemli olduğunu incelemektedir. Kargo taşımacılığında dron kullanımı, teslimat vaktini ve maliyetini azaltıp, müşteri memnuniyetini arttıracaktır. Engelden kaçınmanın avantajı uçan cisimler ile çarpışmanın ve insanların üzerine düşmesine neden olmayacağı garanti edilebilir. İç mekânlarda manuel olarak dron insanlara ihtiyaç duyar. Ancak engellerden kaçınma sensörleri ile çarpışmadan iç mekânlarda gezinmesine izin verecektir. Dron avantajı ise yapılacak olan taşıma işlemi hem hızlı hem de ürünün hasar görmemesi için son derece sağlıklıdır. Yazılım süreç modellerinin simülasyonlarını üstlenmek için çok çeşitli nedenler vardır. Çoğu durumda simülatör, karar vermeye yardımcı olmaktadır. Ayrıca risklerin azaltılmasına yardımcı olur ve stratejik, taktik ve operasyonlu seviyelerde yönetime yardımcı olmaktadır (Kellner, Madachy, Raffo, 1998).

1.2. TEZ KAPSAMI

1.2.1 Dron'un Kısa Tarihçesi

İnsansız Hava Araçlarının kullanımı tahmin edilenden daha erken kullanılmaya başlandı. 22 Ağustos 1849'da Avusturya'nın basınçlı bombalar kullanarak 200 pilotsuz balonu Venedik kentine gönderilmesi, bir İHA'nın hava saldırısında birinci kullanımı olarak kaydedilmektedir. Hatta bombaların yarısı amaçlar üzerinde patlamış, diğer yarısı ise şiddetli yelin etkisi ile Avusturya'ya geri dönmüş ve infilak etmiştir. Bu balonlar 1793 yılında Amerika Birleşik Devletlerinde askeri amaçlarla kullanılmış ve iç savaş sırasında sadece keşif amacıyla kullanılmıştır. İlk insansız hava aracı 1916 yılında yani Birinci Dünya Savaşı sırasında üretilmiş ve ilk insansız kullanım "uçan bombalar" ismiyle malum "Hewitt-Sperry" hidromekanik otomatik uçaklarla olmuştur. Bundan sonrasında Kasım 1917'de "otomatik uçan uçak" Amerika Birleşik Devletleri Silahlı Güçlerinin resmi uçağı oldu ve 1918'de ilk uçuşunu yapan bu uçak Birinci Dünya Savaştığında kullanılamadı. Önceki zamanlarda askeri amaçla kullanılan dron'lar, son kırk yılda daha fazla taşımacılık, tarım ve reklamcılık ve saire alanlarında kullanımı artmıştır.

1.2.2 Kargo Taşımacılığında Dron Kullanımının Başlangıcı

Kargo taşımacılığında dron'u ilk başta Amazon şirketi, dron aracılığıyla Cambridge'deki müşterilerine siparişlerini teslim almalarını sağlayacak bir test yapmasıyla başlamaktadır. Bu test başarı ile sonuçlanıp Amazon şirketinin dron ile tevzi maksadında bir atılım oldu. Amazon şirketi, patlamış mısır çantasının siparişini 13 dakikada gerçekleştirip, kargo teslimatında vakit artırım adına büyük bir ilerleme elde etmektedir.

Bu mevzuda İzlanda yaptığı sürekli dron taşımacılığıyla öndedir. İzlanda'daki en büyük AHA e-ticaret şirketi teslimat ağını genişletip, Reykjavik'teki müşterilerine daha patetik bir şekilde teslimat yapabilmek için Flytrex adlı dron üreten şirketi ile ortaklık yapıp, teslimat süresini azaltmaktır. Flytrex'in geliştirdiği dron, şehri ikiye bölen nehir üzerinden geçerek kargo taşımacılığı dağıtım yapma işleminde yol sürelerini azaltmak ile birlikte maliyetleri düşürmektedir. Geliştirilen bu dron sistemini kullanan AHA, günlük teslimat kapasitesini arttırırken bunu iş gücünde (çalışan sayısında) bir artırıma gitmeden gerçekleştirdi. Reykjavik'te arabayla 25 dakikada yapan teslimatları, dron ise yalnızca dört dakikada tamamlanabilmektedir. İzlanda'da ticaret yapan alıcılar, teslimat vaktini mühim bir ölçüde hızlandıran dron'lar ile adrese kadar doğrudan gıda ve siparişlerini teslim alabileceklerdir. Bu hizmet şimdilik İzlanda şehrinin en büyük ağı olan AHA'nın alıcılarına takdim edilmekte, yalnız bu müşareket nihayetinde diğer memleketlere de hizmet verilmesini planlamaktadır.

1.2.3. İnsansız Hava Araçlarının Gelişmesi

19. Asırda İnsansız Hava Araçlarının 2. Dünya Savaş'ta etkili olması ve savaşta sürücü kayıplarını minimum indirip, istihbarata yardım sağlamasıyla öne çıkmıştır. Askeri amacıyla birlikte hobi ve sivil havacılıkta kullanılmaya başlanan İHA'lar sonradan NASA tarafından uzay çalışmalarında kullanılmaya başlanılmıştır. İnsansız Hava Araçları geliştikçe, kullanım alanları çoğalıp, İHA'ların kullanımını arttırarak İHA sektöründe aktif gelişmelere neden olmuştur.

Teknolojiğin gelişimi, İHA'ların kontrolünü kolaylaştırıp, bilhassa dron'ları herkes tarafından kullanabilecek şekilde tasarlamaktadır. Bu evrim, dron'larda uzaktan kumanda yerine tablet

veya telefon kullanılması, uçuş hareketin WI-FI veya GPS ile sağlanabilmesi, dron üretim maliyetini bir hayli azaltmaktadır.

1.2.4. Türkiye'de Dron Sektörünün Ekonomik Yansıması

Son zamanlarda Türkiye'de havacılık sektörüne yatırımcılar çoğalmaktadır. Bununla birlikte piyasa serbestleşti ve sektörde mühim büyümeler yaşandı. Yerel mal icraatına tüm sektörde para yatırmak, bu noktada emek harcayan Türkiye, dron alanı için de yerel mal konusundaki çabalarına yola koyulmaktadır.

Yeryüzünde dron piyasasına baktığımızda, kullanım sektörünün çeşitlenmesi nedeniyle çoğalması ve sektörel gelişmenin Türkiye için de geçmektedir. Türkiye'de haritacılık sektörü, yapım alanı, kuvvet sektörü, bağcılık ve arazı alanında dron kullanımı yayılmıştır. 2018 yılı itibarıyla Türkiye'de, dron pazar büyüklüğü 30 milyon dolara ulaşmıştır. SHGM'ye kaydedilmiş dron sayısının 25 bine ulaştığı anlaşılmaktadır. Türkiye'deki dron'ların kullanımı hem hobi hem de ticari amaçlı kullanımları içermektedir. 2018 yılında dron'ların kurumsal ve endüstriyel uygulamalarda çok daha fazla yer alması ile 2019 senesinde kurumsal olarak dron piyasası 2-3 kata büyüme beklenilmektedir.

Gelişmekte olan sektör birçok ticari sektörde yapısal değişiklikler yaratırken, dron sürücüsü ve "dron yazılım geliştiricisi" gibi yeni iş yerlerini çoğalmasına neden olmaktadır. Bilhassa dron sürücüsü iş duyurusunda yaygın duruma gelirken SHGM'ye kayıtlı 55.000 dron pilotundan 38.000'inin olduğu biliniyor. Önümüzdeki 10 yıl boyunca bu sayının artması ve daha fazla istihdam sağlaması öngörülmektedir. Türkiye'nin drone üretiminin çoğu güvenlik ve haritalama amaçlı olarak iki gruba ayrılmaktadır. Drone üretimine ait yerelleştirilmesi devam ederken, maksadın Çin gibi bir drone hub ülke olmak olduğu tartışılmaktadır.

İlerleyen sektör ticari amaçlı birçok sektörde yapısal değişimler yaratmakla birlikte dron pilotluğu, "dron teknolojisi yazılımcısı" benzeri yeni mesleklerin ortaya çıkmasına neden olmaktadır. Bilhassa dron pilotluğu, iş ilanlarında popüler hale gelirken, SHGM'ye kayıtlı 50.000 kişiden 33.000 dron pilotluğu ehliyetine sahip birey olduğu bilinmektedir. İlerleyen 10 yıllık tahminlere göre bu sayının artarak daha yüksek istihdam sağlayacağı tahmin edilmektedir. Türkiye'deki dron üretiminin birçoğu güvenlik ve haritalama maksatlı olarak iki

gruba ayrılmaktadır. Dron üretimi konusunda yerlileşme icraatı devam ederken, hedeflerin Çin gibi dron üretim merkezi bir ülke haline gelmek olduğu da ortaya sürülmektedir. Son çalışmaların amacı bilhassa elektriksel donanımlarıyla birlikte yazılımların yerlileştirilmesine ehemmiyet göstererek devam etmektedir (SHGM, 2018).

1.3. DRON'LARIN KULLANIM ALANLARI

1.3.1. Askeri Alanda Dron Kullanımı

Günümüzde kullanılan İnsansız Hava Araçları son derece gelişmiş takip etme hususiyetine sahiptirler. Askeri alanda ordu tarafından kullanılan dron'lar, ısı sensörleri, radar, canlı yayın video kameraları ve ilk yardım paketleri gibi ekipmanlar taşıyabilme gücüne sahiptirler.

Askeri dron'ların bazıları uzak vakit havada uçabilmekte ve yeni teknoloji kameralar sayesinde belirli amaçta tüm şehri tarayabilmektedir. Dron üzerindeki kamera ile 60.000 fit uzaklıktan tablodaki yazıyı okuyabilir ve radar sayesinde konumunu belirleyecek teknolojiye sahiptir. Askeri alanda dron'un en mühim kullanım faktörü, kamyon ve dron'un birleşmesidir. Burada karşılaşacağımız problem İHA'nın hedefe gitmesinde kullanılan taşıyıcı ile ilgilidir. Uçuş esnasındaki toplam seferin mesafesini en aza indirmek için, insansız hava aracının uçuş ve kara lokasyonlarının hesap etmesi yaşamsal ehemmiyet taşımaktadır. Böyle bir hususta İHA birçok nişan noktasına donatılmış ve İHA uçuş esnasında gökyüzündeyken nakliyeci daima hareket etmektedir. Genetik algılayıcı kullanmasıyla öne çıkan problem çözümünü buldu ve diğer çözümler ile sonuçları karşılaştırılmıştır, önceki deneyim çözümler elde edilen son çözümlerden %90'a yakın daha düşüktür.

Askeri dron'larda yerleşen sensör, bilgileri toplayıp gerekli ve bol içerme alanı sağlar ve hat belirleme maliyetini de azaltmaktadır. Askeri dron'lardaki sönser, verileri toplama esnasında lazım olunan ve yeterli kapsama alını içine alır ve rota belirleme giderini de en aza düşürmektedir (Ahmet. K 2018) .

1.3.2. Sağlık Alanında Dron Taşımacılığı

İsviçre Posta ve İHA üretici Matternet Lugano şirketi, İsviçre'deki iki hastane arasında test laboratuvar numunelerini dron'lar sayesinde taşıma işlerinde başarılı oldu. Hastaneler arasında

2018’den sonra dron’lar yardımıyla laboratuvar örneklerini taşımayı düzgün bir işletme duruma getirmeye çalışmaktadır.

Ticino EOC Hastane Gurubu ve Matternet şirketi Mart ayı 2017 senesinde ortak projenin başlamasına karar verdi. Son zamanlara kadar laboratuvar örnekleri klasik kargo metotlarıyla taşınmaktaydı, yalnız bu deneme sonucunda İHA’ların karayolu taşımacılığına karşılaştırdığımızda daha hızlı ve verimli olduğu kesin derecesinde kanıtlanmış oldu. Mart ayından bugüne kadar ortalama 70 deneme uçuşu yapılmıştır. Değerlendirme sonucuna göre 2018 yazı için daha çok deneme planlanıyor ve 4 Nisan 2018'e kadar deneme uçuşları devam edecek. Bu test uçuşta kullanılan İHA’lar 80 santimetre çapında, 20 kilometre uçuş menziline sahip ve 2 kilograma kadar yükü taşıyabilmektedir. Tıbbi malzemelerin kırılmadan karşı tarafa gönderile bilmesi önem arz ettiği için, İHA’nın arızalanma durumunda taşınan tıbbi malzemelerin açık alana güvenli bir şekilde bırakılıp, kendini hasarsız bir şekilde iniş yaparak kurtulabilmesi mümkündür. Tanzanya hükümeti kan ve aşı gibi tıbbi malzemeleri uzak bölgelere dağıtmak için 2018 yılı başında dron kullanmayı planlıyor. Ruanda hükümeti bugüne kadar 1400 başarılı bir şekilde dağıtım yapmak durumundadır (Ahmet. K 2018).

1.3.3. Dron İle Gıda Taşımacılığı

Hazır gıda alanındaki yarışın bir getirisi olarak İHA ile teslimat daha alımlı bir duruma gelmiştir. Zaman açısından kısa bir sürede ve siparişi sıcak bir şekilde teslim edebilmesi için fast-food şirketleri dron ile denemelere başlayıp sonuçta en hızlı teslimatın dron ile yapıldığına kanaat getirmiş olmaktadır.

2016 tarihinde ilk olarak Yani Zalanda, Auckland şehrinde Dominoz pizza ve Google şirketi’nin bir başka kurumu olarak bulunan Alphabet, Blacksburg’taki Virginia Tech isminde fast-food şirketi yemek siparişlerini dron ile teslim yapmaktan başarılı olmuştur.
Denemeler sonucu ne kadar olumlu sonuçlar versede ABD’li alıcıların birçoğu bununla ilgili şüphelidirler lakin ABD Posta Servisi’nin yaptığı anket sonuçlarına göre %75’lik oranı 5 yıl içerisinde dron’u teslimatta kullanacağını düşünmektedir. ABD halkının yaptığı anket sonuçlarına göre %69’luk kesim dron ile gıda teslimatının güvenli olacağına inanmaktadır ve %31'lik kesim ise bu konu hakkında görüş beyan etmemişlerdir (Ahmet. K 2018).

1.4. KARGO TAŞIMACILIĞINDA DRON KULLANAN ÜLKELER

Afrika günümüzde dron'u en çok kullanan ülkelerden biridir. Afrikada'ki dron kullanımının en büyük bölümü destek amacı taşımaktadır; kargo taşımacılığında dron kullanan ve dron geliştirme merkezleri bulunan ülkeler Amerika, İngiltere ve Almanya. Özellikle Doğu Afrika dron teslimatının en çok kullanıldığı bölgedir. Bu bölgede Ruanda ve Tanzanya ilk kullanan şehir olarak sayılır. Bazı insanlar; ABD'de zengin olduğundan dolayı; dron, yapay zekâ, robotik kullanıyor diyen yanlış düşünceler var. Bu ülke teknolojiden zengin değil, ABD yeni ve modern denemeler yapmaya istekli ülkedir (Keller. R, 2019).

İngiltere'deki ilk testinde, Amazon şirketi Cambridge merkezli isteme aldı. Dron gemiden paket ile kalkıp, GPS ile belirlenen konuma 2,8 kilo ağırlığında paketi teslim etmiştir. Amazon şirketi 30 dakika'da dron ile teslimat yaptığını açıklamıştır (Ahmet. K 2018).

Türkiye'de ise dron ile ilk teslimat testi PTT tarafından 2018 senesinde gerçekleştireceğini açıklamıştır (Hürriyet Haber, 2017).

Şekil 1.4.1: Ht 448 Modüllü Kargo Taşıma Dronu

Bu HT488 modelli Turkish Airlines tarafından tasarlanmış 80 kg ağırlığında yük kaldıra bilen dron'dur. Maksimum hızı 115km/saat. 2025 senesinden itibaren faydalanmaktadır.

1.5. KARGO TAŞIMACILIĞINDA DRON KULLANIMININ GÜVENLİĞİ

Kargo taşımacılığında dron kullanımının güvenliği, büyük problemlerinden biridir. Son zamanlarda Amazon şirketi dron ile kargo taşımacılığının güvenliği için çok çalışma yapmaktadır. 2017 senesinde Amazon şirketi Londra Stansted ve Conwall Havalimanlarında yaptığı deneme uçuşlarında toplanan raporlarını incelemektedir.

Bizim en büyük önceliğimiz güvenliktir. Günümüzde uçuşlarımızı şiddetli rüzgârlar dahi gerçekleştirebilmektedir, ama kar ve yağmur gibi havalarda şimdilik gerçekleştirememektedir (Jeff. B, 2019).

1.5.1. Dron'un Güvenliği

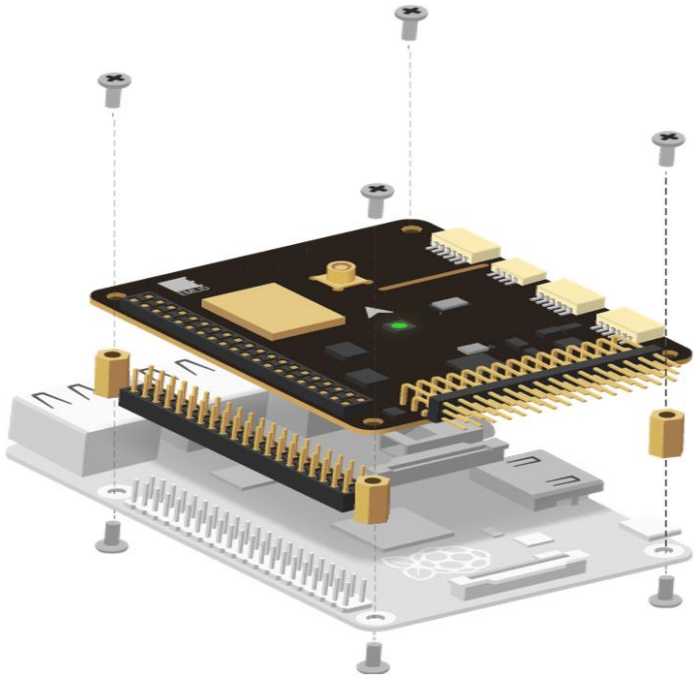
İnsansız hava araçları uçarken kendi güvenliği ile birlikte başka taşıma araçların güvenliğini de riske attığı hakkındaki görüş yanlıştır. İHA'lar uçaklar için güvenlik acısından bir tehlike yoktur, zira ticari amaçla yapılan İHA'lar 400 ft'nin üzerinde herhangi bir uçuş yapamazlar ve yolcu uçakları ile aynı hava alanında uçmamaktadır. Ancak havalimanları ve başka alçak uçuş alanlarında, yolcu uçakların inip kalktığı bölgelerde güvenlik nedeni ile İHA uçuşları gerçekleşememektedir. İHA'lar sensör ve kamera yardimi ile etrafındaki nesneleri tespit etmektedir (Ahmet. K 2018).

2. GENEL KISIMLAR

2.1. DRON DONANIMLARI

2.1.1. Dron Uçuş Kontrol Kartı (FC)

Uçuş Kontrol kartları dron'un beynidir. Uçuş kontrolörü, GPS modülü, pusula, engellerden kaçınma sensörleri ve uzaktan kumandadan gelen verileri alır ve motorları kontrol etmek için ESC'lere verilen bilgilere dönüştürür. Kullanıcı tarafından belirlenir. Bunun bir örneği, rüzgârlı koşullarda bir dron havadayken görülür. Geçmişte veya ucuz bir dron'unuz varsa, dronun konumu ve bu değişikliklerin nasıl düzeltileceği hakkında bilgi aktaran sensörler olmadığı için sadece etrafta sürüklenecektir. Ancak bu dron'da, dron GPS ve aşağı görüş sensörlerinden tam konumunu biliyor, bu nedenle rüzgâr esse bile tam yerinde kalacak, çünkü uçuş kontrolörü ESC'lere uygun talimatları rüzgâr faktörünü telafi etmek için gönderiyor ve motorları çalıştırmaktadır.



Şekil 2.1.1: Uçuş Kontrol Kartı

2.1.2. Dron Çerçevesi

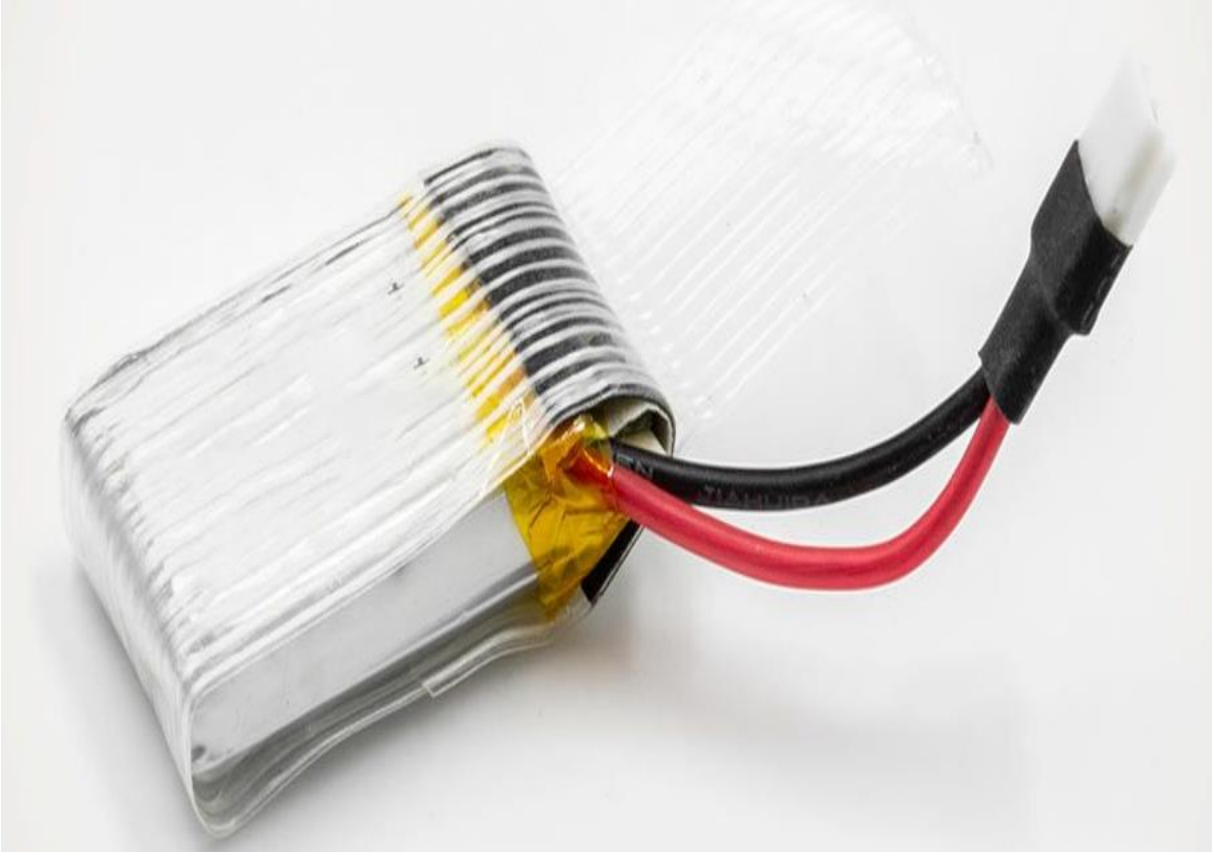
Dron çerçevesi İHA'ların tüm donanımlarının yerleştiği bir iskelettir. Üstünde sensörleri, anten, batarya, kontrolörler benzeri donanımları yerleştirmek için yerler, döngeç ve motorlar için ayrı yerleri mevcuttur. Çerçeve zayıfsa, motorun kuvveti dron'u kolayca kaldırabilir ve yüksek irtifada ve daha hızlı uçabilmektedir. Çerçeveler çoğunlukla alüminyum, mühendislik polimerler ve karbon fiberden yapılmaktadır.

Şekil 2.1.2: Dron Çerçevesi

2.1.3. Dron Bataryası

Batarya, multikopterin güç deposudur. Küçük ebatlı multikopterlerde umumiyetle LiPo piller veya NiMH bataryalar kullanılmaktadır. Batarya seçiminde voltaj, kapasite ve deşarj oranı önemlidir. Bu piller, aşırı şarj korumasına, sıcaklık verilerine, şarj döngüsü geçmişine sahip oldukları ve dron güç çıkışını ilettikleri anlamına gelen "akıllı" pillerdir. Bu, pilin tekrar tekrar

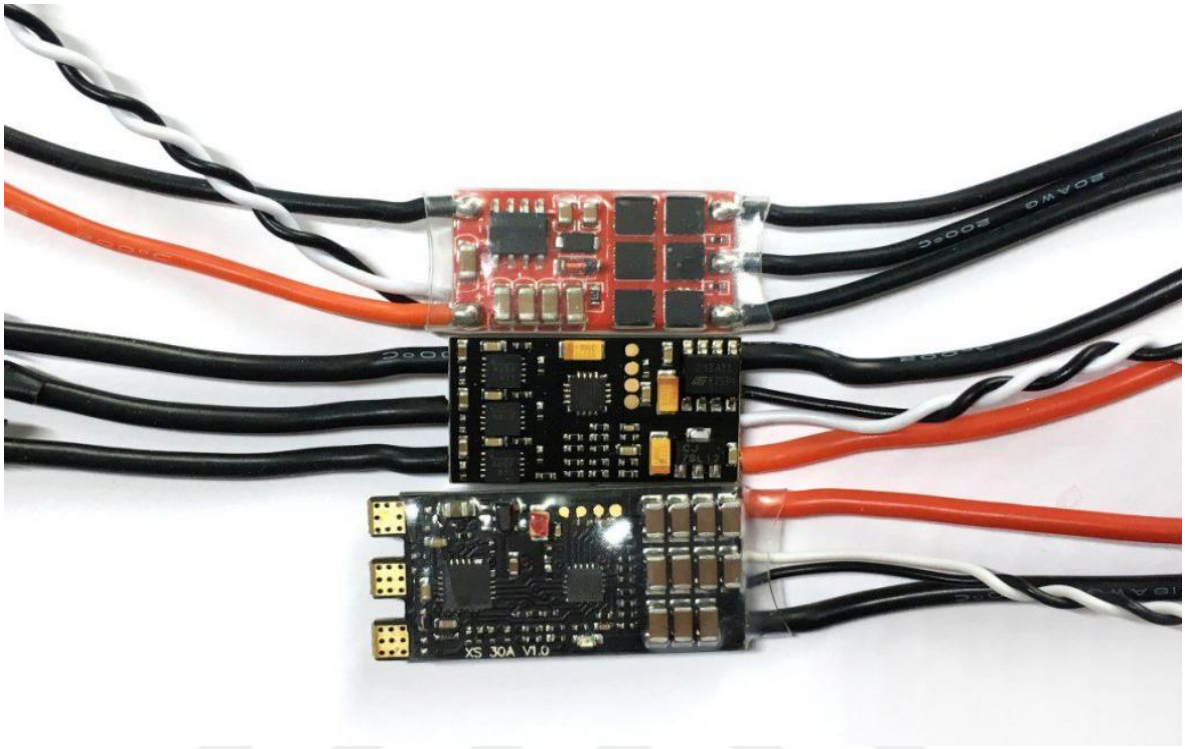
kullanılmasının güvenli olmasını ve uçuş sırasında herhangi bir sorun olmamasını sağlamak içindir.



Şekil 2.1.3: Dron Bataryası

2.1.4. Elektronik Hız Kontrol Cihazı (ESC)

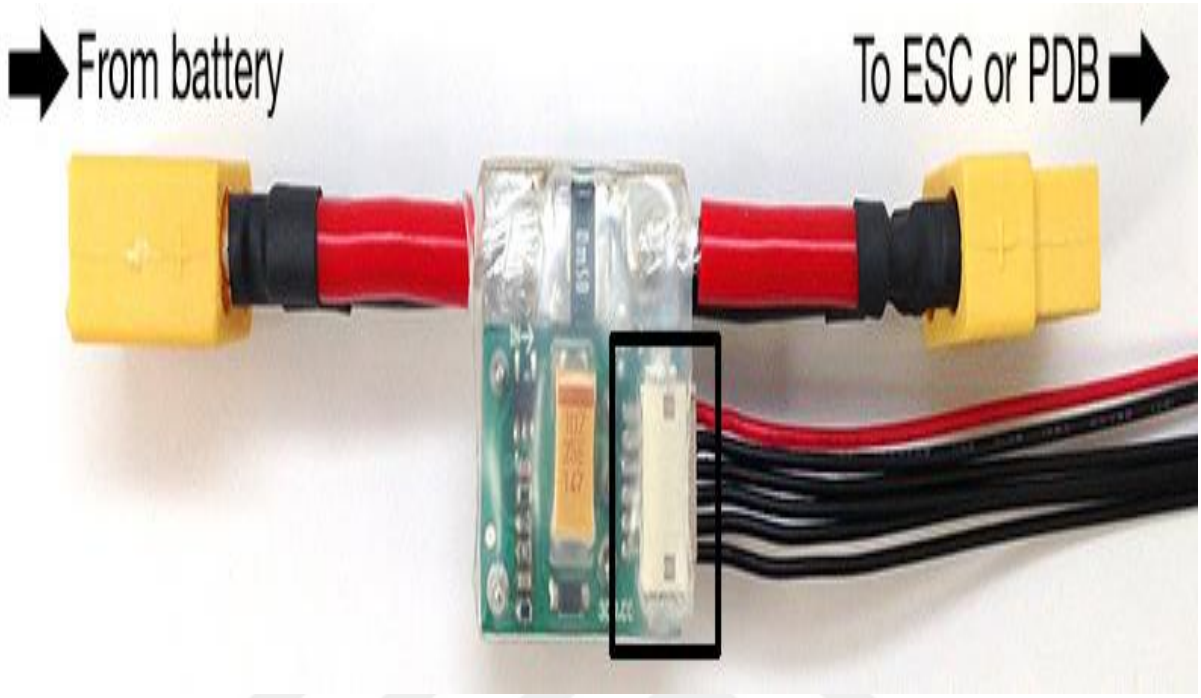
Elektronik hız kontrolcüsü, motora giden gaz kolu işaretine uygun olarak motorun dönme şiddetini uyduran elektronik sürücü devresidir. ESC pil ve motor tipine uygun olarak seçilmesi gerekmektedir. ESC'nin bir tarafı motora, öbür tarafı ise uçuş kontrolcüsüne bağlanmaktadır. Dron'larda her bir motora göre bir ESC bulunmaktadır. Dikkate alınması gereken ilk madde, itki yönteminin çekeceği daima ve geçici akım değerleri ve kullanılacak pilin gerilim değerleridir. Pil - döngüç - motor seti değerlendirilerek yeterli akım ve gerilim değerlerini karşılayabilecek ESC modelinin seçilmesi gerekir. ESC, pile ve uçuş kontrolörüne bağlıdır, ESC'ler uçuş kontrolöründen sinyal aldığından, motorların her birine verilen güç miktarını değiştirmektedir.



Şekil 2.1.4: Elektronik Hız Kontrol Cihazı (ESC)

2.1.5. Dron Güç Dönüştürücü Modülü (PDB)

Otomatik pilot hassas elektronik cihazdır ve temiz bir güç alması mühimdir. Kuvvet mödülü, İHA bataryanızdan akümülatör voltajını, otomat pilotun kullandığı az bir voltaja indirmek amacıyla kullanmaktadır. Güç modülü, pilden gelen güç miktarını izler ve bunu dron'nun ESC'lerine ve dövüş kontrolörüne dağıtır.



Şekil 2.1.5: Güç Modülü

2.1.6. Dron motoru

Motorlar pervaneleri döndürüyor, her bir pervane' ye ait motor gerekmektedir. Çoğunlukla kuyruksuz motorlar kullanmaktadır. Motor İHA ağırlığına dair seçilir ve genelde motorların tüm itme kuvveti dron ağırlığınından fazla hesaplanmaktadır. Demek ki yarım kuvvet harcadığında dron gökyüzünde asılı kalabilmektedir. Dron'lar, dönen pervaneler tarafından üretilen dönüş kuvvetini eşitlemek için iki saat yönünde motora ve iki saat yönünün tersine motora sahiptir. Bunun nedeni, Newton'un her eylem için eşit ve zıt bir tepki olduğunu belirten Üçüncü Yasasıdır. Bu nedenle birbirine zıt hareket eden eşit sayıda motora sahip olmak, dönüş kuvvetini eşitleyerek stabilize sağlar. Bu nedenle helikopterlerde, tek ana rotordan gelen dönme kuvvetine karşı koymak için bir kuyruk rotoru bulunur.

Şekil 2.1.6: Dron Motoru

2.1.7. Dron Pervaneleri

Dron'lar saat yönünün tersine iki motora ve saat yönünde motora sahip olduğundan, ayrıca her motor yönü için bir tane olmak üzere iki farklı pervaneye sahiptir. Her pervane, havayı kanat profili yüzeyinde aşağı doğru iterek pervanenin üstünde daha düşük bir basınç alanı ve altında daha yüksek bir basınç alanı oluşturarak bir basınç farkı oluşturarak dron'u yukarı doğru iter.

Şekil 2.1.7: Dron Pervaneleri

2.1.8. GPS Modülü

Küresel konumlandırma uydu modülü, dron'nun yerini tam olarak belirlemek için iki farklı küresel konumlandırma sistemi kullanır. Dünya yörüngesinde dönen 24 uydudan oluşan GLONASS olarak bilinen Rus ağını kullanır. Bu, 31 uydudan oluşan Amerika Birleşik Devletleri ağı ile birlikte kullanılır. Bu uydular, konumuyla ilgili bilgileri Dünya yüzeyine iletir. Bu sinyaller ışık hızında hareket eder ve dron üzerindeki GPS modülü tarafından okunur. Buradan, dron, sinyallerin çeşitli uydulardan gelmesi için geçen süreye göre coğrafi konumunu hesaplar. Bu küresel konumlandırma uyduları, dron'a Dünya'da nerede olduğunu anlama ve konumunu koruma yeteneği verir.

Şekil 2.1.8: GPS Modülü

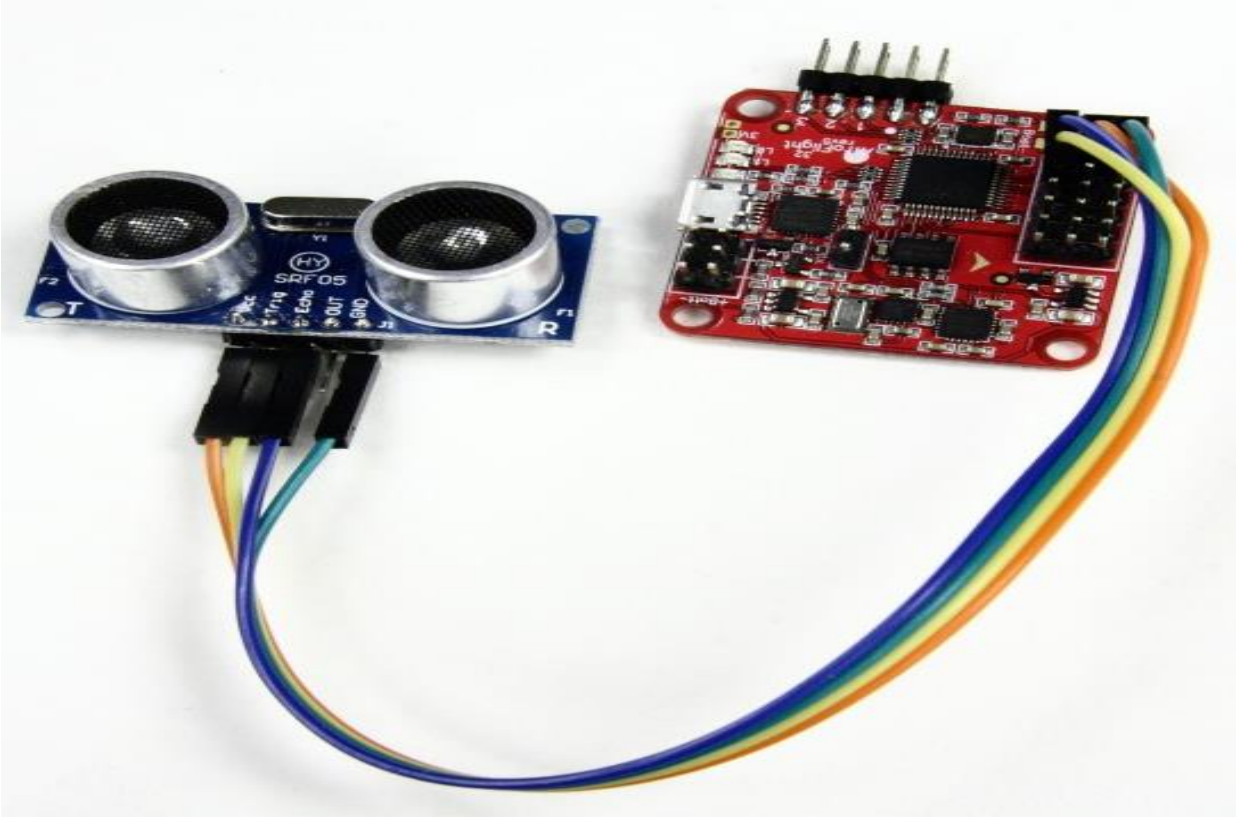
2.1.9. Dron Antenler

Dron'nun bacaklarının içinde, dron'dan kontrolöre ve kontrolörden dron'a bilgi aktaran iletim sistemi bulunur. Ayrıca, bu dronun bacaklarında yönünü uçuş kontrol cihazına ileten iki pusula sensörü vardır.

Şekil 2.1.9: Dron Antenleri

2.1.10. Engelden Kaçınma Sensörleri

Sensör dron'nun önünde ve altında stereo görme sensörleri var, bu sensörler tıpkı gözleriniz gibi çiftler halinde çalışıyor. Bu sensörler, her bir sensörden gelen görüntü piksellerinin aynı noktaya karşılık geldiğini belirleyerek derinliği hesaplar. Buradan dron, sensörler arasındaki mesafe sabit olduğu için önündeki nesneye olan mesafeyi hesaplayabiliyor. Başka bir deyişle, dron, bir nesnenin dron'dan uzaklığını hesaplamak için Pisagor Teoremini tekrar tekrar çözer.



Şekil 2.1.10: Endelden Kaçınma Sensörleri

2.1.11. Aşağıya Doğru Ultrasonik Engelden Kaçınma Sensörü

Bir sensör yüksek frekanslı bir ses darbesi gönderir ve diğer sensör darbeyi alır. Nabzı gönderme ile nabzı alma arasındaki süreye bağlı olarak dron, dron'nun yerden yüksekliğini hesaplar.

Şekil 2.1.11: Aşağıya Doğru Ultrasonik Engelden Kaçınma Sensörü

2.1.12. 3 Eksenli Gimbal

Dron'lar uçarken titreşim oluşturuyor ve bu titremeden koruyan bir teknik gerekmektedir. Bundan dolayı gimballar'a direk kameraları kullanmak gerekmektedir. Bunun sayesinde profesyonel çekimler yapılmaktadır. Dron görüntüleri bu şekilde sabit tutulur. Kameranın etrafındaki 3 farklı eksene bir motor yerleştirilmiştir. Sensörler bu eksenlerden herhangi birinde hareket algıladığında, motorlar hareketi iptal etmek için harekete geçer. Bu, pürüzsüz görüntü sağlamak için binlerce hesaplama yapıldığından neredeyse anında gerçekleşir.

Şekil 2.1.12: 3 Eksenli Gimbal

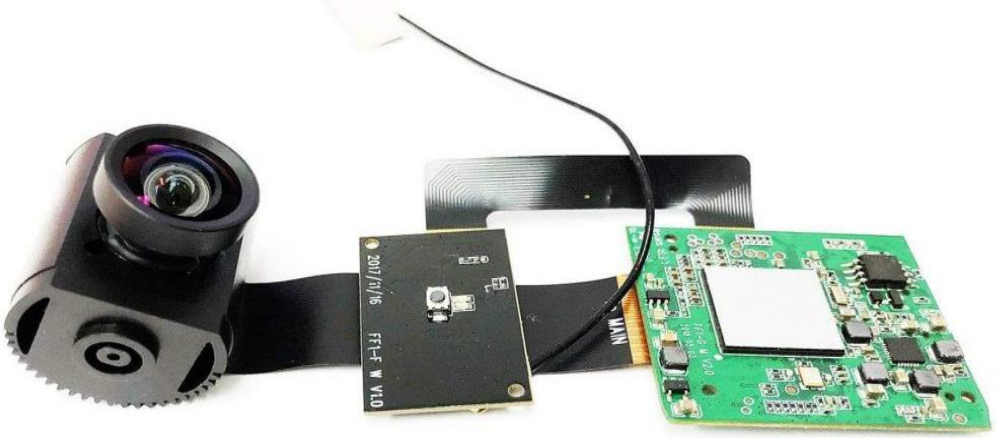
2.1.13. Dron kamerası

Dron kamerası çekim yapmak veya pilota uçtuğu yeri göstermek için kullanmaktadır. Kameralar güçü İHA pilinden alıp ve onun üstüne yerleşen bellek kartlarına çekim yapmaktadır. Kameranın önünde bir mercek açılır ve ışık içeri girer. Bir görüntüleme sensörü, gelen ışık ışınlarını yakalar ve ardından onu dijital bir görüntüye dönüştürür.

Şekil 2.1.13: Dron kamerası

2.1.14. Ana Kamera Kartı

Kamera kartı sabit görüntü sağlamak için görüntüleme sensöründen ve gimbal motorlarından gelen bilgileri işler. Bu kart aynı zamanda kamera bilgilerini işleyerek görüntüyü micro SD karta yazar.



Şekil 2.1.14: Ana Kamera Kartı

2.1.15. Uçuş LED'i

LED kullanıcıya dron'un hangi yöne baktığını göstermek için çeşitli renkleri yanıp söner. Yanıp sönen iki kırmızı ışık, dron'un önünü gösterir (kameranın baktığı yön). İki yeşil yanıp sönen ışık, dron'un arkasıdır.

Şekil 2.1.15: Uçuş LED'i

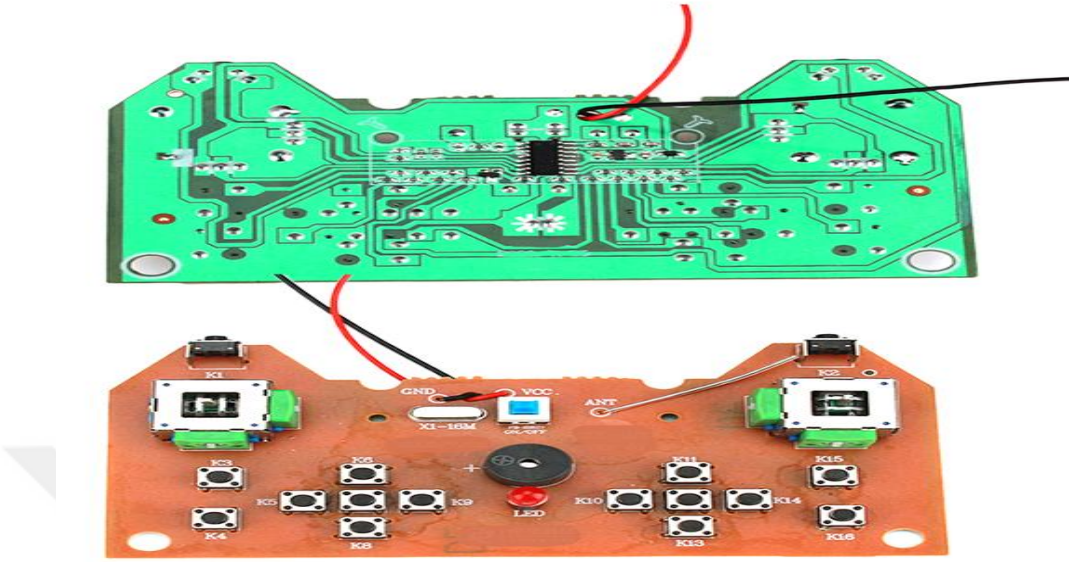
2.1.16. Uzaktan Kumanda

Çoğunlukla dron'lar uzaktan kontrol etmek için tasarlanmış cihazlardir. Günümüzdeki dron'lar ise uzaktan kumanda adına sadece akıllı cihaz veya tabletleri kullanmaktadır. İHA'ların üstündeki kontrollerin tümünü çalıştırabilen doğru kontrol cihazı ile uçuşa başlamak, doğru ve kontrollü uçuş için esastır. Bunlar, çubukların fiziksel hareketini, kontrolörün dron ile iletişim kurmak için kullanabileceği bilgilere dönüştürür. Sol oyun kolu, dron'u yukarı ve aşağı hareket ettirir ve sağa ve sola kaydırma yapar. Sağ oyun kolu, dron'yu ileri ve geri hareket ettirir ve sağa ve sola sürüklenir.

Şekil 2.1.16: Uzaktan Kumanda

2.1.17. Ana Uzaktan Kumanda Kartı

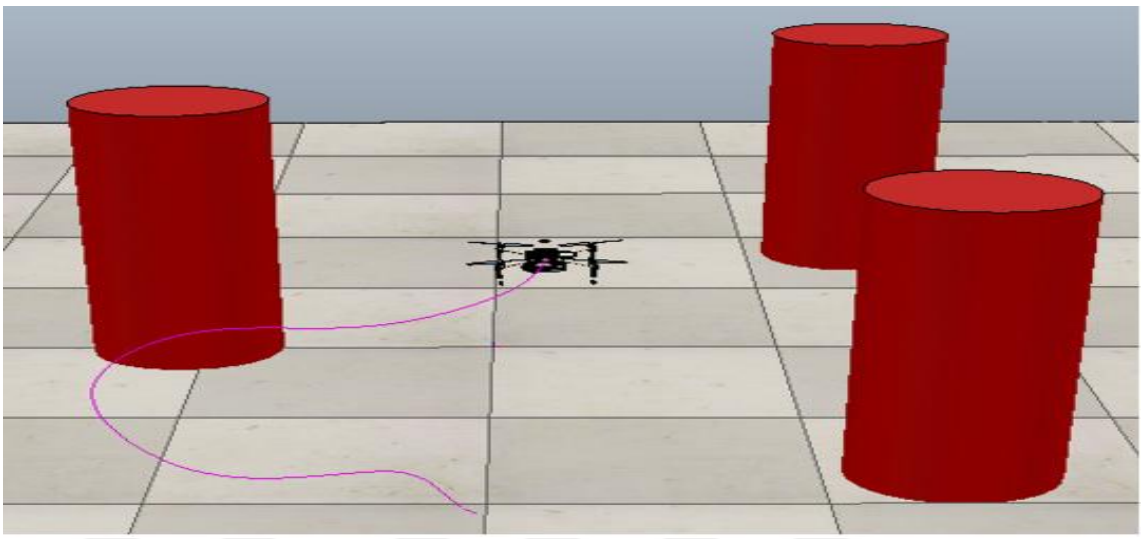
Kumanda kartı dron'dan konumu, yüksekliği ve kameranın ne gördüğü hakkında bilgi alır. Ayrıca yönetme kolundan girdi alır ve komutları uçuş kontrolörüne gönderir.



Şekil 2.1.17: Ana Uzaktan Kumanda Kartı

2.2. DİNAMİK ORTAMLARDA HAVADAN ÇARPIŞMADAN KAÇINMA İÇİN MODEL ÖNGÖRÜLÜ KONTROL.

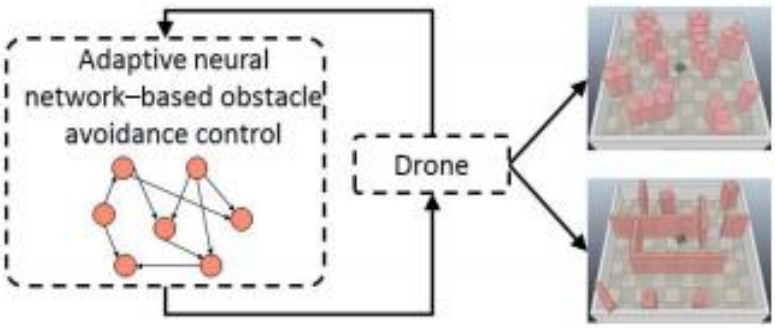
İnsanlar ve diğer robotlar tarafından doldurulan bilinmeyen ortamlarda otonom navigasyon, mobil robotlarla çalışırken ana zorluklardan biridir. Bu yazıda, çok rotorlu insansız hava araçları için dinamik çarpışmadan kaçınmaya yönelik yeni bir yaklaşım sunuyoruz. Statik ve/veya hareketli engellerle dolu bir çalışma alanında güvenli bir şekilde gezinmek için yeni bir doğrusal olmayan model öngörücü kontrol (NMPC) yaklaşımı önerilmiştir. Yaklaşımımızın benzersizliği, gerçek zamanlı olarak onlardan kaçınarak birden fazla engelin dinamiklerini öngörme yeteneğinde yatmaktadır. Aktif küme algoritmalarından yararlanarak, her örnekleme zamanında yalnızca tahmin ufku sırasında İHA'yı etkileyen engeller dikkate alınır. Ayrıca, yeniden formüle ederek kaçınma manevralarının akıcılığını da geliştirmektedir. Yönlendirilebilir elipsoidler olarak engeller, yerel minimumlara daha az eğilimlidir ve tercih edilen bir kaçınma yönünün tanımlanmasına izin vermektedir. Son olarak, simülasyona dayalı iki gerçek zamanlı uygulama sunuyoruz. İlki, yaklaşımımızın güvenlik ve verimlilik açısından analog statik formülasyonundan daha iyi performans gösterdiğini göstermektedir. İkincisi, çoklu dinamik engellerden kaçınma yeteneğini göstermektedir (M. Castillo-Lopez; S. Jose; M. Miguel,2018) .



Şekil 2.2.1: İHA, insan boyutundaki dinamik engellerden kaçınmaktadır.

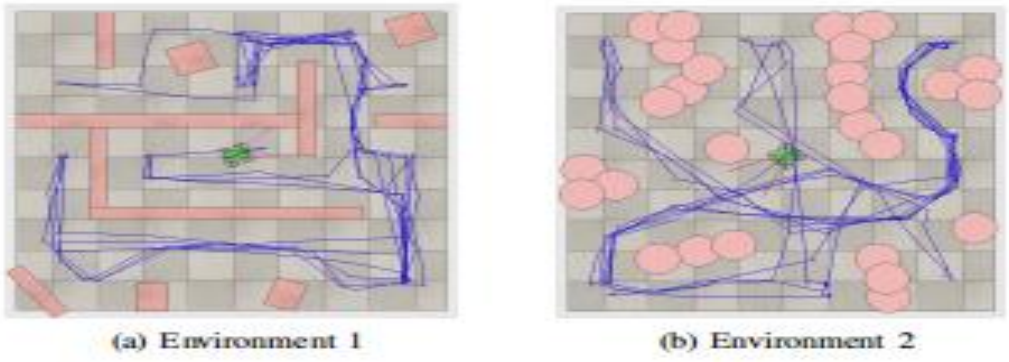
2.3. GERÇEK DÜNYA ORTAMLARINDA UYARLANABİLİR ENGELLERDEN KAÇINMA İÇİN OTONOM DRONE'LARIN GELİŞTİRİLMESİ

Son zamanlarda insansız hava araçları, altyapı denetimi, kriz müdahalesi ve arama kurtarma operasyonları gibi birçok kritik görevde yer aldı. Bu tür insansız hava araçları, engellerden etkili bir şekilde kaçınmak için çoğunlukla karmaşık bilgisayar görme teknikleri kullanır ve bu nedenle yüksek hesaplama gücü gerektirir. Bu nedenle, bu çalışma, bir dronun uyarlanabilir engellerden kaçınma kontrolü için daha önce yazarlar tarafından geliştirilen, hesaplama açısından ucuz bir algoritmayı ayarladı ve test etti. Algoritma, dronun kilitlenme ve köşe gibi karmaşık durumlara girmesini önlemeyi amaçlıyor. Algoritma simülasyon yoluyla doğrulandı ve altyapı denetimi için yeni geliştirilen bir drone platformuna uygulandı. Drone platformunun tasarımı ve deneysel sonuçları bu çalışmada sunulmuştur.



Şekil 2.3.1: Uyarlanabilir engellerden kaçınma kontrolü, simüle edilmiş ve gerçek dronlara doğrudan uygulanabilen bir kapalı döngü kontrol sistemi görevi görür.

Bu yazıda, otonom engellerden kaçınma ve keşif için sinaptik plastisite ile uyarlanabilir bir sinir kontrolü açıklanmıştır. V-REP simülatörü, bir drone ve farklı ortamları simüle etmek ve ayrıca geliştirilen kontrolörün performansını değerlendirmek ve engellerden kaçınma davranışını göstermek için kullanıldı. Bir drone platformunun komple tasarımı sunulmaktadır. Dron, uyarlanabilir obstetrik kaçınma algoritmasının gereksinimlerine uyacak şekilde geliştirilmiştir. Bu nedenle, iki hafif LiDAR sensörü seçildi ve tespit edilebilir engellerin minimum genişliği gibi önceden tanımlanmış tasarım gereksinimlerini karşılamak için LiDAR'ların tam konumlarını bulmak için matematiksel denklemler formüle edilmektedir (E. Devos, E. Ebeid, P. Manoonpong 2018).



Şekil 2.3.2: Uyarlanabilir otonom engellerden kaçınmanın sonuçları

2.4. PANTHER: ALGIYA DUYARLI YÖRÜNGE PLANLAYICI DINAMIK ORTAMLAR.

Bu makalede, dinamik ortamlarda gerçek zamanlı bir algıya duyarlı (PA) yörünge planlayıcısı olan PANTHER'i sunmaktadır. PANTHER, dinamik engellerden kaçınan ve aynı zamanda onları sensör görüş alanında (FOV) tutan ve nesne izlemeye yardımcı olmak için bulanıklığı en aza indiren yörüngeler planlar. İnsansız Hava Aracı'nın dönüşü ve çevirisi ortaklaşa optimize edilmiş ve bu da PANTHER'in çoklu rotorların diferansiyel düzlüğünden tam olarak yararlanmasına olanak tanımaktadır. Gerçek zamanlı performans, bu kısıtlamayı Hopf fibrasyonu aracılığıyla dolaylı olarak dayatarak elde edilir. PANTHER, FOV içindeki engelleri

sırasıyla intikal ve sapmayı ayıran PA olmayan yaklaşmalara ve PA yaklaşmalarına gereğince 7,4 ve 1,5 kat daha fazla tutabilmektedir. Öngörülen hız (ve dolaysıyla bulanıklık) ~%30 oranında azaltılmaktadır. Yakında türetilen MINVO tabanımız, konum ve hız uzayında düşük korunumla çarpışmadan kaçınma kısıtlamaları uygulamak için kullanılır. Sonuç olarak, tüm hesaplamaların yerleşik olarak çalıştığı, bilinmeyen dinamik ortamlarda kapsamlı donanım deneyleri, 5,8 m/s'ye kadar hızlarda ve 6,3 m/s'ye kadar bağıl hızlarda (engellere göre) sunulmaktadır (Jesus. T, Jonathan. P, 2021).

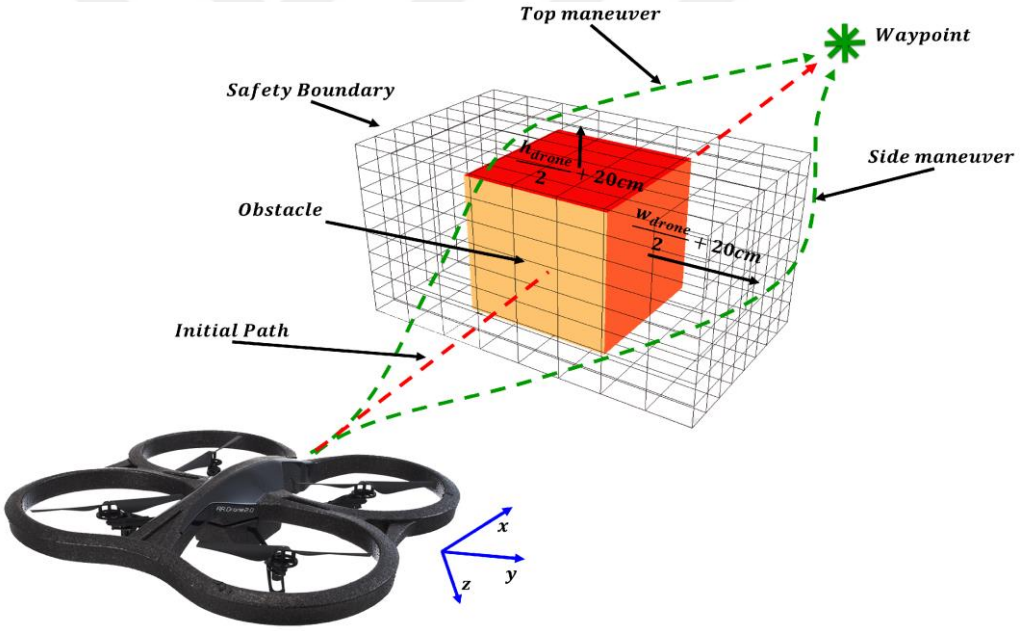
2.5. SİVİL UYGULAMALAR İÇİN İNSANSIZ HAVA ARACI TASARIMI VE GELİŞTİRMESİ

İnsansız Hava Araçları, insan operatörü taşımayan, araç kaldırmasını sağlamak için aerodinamik kuvvetleri kullanan, otonom olarak uçabilen veya uzaktan kumanda ile kontrol edebilmektedir. Yerleşik bilgisayarlar tarafından otonom olarak veya yerdeki bir pilotun uzaktan kumandası ile kontrol edilir. Kullanımı şu anda uydu iletişimi ve maliyet gibi zorluklarla sınırlıdır. Radyo frekansı kontrolörü tarafından çalıştırılabilen ve canlı görsel-işitsel geri bildirim gönderebilen bir dron inşa edilmektedir. Geliştirilen Drone kontrolü sistem MATLAB/Simulink ortamında simüle edilmiştir. Simülasyon, geliştirilen dron'un çok kararlı bir şekilde çalıştığını ve kontrolünü göstermektedir. Mikro denetleyici tabanlı drone kontrol sistemi de dron için uzaktan çalıştırma için 2,4 GHz frekansında çalışan bir RF vericisi ve alıcısının kullanıldığı geliştirilmiştir. Daha önce, dron'lar askeri uygulamalar için kullanmıştır. Bu çalışmada geliştirilen dron, polislik, yangınla mücadele, selden etkilenen alanların izlenmesi, geçilmez alanlardan ve hem askeri hem de askeri olmayan güvenlik çalışmalarından video görüntüleri kaydetmektir. Ayrıca, canlı konum için GPS ile birleştirilmiş bir Android mobil cihaz kullanılarak kullanılmıştır. Dron izleme ve gerçek zamanlı görsel-işitsel geri bildirim yapmaktadır (R.Azim Bappy, S.Islam, A.Sajjad, N.Imran 2017).

2.6. MONOKÜLER KAMERA KULLANAN QUADROTOR ENGEL ALGILAMA VE KAÇINMA SISTEMI

Engelleri tespit etme ve kaçınma yeteneği, otonom hava araçları sistemlerinde temel bir özelliktir. Çoklu sensörler ve kameraların bir kombinasyonunu kullanarak engelleri tespit etmek için en geleneksel yöntemler, ancak bir quadrotor, taşıyabilecek yük açısından büyük bir

sınırlamaya sahiptir ve bu nedenle gemideki sensörlerin sayısı son derece sınırlıdır. Bu problemle başa çıkmak için, bu makale, ucuz bir dört rotorlu yerleşik tek bir kamera kullanarak göze çarpan nesne algılama ve restleştirme görüntüsüne dayalı yeni bir engel algılama yöntemi sunmaktadır. Quadrotor kameranın videosu, kablosuz bağlantı yoluyla yer istasyonuna gönderilir ve burada bir histogram geri projeksiyon algoritması kullanılarak göze çarpan bir harita oluşturulur ve bu harita Nx × Ny karelere bölünür, son olarak her karenin yoğunluğunu bir eşik ile hesaplar ve karşılaştırır. Olası bir engeli tespit edebilir ve bulabiliriz. Önerilen yöntemi bir papağan quadrotor ile başarıyla gösteriyor ve değerlendirmektedir. İyi deneylerdeki performans önerilen yöntemi destekler, herhangi bir ön işleme verisi olmadan engelleri tespit edebilmektedir (D.Valencia, D.Kim, 2018).

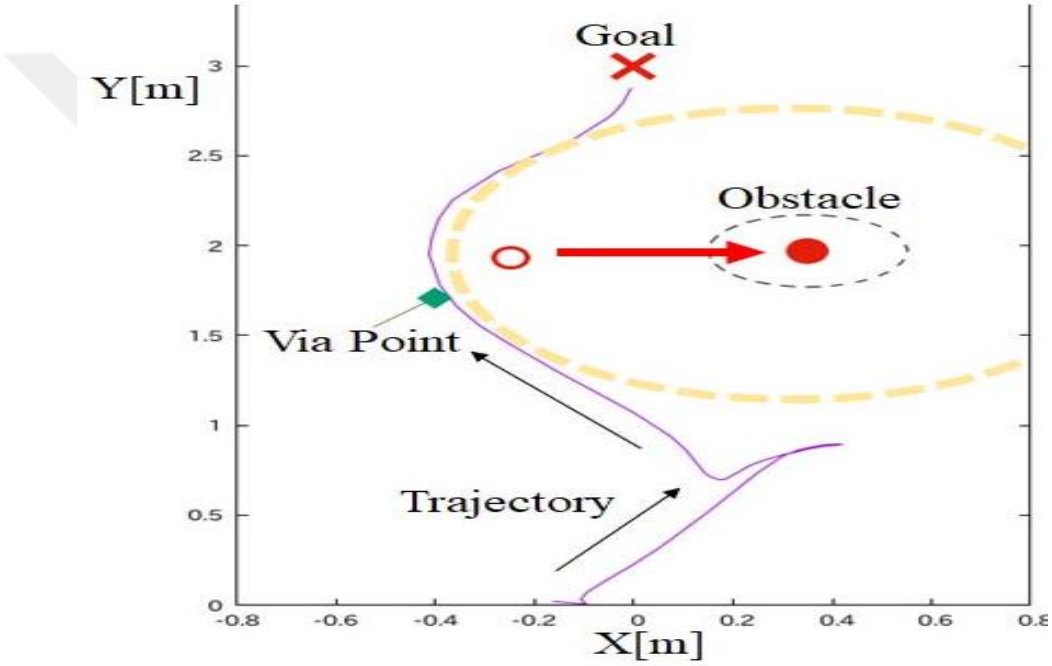


Şekil 2.6.1: Engelden Kaçınma Yolu

2.7. KAPALI ORTAMDA UÇAN DRONE YATAY HAREKET KULLANARAK ENGELDEN KAÇINMA NAVİGASYONU.

Eskiren altyapıları denetlemek için bir dron, GPS sinyallerinin alınamadığı binaların yakınında hareket edebilme yeteneğine sahip olmalıdır. GPS'e dayanmayan bir navigasyon teknolojisi elde etmek için bu çalışmada bir dron'nun kapalı ortamlardaki konumunu kontrol edebilen bir sistem geliştirilmiştir. GPS bilgisi olmayan kapalı ortamlarda bile dron'nun konum bilgilerini

elde etmek için bir izleme kamerası olan Intel Real Sense İzleme Kamerası T265'i kullanan bir konum tahmin algoritması kullanıldı. Ardından, komut verilen hedef konumun konumunu kontrol etmek için bir sistem oluşturulmuş ve deneylerle doğrulanmıştır. Ayrıca yatay hareket kullanarak engellerden kaçınmak için bir kontrol algoritması geliştirilmiş ve geliştirilen sistem ile test edilmektedir (S. Kawabata, J. Hoon Lee, S. Okamoto, 2019).

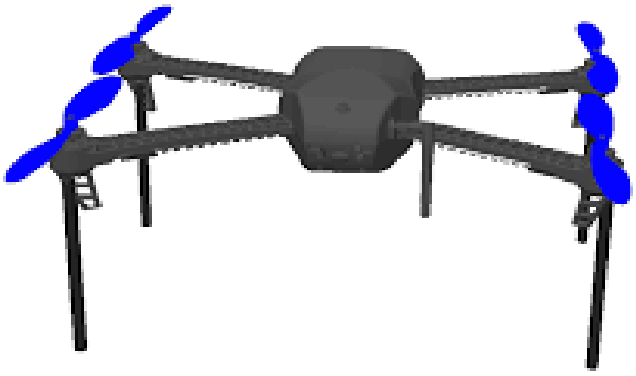


Şekil 2.7.1: Engel (-0,3, 1,9)'dan (0,3, 1,9)'a hareket ettiğinde hareketli bir engelden kaçınma deneyinin sonucu

3. MALZEME VE YÖNTEM

3.1. GAZEBO SIMÜLASYONU

Gazebo simülatörü bir veya birden fazla robotların 3 boyutlu ortamda simüle edebildiği bir programdır. Kapalı ve açık mekânlar için geliştirilmiş, çeşitli sensörler ve arayüzler sunmaktadır. Robotik algoritmalarının test edilmesi ve robot tasarımı Gazebo'nun genel kullanım amaçlarındandır.



Şekil 3.1.1: Gazebo Iris Çoklu Döner Kanatlı İHA Modeli

GitHub, sürüm kontrol sistemi olarak Git kullanan yazılım geliştirme projeleri için web tabanlı bir depolama servisidir. Projede kullanılacak bazı kütüphaneleri GitHub servisinden yüklememiz için aşağıdaki komut satırlarını Linux İşletim sistemindeki terminalde çalıştırarak Git uygulaması kurulmuştur.

```
cd ~
sudo apt install git
```

ArduPilot, otonom kontrol edebilen, açık kaynaklı, insansız bir araç Autopilot yazılım paketidir. Aşağıdaki komut satırlarını terminalde çalıştırarak, Ardupilot uygulama dosyaları kurmuş olduğumuz Git yazılımı ile bilgisayara indirilmektedir.

```
git clone https://github.com/ArduPilot/ardupilot.git
```

İndirmiş olduğumuz Ardupilot dosyasından aşağıdaki komut satırını kullanarak Copter Kütüphanesi versionunun uyumlu olup olmadığını kontrol etmektedir. Copter 3-6 versionundan yeni version var ise otomatik şekilde yeni versionu güncellemektedir.

```
cd ardupilot
git checkout Copter-3.6
git submodule update --init –recursive
```

Ardupilot uygulamasında engelden kaçınmak için tasarlanmış dronun özelliklerini içeren yazılım kütüphaneleri bir satıra ekleyerek Python dilindeki kütüphaneleri yüklemektedir.

```
sudo apt install python-matplotlib python-serial python-wxgtk3.0 python-wxtools python-lxml
python-scipy python-opencv ccache gawk python-pip python-pexpect
```

MAVProxy, İHA'lar için tam işlevli bir GCS'dir. Amaç, MAVLink protokolünü destekleyen (ArduPilot kullanan biri gibi) herhangi bir UAV için minimalist, taşınabilen ve genişletilebilen bir GCS'dir. MAVProxy uygulamasını Python PİP (Python package installer) yardımıyla indirilmektedir.

```
sudo pip install future pymavlink MAVProxy
gedit ~/.bashrc
```

Linux işletim sisteminden varsayılan text editor kullanarak vim veya sublime text programını kullarak aşağıdaki komutları son satıra eklemekteyiz ve bu komutlar ileride çalıştırmak istediğimiz dosyaların yollarını belirtmektedir.

```
export PATH=$PATH:$HOME/ardupilot/Tools/autotest
export PATH=/usr/lib/ccache:$PATH
```

Parametreleri ayarlamak için SITL'yi (Döngüdeki Yazılım) çalıştırılmaktadır.

Ubuntu Linux 16.04 LTS veya üzeri için QGroundControl'ü kurulumu. QGroundControl, PX4 veya ArduPilot ile çalışan araçlar için tam uçuş kontrolü ve araç kurulumu sağlar. QGroundControl AppImage uygulaması indirmektedir.

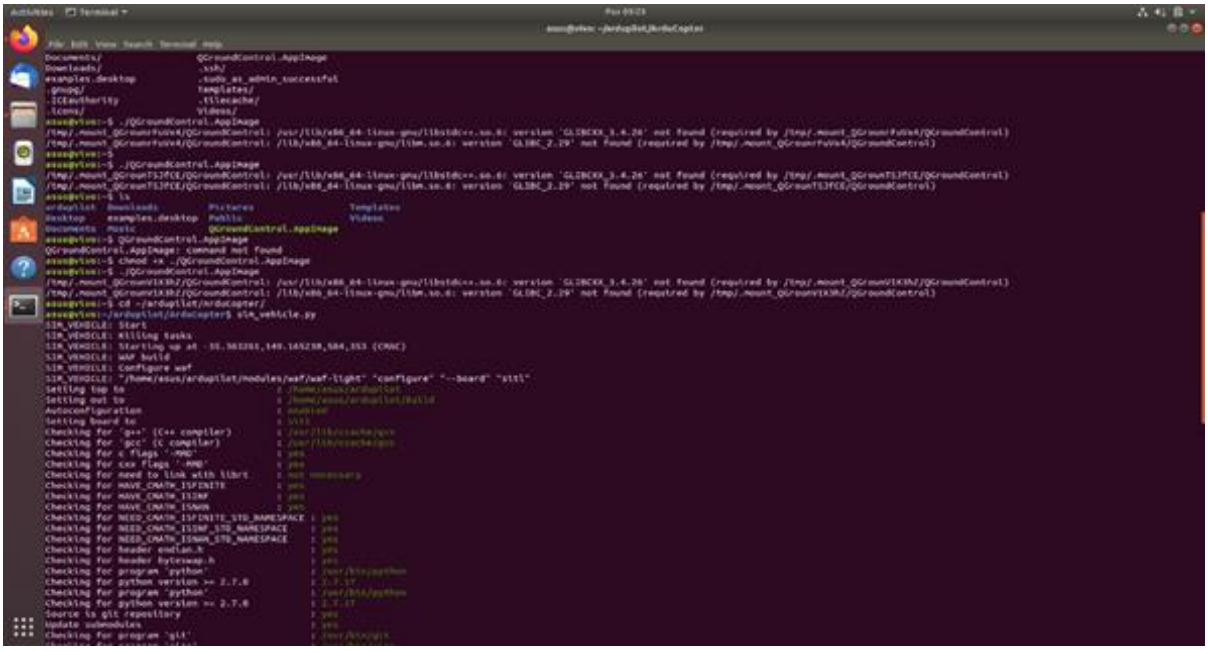
```
wget https://s3-us-west 2.amazonaws.com/qgroundcontrol/latest/QGroundControl.AppImage
```

QGroundControl'ü kurmadan önce modem yöneticisini kaldırmalı ve seri bağlantı noktasına erişim için izin vermelisiniz. Video akışını desteklemek için GStreamer'ı da yüklemeniz gerekir. Aşağıdaki komut izin modunu değiştirerek uygulamayı çalıştırmaktadır.

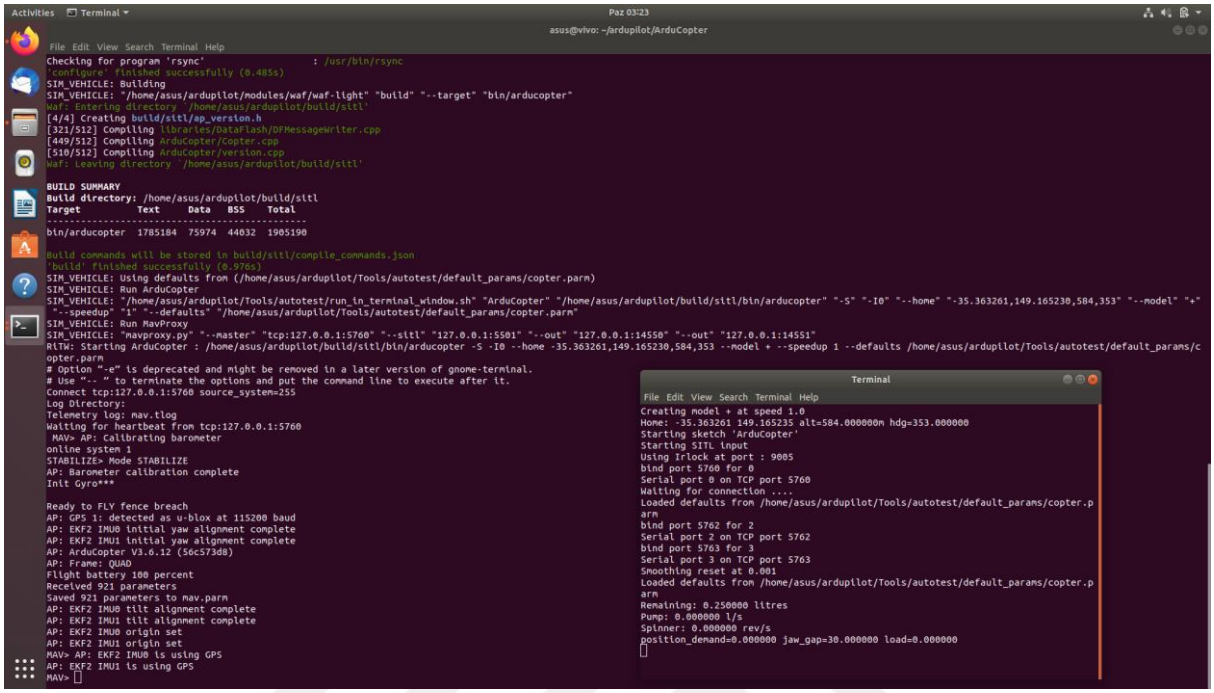
```
chmod +x ./QGroundControl.AppImage
./QGroundControl.AppImage
```

SITL'i çalıştırarak QGroundControl ile bağlantı kurulmaktadır. QGroundControl basit ve tutarlı bir komut satırı arabirimi sunduğundan genellikle açıklamak için MAVProxy'i kullanır. Bu işlemlerin çoğu, Görev Planlayıcı'da veya başka herhangi bir GCS'de de gerçekleştirilebilir.

```
cd ~/ardupilot/ArduCopter/
sim_vehicle.py
```



Şekil 3.1.2: GStreamer'in yüklemesi



Şekil 3.1.3: SITL'i çalıştırarak QGroundControl ile bağlantı kurulması

Robot simülasyonu, her robotistin araç kutusunda bulunan önemli bir araçtır. İyi tasarlanmış bir simülatör, algoritmaları hızlı bir şekilde test etmeyi, robotları tasarlamayı, regresyon testi gerçekleştirmeyi ve gerçekçi senaryolar kullanarak AI sistemini eğitmeyi mümkün kılmaktadır. Gazebo, karmaşık iç ve dış ortamlardaki robot popülasyonlarını doğru ve verimli bir şekilde simüle etme yeteneği sunmaktadır. Sağlam bir fizik motoru, yüksek kaliteli grafikler ve kullanışlı programatik ve grafik arayüzler parmaklarınızın ucunda. Hepsinden iyisi, Gazebo, canlı bir toplulukla ücretsiz olmasıdır.

3.2. UBUNTU İÇİN GAZEBO UYGULAMASININ KURULUMU.

Yeni terminalde Gazebo uygulamasının kurulumu gösterilmektedir.

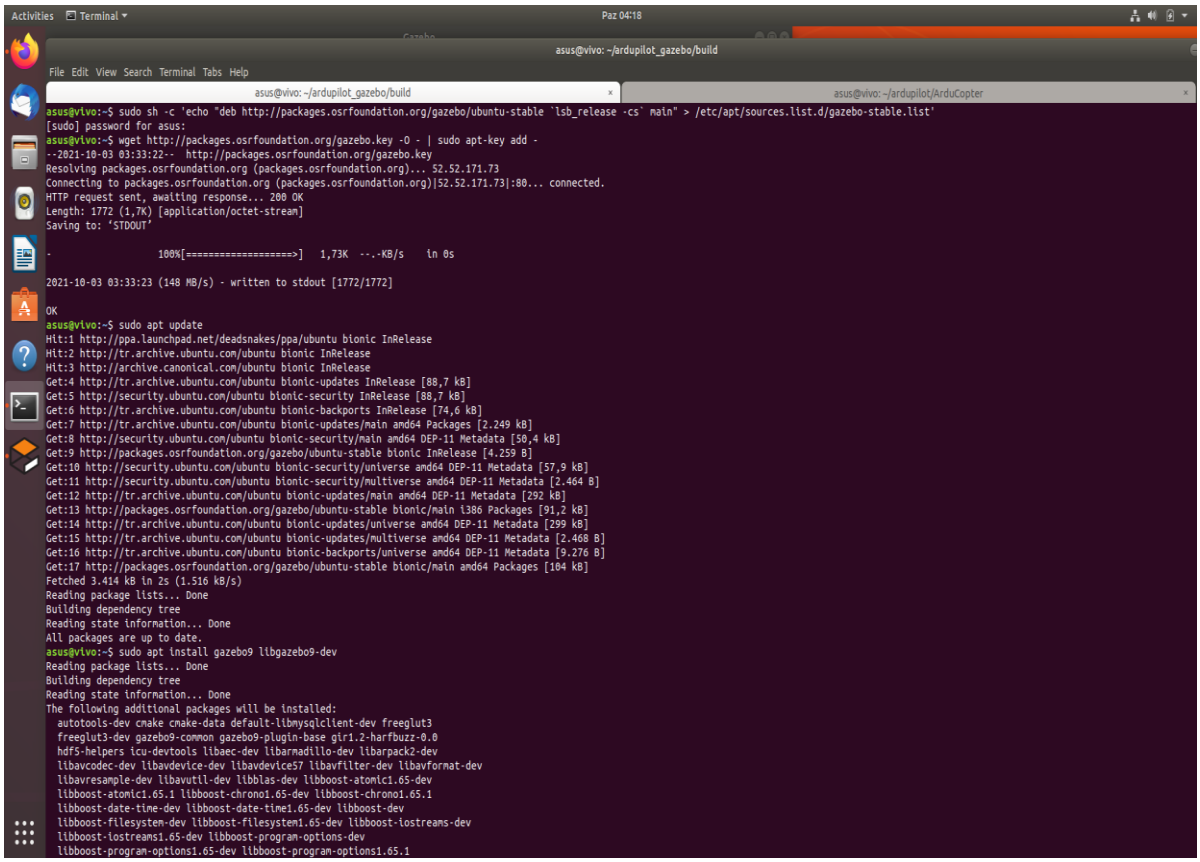
```
sudo sh -c 'echo "deb http://packages.osrfoundation.org/gazebo/ubuntu-stable `lsb_release -cs` main" > /etc/apt/sources.list.d/gazebo-stable.list'
```

Bilgisayarınızı, package.osrfoundation.org'dan gelen yazılımı kabul edecek şekilde ayarlama yaptıktan sonra Gazebo anahtar kurulumu asağıdaki komutlar ile yapılmaktadır.

```
wget http://packages.osrfoundation.org/gazebo.key -O - | sudo apt-key add -
sudo apt update
```

Ubuntu 18.04 versionunda Gazebo ile birlikte kullanacağımız açık kaynak Robotik Simülatörü Geliştirme Dosyalarının yüklenmesi gösterilmektedir.

```
sudo apt install gazebo9 libgazebo9-dev
```



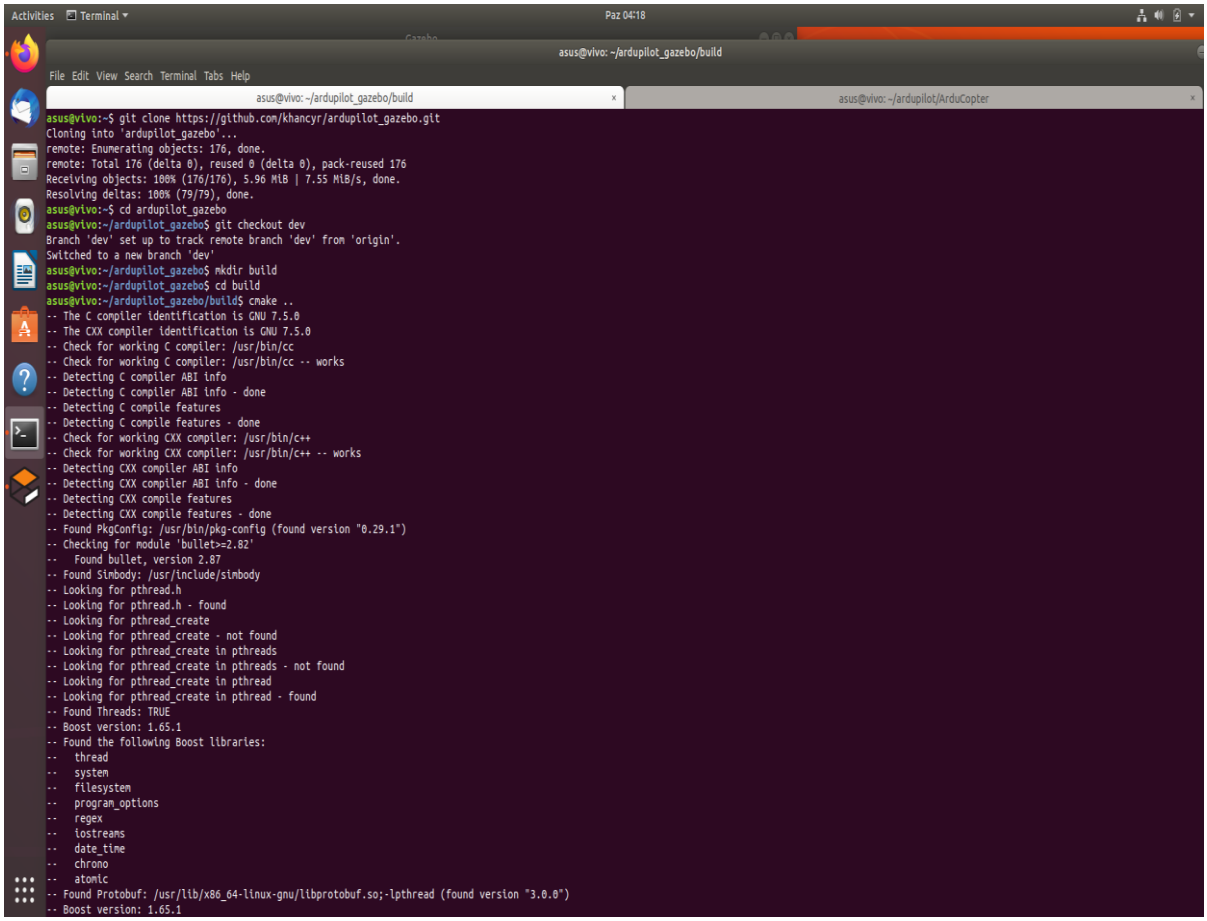
Şekil 3.2.1: Robotik Simülatörü Geliştirme Dosyalarının yüklenmesi

Aşağıdaki eklenti Gazebo eklentisidir, dolayısıyla onu kullanmak için ROS'a gerek yoktur. Gazebo ile ROS'u normal gazebo-ros paketleri ile kullanmaya devam edilebilmektedir. Eklentinin iki versiyonu vardır khancyr ve SwiftGust one. Her ikisi de aynı eklentiyi kullanır, yalnızca sağladıkları belgeler ve örneklerde farklılık gösterirler.

```
cd ~
git clone https://github.com/khancyr/ardupilot_gazebo.git
cd ardupilot_gazebo
```

Daha önce indirdiğimiz tüm kütüphaneleri bir ortamda derlemek için cmake komutu kullanılır. Gazebo pluginlerinin yüklemesi aşağıdaki komutlar ile yapılmaktadır.

```
mkdir build

cd build
cmake ..
make -j4
sudo make install
echo 'source /usr/share/gazebo/setup.sh' >> ~/.bashrc
```



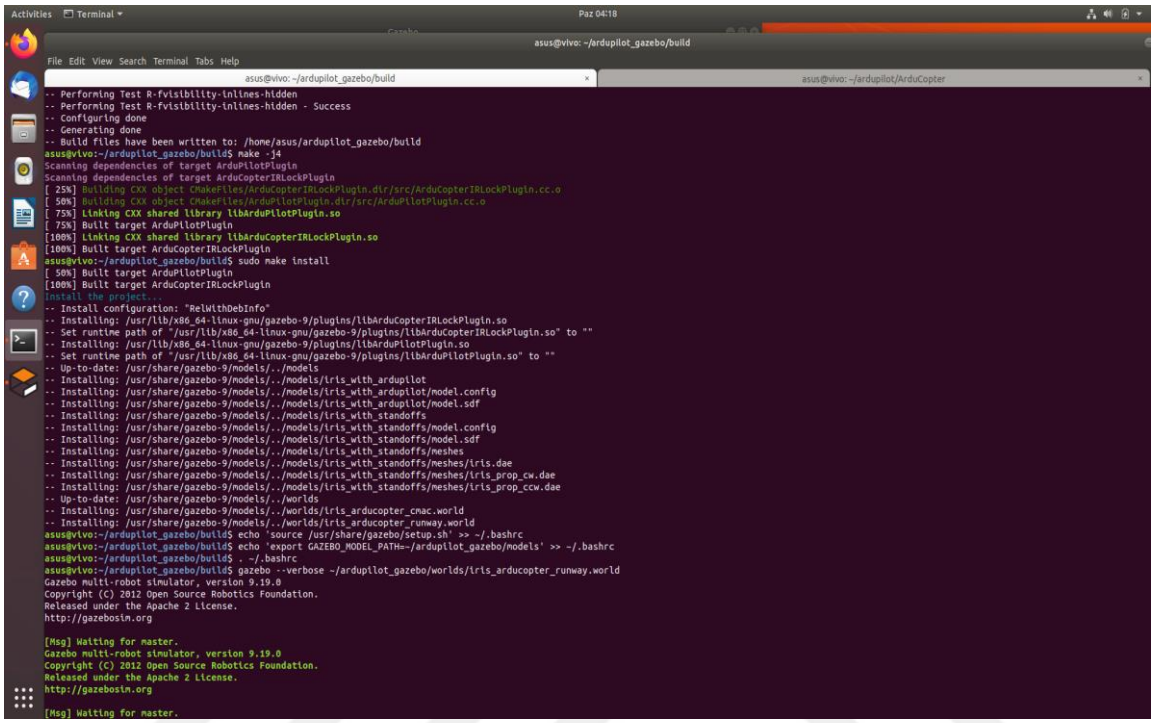
Şekil 3.2.2: Gazebo pluginlerinin yüklemesi

Gazebo Modellerinin Yol Ayarları aşağıdaki komutlar ile çalışmaktadır.

```
echo 'export GAZEBO_MODEL_PATH=~/ardupilot_gazebo/models' >> ~/.bashrc
. ~/.bashrc
```

Gazebo simulatörünün çalışması için ayrı ayrı 2 terminal kullanılmaktadır. İlk terminalde runway.world isimli haritayı çalıştırmaktadır.

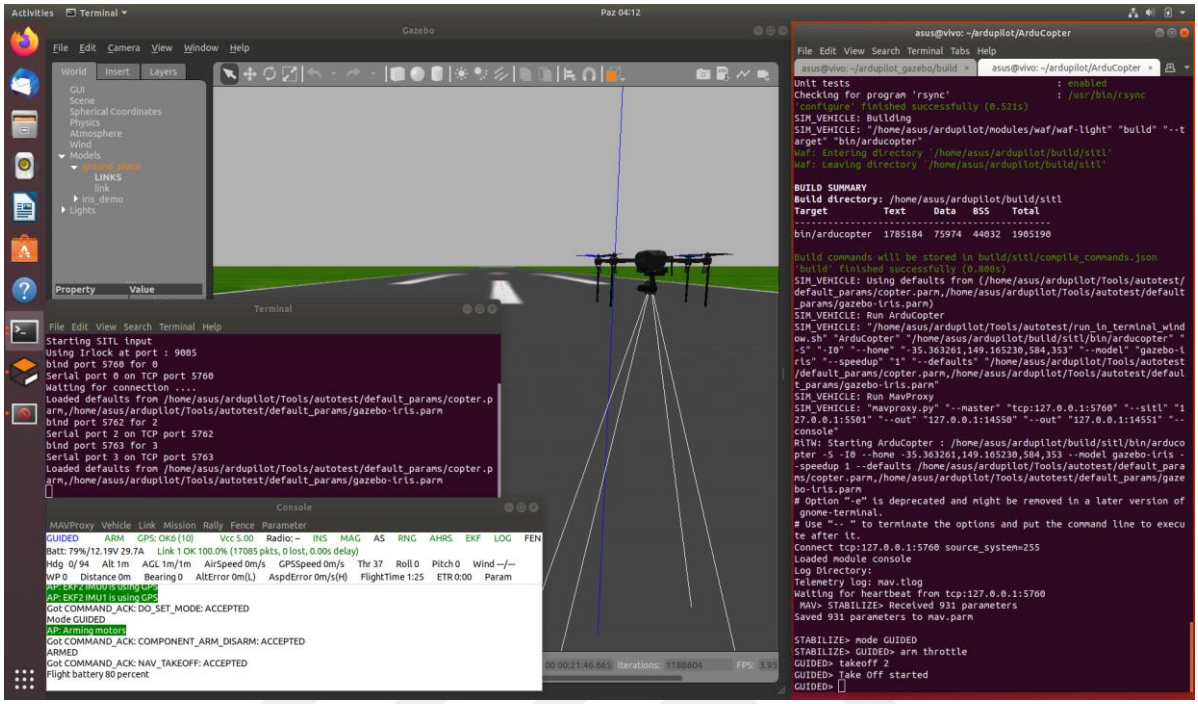
```
gazebo --verbose ~/ardupilot_gazebo/worlds/iris_arducopter_runway.world
```



Şekil 3.2.3: Gazebo simulatörü için runway.world isimli haritayı çalıştırması

İkinci terminalde ise ArduCopter'i kullanarak dron'umuzu uçuş hale getirmektedir.

```
cd ~/ardupilot/ArduCopter/
sim_vehicle.py -v ArduCopter -f gazebo-iris --console
```



Şekil 3.2.4: ArduCopter'i kullanarak dron'un uçuş hale getirilmesi

3.3. ROBOT İŞLETİM SİSTEMİ (ROS) MELODİC'İN UBUNTU'YA KURULUMU.

ROS, yazılım geliştiricilerin robot uygulamaları oluşturmasına yardımcı olacak kitaplıklar ve araçlar sağlamaktadır. Donanım soyutlaması, aygıt sürücüleri, kitaplıklar, görselleştiriciler, mesaj iletme, paket yönetimi ve daha fazlasını sağlar. ROS, açık kaynak BSD lisansı altında lisanslanmıştır. Bilgisayarı, package.ros.org'dan gelen yazılımı kabul edecek şekilde ayarlanmaktadır.

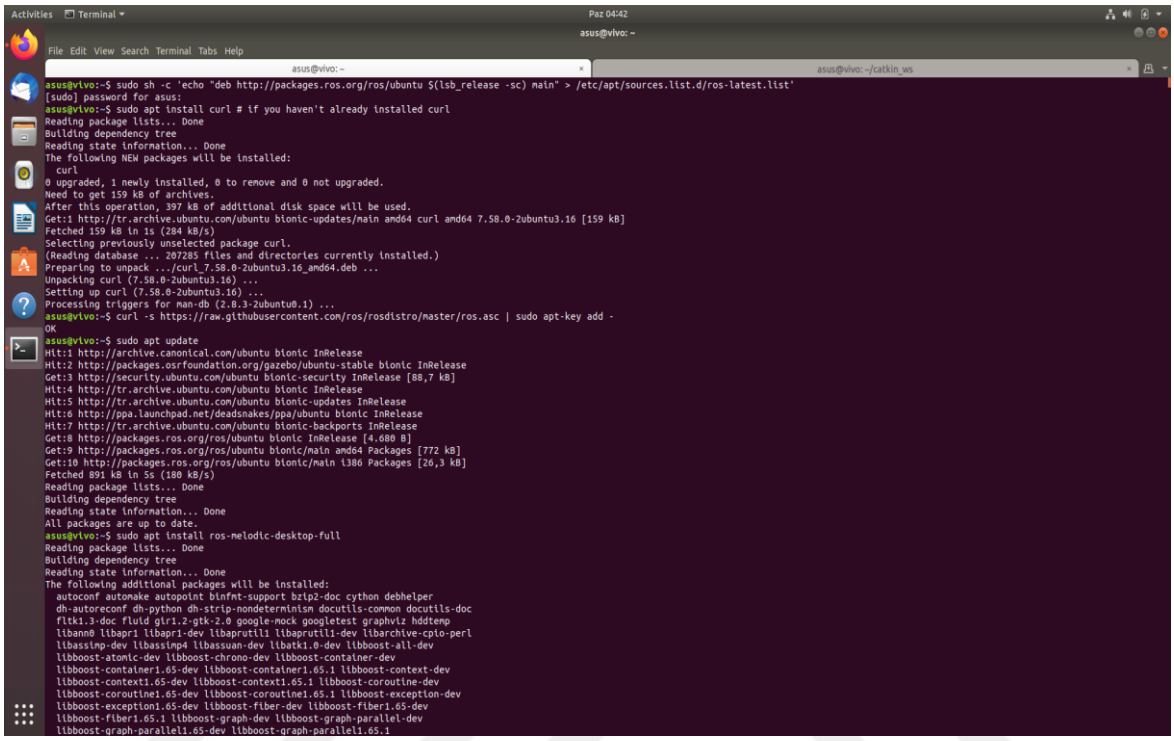
```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

Anahtarlar alttaki kod ile ayarlanmaktadır.

```
sudo apt install curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add –
```

ROS'ta birçok farklı kitaplık ve araç vardır. Başlaması için dört varsayılan yapılandırma sağlamaktadır. ROS paketlerini ayrı ayrı da kurabilmektedir. ROS, rqt, rviz, robot genel kitaplıkları, 2D/3D simülatörleri ve 2D/3D algısıdır.

```
sudo apt install ros-melodic-desktop-full
```



Şekil 3.3.1: Ortamın kurulumu

Her yeni kabuk başlatıldığında, ROS ortam değişkenlerinin bash oturumuna otomatik olarak eklenmektedir:

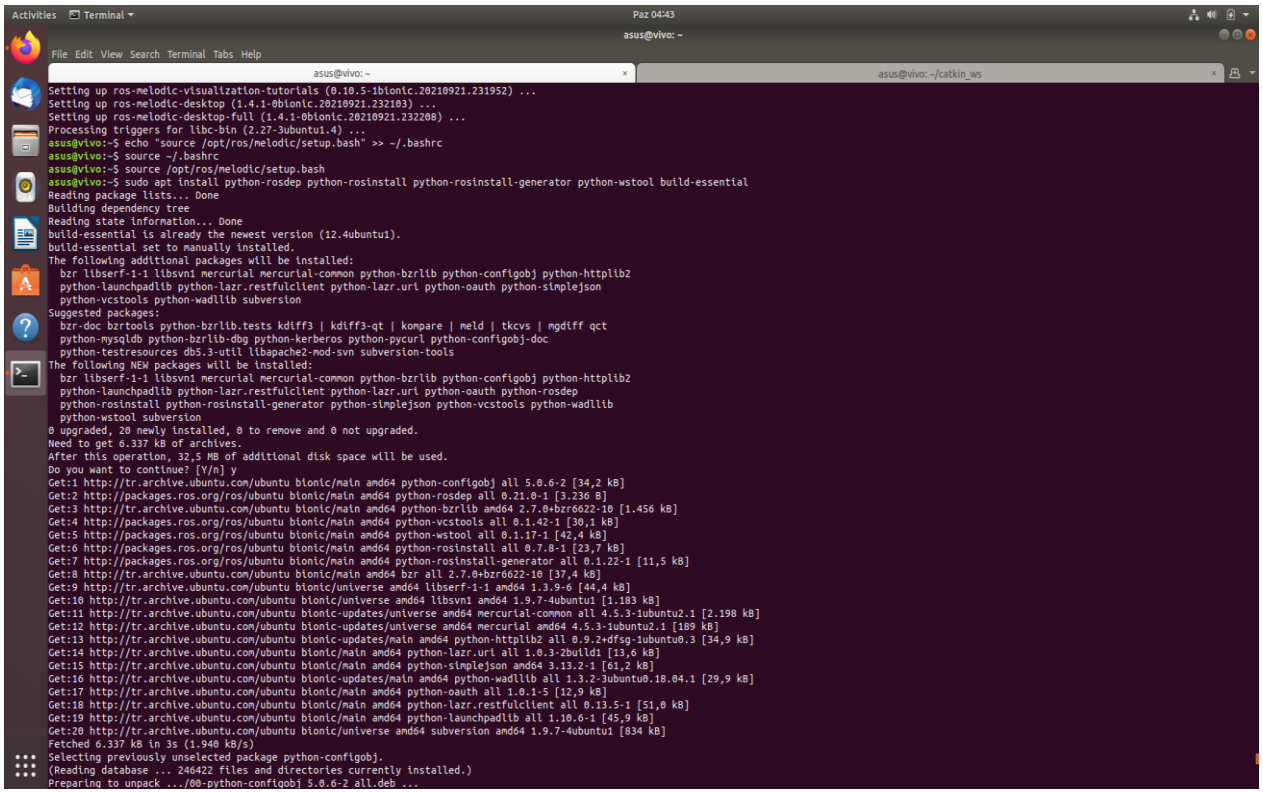
```
echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc source ~/.bashrc
```

3.4. PAKET OLUŞTURMA BAĞIMLILIKLARI.

Bugüne kadar Core ROS paketlerini çalıştırmak için ihtiyacı olanı kurulmaktadır. Kendi ROS çalışma alanlarınızı oluşturmak ve yönetmek için ayrı olarak dağıtılan çeşitli araçlar ve gereksinimler vardır. Mesela, ros install, tek bir komutla ROS paketleri için birçok kaynak ağacı indirmeyi kolaylaştıran popüler bir komut satırı aracıdır.

Mesela, ros install, tek bir komutla ROS paketleri için birçok kaynak ağacını kolayca indirmenizi sağlayan, sık kullanılan bir komut satırı aracıdır. Bu aracı ve ROS paketleri oluşturmaya yönelik diğer bağımlılıkları yüklemek için şunu çalıştırmaktadır.

```
sudo apt install python-rosdep python-rosinstall python-rosinstall-generator python-wstool
build-essential
```

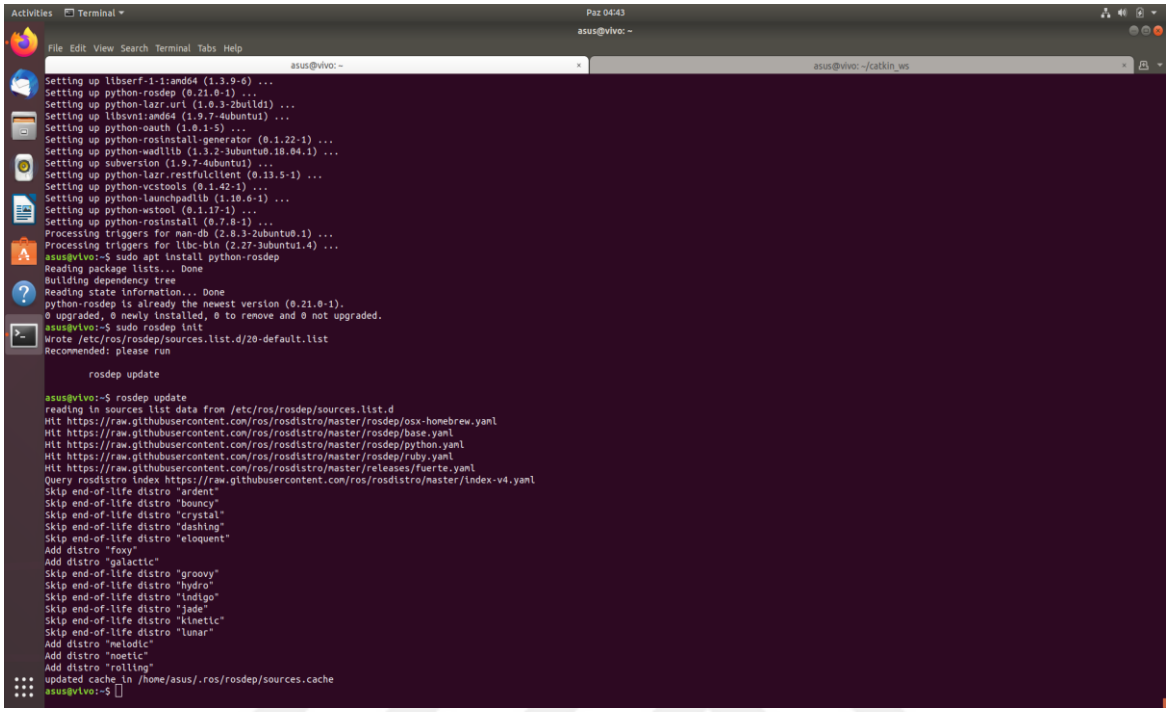


Şekil 3.4.1: ROS paketleri oluşturmaya yönelik diğer bağımlılıkların yüklenmesi

3.5. ROSDEP KURULUMU.

Birçok ROS aracını kullanmadan önce rosdep'i başlatmaktadır. Rosdep, derlemek istediğiniz kaynak için sistem bağımlılıklarını kolayca kurmanızı sağlar ve bazı çekirdek bileşenleri ROS'ta çalıştırmak için gereklidir.

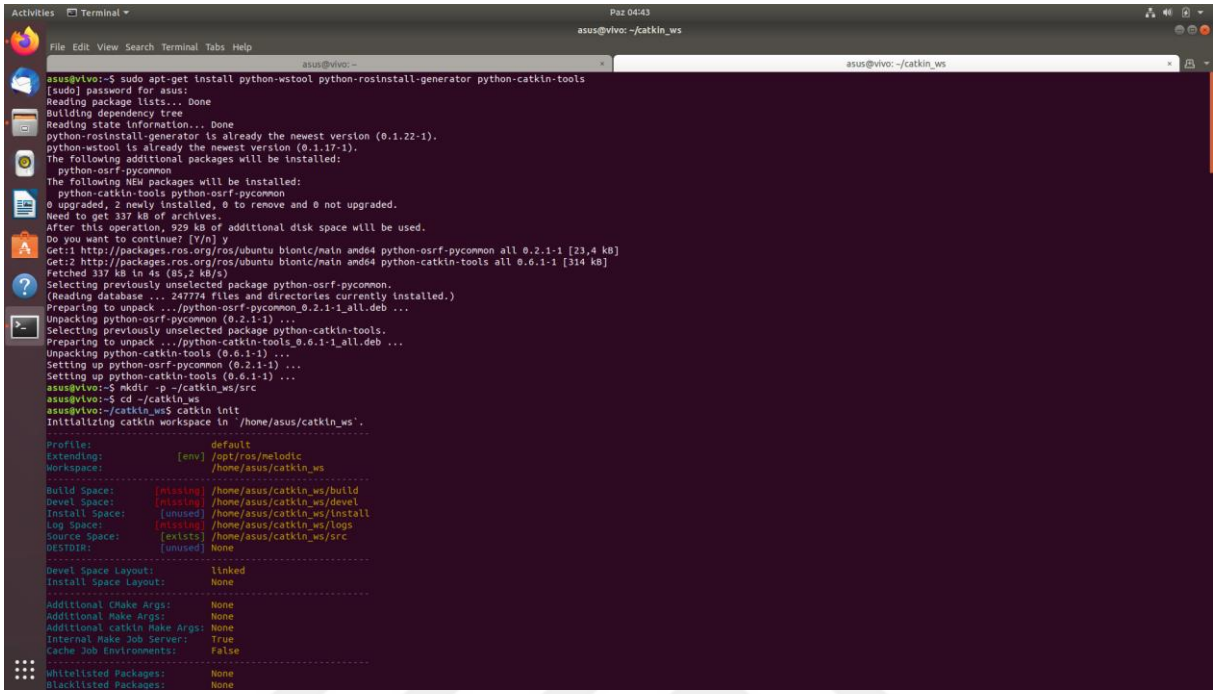
```
sudo apt install python-rosdep

sudo rosdep

init rosdep update
```



Şekil 3.5.1: Rosdep Kurulumu

Ros install generator, ROS paketleriyle depolar hakkında bilgi içeren ros install dosyaları oluşturmaktadır. Engelden kaçınmasında hesaplanacak değer kütüphanelerinin kurulumu aşağıdaki kod ile yapılmaktadır.

```
sudo apt-get install python-wstool python-rosinstall-generator python-catkin-tools
mkdir -p ~/catkin_ws/src
cd ~/catkin_ws
catkin init
```



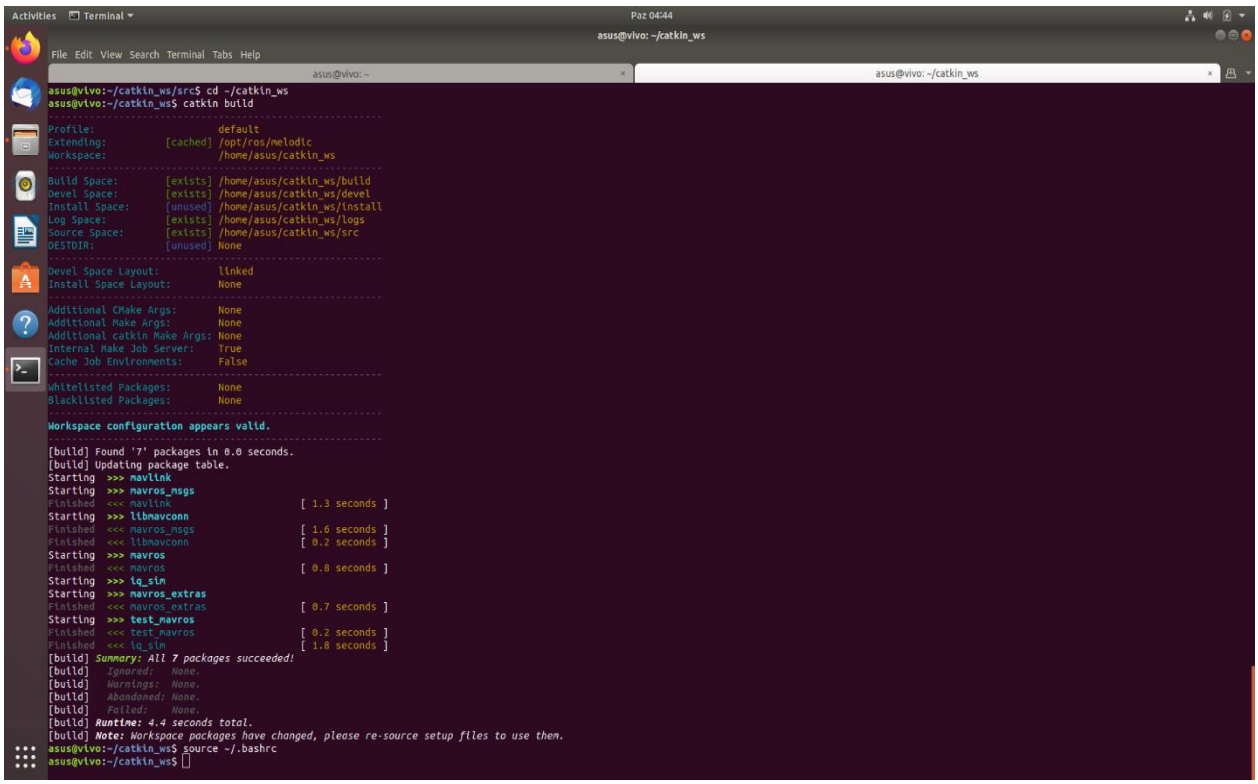
Şekil 3.5.2: Catkin'de kütüphanelerin kurulumu

Mavros herhangi bir MAVLink etkin otopilot, yer istasyonu veya çevre birimi için ROS çalıştıran bilgisayarlar arasında MAVLink genişletilebilir iletişimi sağlayan bir ROS paketidir. MAVROS, ROS ile MAVLink protokolü arasındaki resmi desteklenen köprüdür. Mavros ve mavlink kütüphanelerini indirerek yüklenmektedir.

```
cd ~/catkin_ws
wstool init ~/catkin_ws/src
rosinstall_generator --upstream mavros | tee /tmp/mavros.rosinstall
rosinstall_generator mavlink | tee -a /tmp/mavros.rosinstall
wstool merge -t src /tmp/mavros.rosinstall
wstool update -t src
rosdep install --from-paths src --ignore-src --rosdistro `echo $ROS_DISTRO` -y
catkin build
```

Aşağıdaki komutu çalıştırarak ~/.bashrc sonuna bir satır ekleyerek global değişkenleri güncellemektedir.

```
echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc
source ~/.bashrc
```



Şekil 3.5.3: Global değişkenlerin güncellenmesi

GeographicLib, küçük (ama önemli!) bir dizi coğrafi dönüşüm için bir C++ arabirimi sunar. Jeomanyetik modeller ve jeodezik hesaplamalar için sınıflar sağlar. İleri aşamada nano hesaplamalar yapmak için kullanılmaktadır.

```
sudo ~/catkin_ws/src/mavros/mavros/scripts/install_geographiclib_datasets.sh
```

Depomuz şimdi ~/catkin_ws/src/iq_sim/ dizinine kopyalayıp Gazebo'ya iq modellerini nerede arayacağınızı söylemek için aşağıdaki komutu çalıştırılmaktadır.

```
echo
"GAZEBO_MODEL_PATH=${GAZEBO_MODEL_PATH}:$HOME/catkin_ws/src/iq_sim/
models" >> ~/.bashrc
```

3.6. OTONOM DRON'LAR İÇİN ROS'A GİRİŞ.

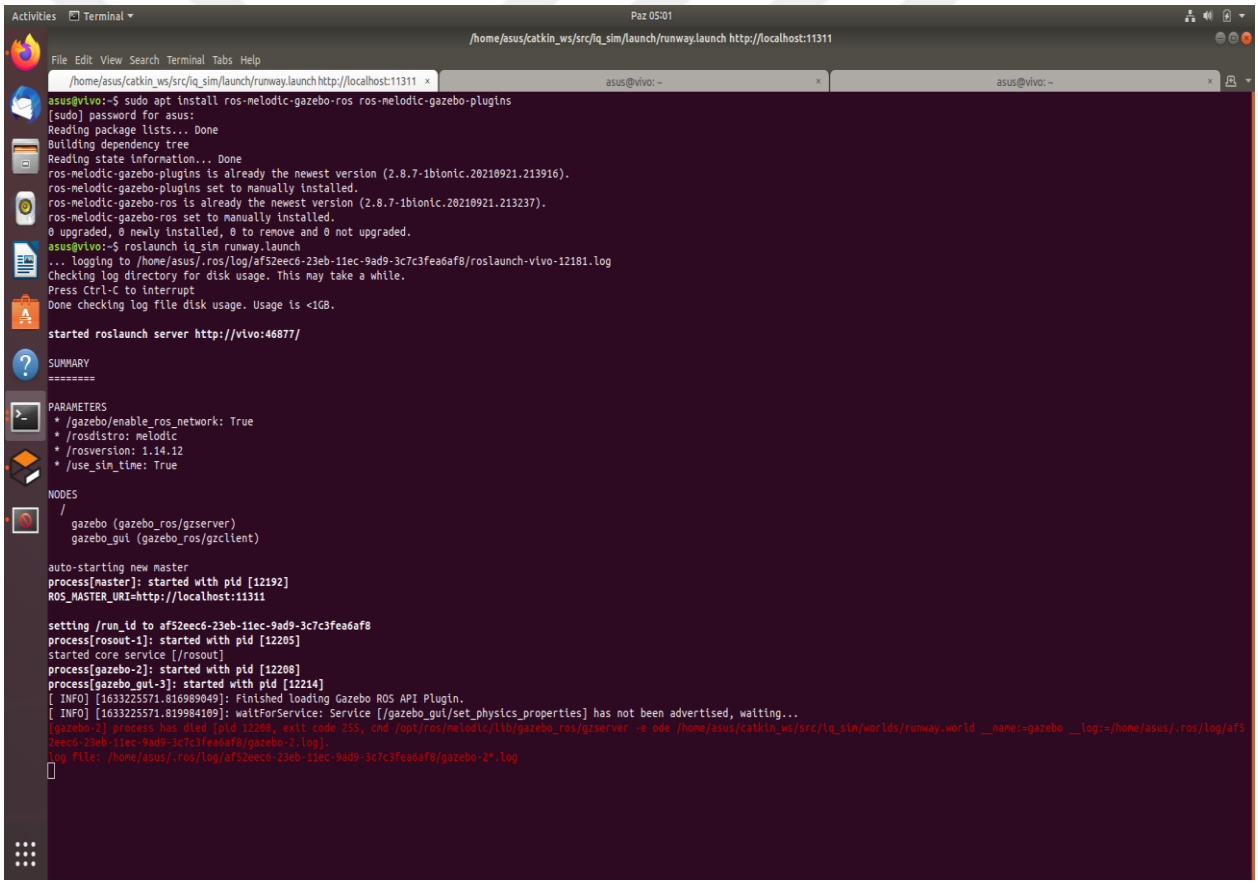
ROS (Robot İşletim Sistemi) bir işletim sistemi olmamasına rağmen, donanım soyutlama, düşük seviyeli cihaz kontrolü, yaygın olarak kullanılan işlevlerin uygulanması, süreçler

arasında mesaj geçişi ve paket yönetimi gibi heterojen bir bilgisayar kümesi için tasarlanmış hizmetler sunar. Çoğunlukla mesaj aktarma işlevini kullanacağız. Bunu göstermek için simülatörümüzü yeniden başlatılmaktadır ve birkaç komut çalıştırılmaktadır.
Gazebo için ROS eklentileri yüklenmektedir.

```
sudo apt install ros-melodic-gazebo-ros ros-melodic-gazebo-plugins
```

Gazebo World'ü başlatıp, Gazebo dünyamızı aşağıdaki gibi ROS ile başlatılmaktadır.

```
roslaunch iq_sim runway.launch
```



Şekil 3.6.1: Gazebo dünyasının ROS ile başlatılması

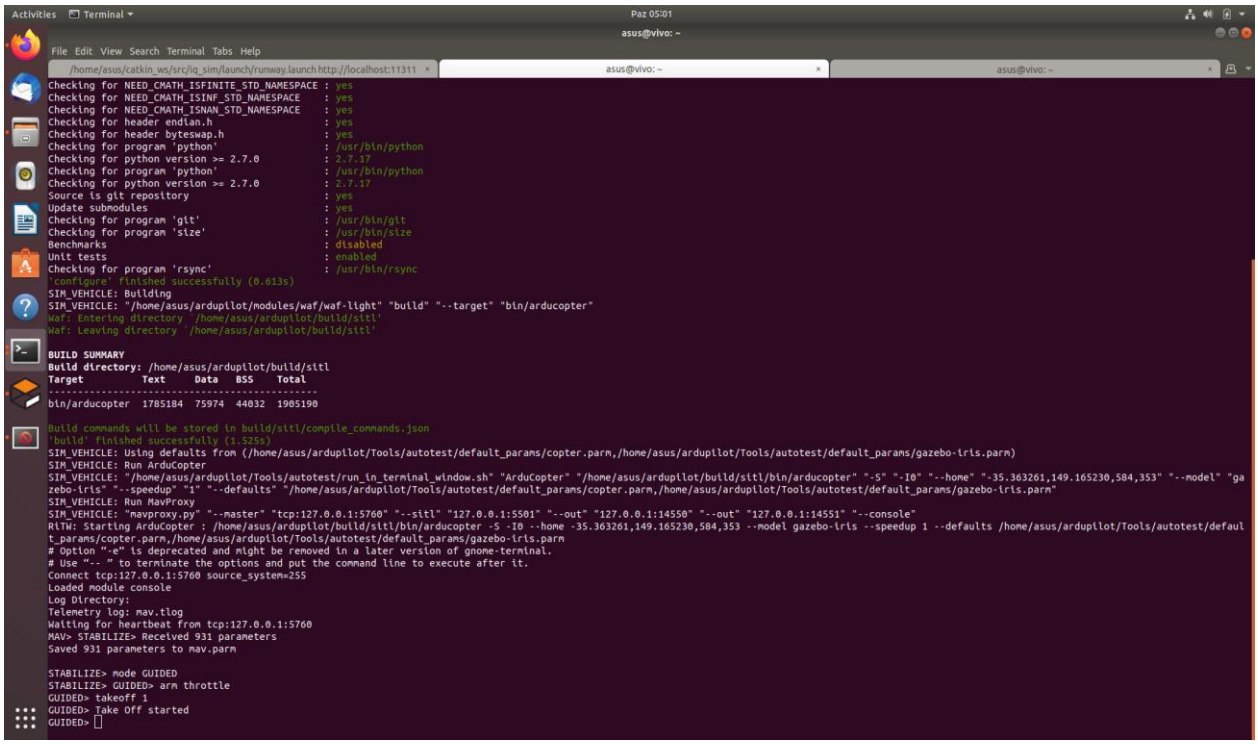
Ayrıca ArduCopter simülatörünü de piyasaya sürmektedir. Yukarıdaki cd ~/ardupilot/ArduCopter/ && sim_vehicle.py -v ArduCopter -f gazebo-iris --console büyük komutunu hatırlamak zorunda kalmamamız için bir komut dosyası hazırlanmaktadır. Erişim

kolaylığı için komut dosyasını ana klasöre taşıyabilmektedir. Bunu aşağıdaki komutu çalıştırarak yapılmaktadır.

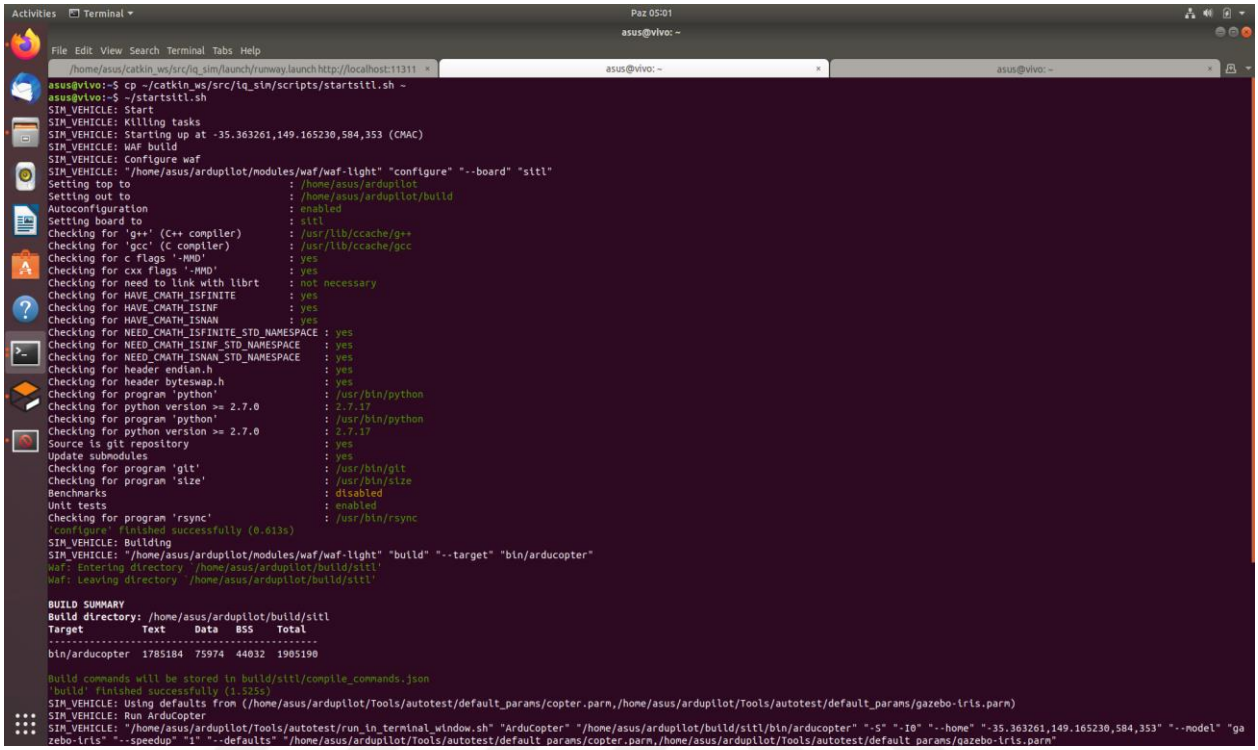
```
cp ~/catkin_ws/src/iq_sim/scripts/startsitl.sh ~
```

Ardupilot sitl'ı çalıştırarak başlatılmaktadır.

```
~/startsitl.sh
```



Şekil 3.6.2: Ardupilot sitl'ı çalıştırılması



Şekil 3.6.3: Ardupilot SİTL'ı çalıştırılması

ROS Komut Satırı Araçlarına Giriş

Yeni bir terminalde (ctrl+shift+T)

```
rostopic list
```

Doğru kurulum yapıp aşağıdaki satırları görebilmektedir.

```
/clock
/gazebo/link_states
/gazebo/model_states
/gazebo/parameter_descriptions
/gazebo/parameter_updates
/gazebo/set_link_state
/gazebo/set_model_state
/rosout
/rosout_agg
/webcam/camera_info
/webcam/image_raw
/webcam/image_raw/compressed
/webcam/image_raw/compressed/parameter_descriptions
/webcam/image_raw/compressed/parameter_updates
/webcam/image_raw/compressedDepth
/webcam/image_raw/compressedDepth/parameter_descriptions
/webcam/image_raw/compressedDepth/parameter_updates
/webcam/image_raw/theora
/webcam/image_raw/theora/parameter_descriptions
/webcam/image_raw/theora/parameter_updates
/webcam/parameter_descriptions
/webcam/parameter_updates
```

Şu anda yayınlanmakta olan farklı konuları gösterir. Bu konular, bir kameradan alınan görüntüler gibi verilerin kaynağı hakkında veriler içerir. Aşağıdakileri çalıştırarak hangi verilerin yayınlandığını görebilmektedir.

```
rostopic echo /gazebo/model_states
```

Şimdi dron'u uçursak, etrafta uçarken pozisyondaki değişiklikleri görebileceğiz. Yükarıda yaptığımız gibi mavproxy terminalinde aşağıdakileri çalıştırarak dron'u uçurabilmektedir.

```
mode guided
arm throttle
takeoff 15
```

3.7. FCU'DAN TELEMETRİ VERİLERİ ALMAK İÇİN MAVROS'U KULLANMAK.

Şimdi /gazebo/model_states konusu, simülatördki gerçek model konumudur. Bu gerçek hayatta kullanabileceğimiz bir şey değil. Gerçek hayatta, sensörlerinin bir kombinasyonundan formüle edilen dron pozisyonunun tahminini kullanmak zorundadır. Bu konum, mavlink adı verilen bir

iletişim protokolü kullanılarak iletmektedir. Bu mesajlar soyulur ve radyo iletimi için optimize edilmektedir. MAVROS, MAVlink mesajlarını, kullanımı kolay ve farklı robot sistemleri arasında yaygın olan ROS mesajlarına çeviren bir aracıdır. Mavros çalıştırmayı başlatmak için

```
roslaunch iq_sim apm.launch
```

rostopic list çalıştırdığınızda bir sürü mavros konusu görebilmektedir.

```
/clock
/diagnostics
/gazebo/link_states
/gazebo/model_states
/gazebo/parameter_descriptions
/gazebo/parameter_updates
/gazebo/set_link_state
/gazebo/set_model_state
/mavlink/from
/mavlink/to
/mavros/adsb/send
/mavros/adsb/vehicle
/mavros/battery
/mavros/cam_imu_sync/cam_imu_stamp
/mavros/companion_process/status
/mavros/distance_sensor/rangefinder_pub
/mavros/extended_state
/mavros/fake_gps/mocap/tf
/mavros/global_position/compass_hdg
/mavros/global_position/global
/mavros/global_position/gp_lp_offset
/mavros/global_position/gp_origin
/mavros/global_position/home
/mavros/global_position/local
/mavros/global_position/raw/fix
/mavros/global_position/raw/gps_vel
/mavros/global_position/rel_alt
/mavros/global_position/set_gp_origin
/mavros/gps_rtk/send_rtcm
/mavros/home_position/home
/mavros/home_position/set
/mavros/imu/data
```

/mavros/imu/data_raw
/mavros/imu/diff_pressure
/mavros/imu/mag
/mavros/imu/static_pressure
/mavros/imu/temperature_baro
/mavros/imu/temperature_imu
/mavros/landing_target/lt_marker
/mavros/landing_target/pose
/mavros/landing_target/pose_in
/mavros/local_position/accel
/mavros/local_position/odom
/mavros/local_position/pose
/mavros/local_position/pose_cov
/mavros/local_position/velocity_body
/mavros/local_position/velocity_body_cov
/mavros/local_position/velocity_local
/mavros/log_transfer/raw/log_data
/mavros/log_transfer/raw/log_entry
/mavros/manual_control/control
/mavros/manual_control/send
/mavros/mission/reached
/mavros/mission/waypoints
/mavros/mocap/pose
/mavros/obstacle/send
/mavros/odometry/in
/mavros/setpoint_velocity/cmd_vel_unstamped
/mavros/state
/mavros/statustext/recv
/mavros/statustext/send
/mavros/time_reference
/mavros/timesync_status
/mavros/trajectory/desired
/mavros/trajectory/generated
/mavros/trajectory/path
/mavros/vfr_hud
/mavros/vision_pose/pose
/mavros/vision_pose/pose_cov
/mavros/wind_estimation
/rosout
/rosout_agg
/tf
/tf_static
/webcam/camera_info
/webcam/image_raw
/webcam/image_raw/compressed
/webcam/image_raw/compressed/parameter_descriptions

/webcam/image_raw/compressed/parameter_updates
/webcam/image_raw/compressedDepth
/webcam/image_raw/compressedDepth/parameter_descriptions
/webcam/image_raw/compressedDepth/parameter_updates
/webcam/image_raw/theora
/webcam/image_raw/theora/parameter_descriptions
/webcam/image_raw/theora/parameter_updates
/webcam/parameter_descriptions
/webcam/parameter_updates

Artık dron'ların konumunu yerel çerçevede çalıştırarak görebiliyoruz. Aşağıdaki eğitimlerde, C++ programlarımızda bu konulardaki veriler erişmektedir. Yayınlanmakta olan mesajın türünü görmek için çalıştırılmaktadır.

rostopic list -v /mavros/global_position/local

Konunun mesajı nav_msgs/Odometry şeklinde yayınladığını görülmektedir. Mesajın yapısını görmek için aşağıdakiler çalıştırılmaktadır.

rosmsg show nav_msgs/Odometry

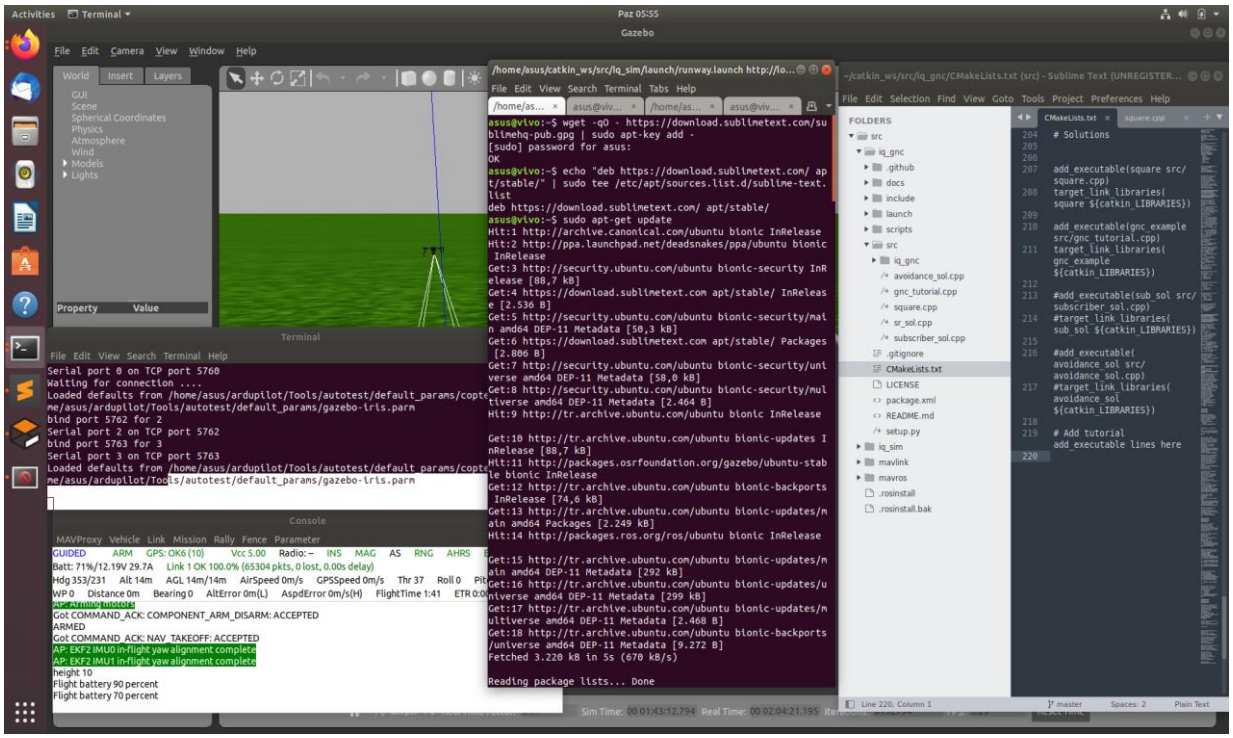
Şekil 3.7.1: Mesajın yapısı

3.8. NAVİGASYON VE KONTROL

Aşağıdaki eğitim, dron'nuzu ara noktalara göndermenize izin veren basit bir programın nasıl yapıldığını göstermektedir. Bu eğitim, kalkış, kara, ara nokta navigasyonu ve navigasyonla

ilişkili tüm referans çerçeveleri dahil olmak üzere çeşitli uçuş işlemlerini yöneten bir dizi üst düzey işleve sahip olan benim API'mi kullanılmaktadır.

```
wget -qO - https://download.sublimetext.com/sublimehq-pub.gpg | sudo apt-key add -
echo "deb https://download.sublimetext.com/ apt/stable/" | sudo tee /etc/apt/sources.list.d/sublime-text.list
sudo apt-get update
sudo apt-get install sublime-text
```



Şekil 3.7.1: API kullanımı

IQ GNC ROS paketi indirilmektedir. İlk olarak, IQ GNC ROS paketini klonlamaktadır. Bu ROS paketi, dron'un komut dosyasını yazmayı kolaylaştıracak GNC API'm ile birlikte gelmektedir. Ayrıca bu eğitim için hazırlanmış bir çözümle birlikte gelmektedir.

```
git clone https://github.com/Intelligent-Quads/iq_gnc.git
```

Bu paket gnc_functions. hpp dosyasını içerir. Bu dosya, akıllı dron uygulamaları oluşturmak için bir dizi kullanışlı işlev içermektedir. q_gnc paketini indirdikten sonra

Mission8_OutOfControls/src içinde square. cpp adlı yeni bir dosya oluşturmaktadır. Ardından CMakeLists.txt dosyasını açın ve aşağıdakini en alta eklenmektedir.

```
add_executable(square src/square.cpp)
target_link_libraries(square ${catkin_LIBRARIES})
```

İlk önce kontrol fonksiyonları dahil etmektedir.

```
#include <gnc_functions. hpp>
```

Bu, tüm kontrol fonksiyonlarımızı ve yapılarımızı kullanmamıza izin verecektir. Ardından ana işlevi ekleyip ve ros'u başlatmaktadır.

```
int main(int argc, char** argv)
{
        //initialize ros
        ros::init(argc, argv, "gnc_node");
        ros::NodeHandle gnc_node;
        //Rest of code here
}
```

Daha sonra init_publisher_subscriber() fonksiyonunu eklemektedir. Bu işlev, ros düğümü tanıtıcımızı girdi olarak alır ve otomatik pilotumuzdan gerekli bilgileri toplayacak aboneleri başlatmaktadır. Daha sonra ön kontrol işlemlerini yürütmek için aşağıdaki işlevler eklemektedir.

```
init_publisher_subscriber(gnc_node);

// wait for FCU connection
wait4connect();

//wait for used to switch to mode GUIDED
wait4start();

//create local reference frame
initialize_local_frame();
```

Wait4connect () işlevi, düğüm uçuş kontrol ünitesi (FCU) ile iletişim kurana kadar döngüde kalacaktır. FCU ile bağlantı kurulduktan sonra, pilot FCU uçuş modunu GUIDED olarak değiştirerek programı yürütene kadar programı tutmak için wait4start () işlevini kullanmaktadır. Bu, Mission Planner veya QGroundControl gibi bir yer kontrol istasyonundan (GCS) veya bir radyo denetleyicisindeki bir anahtardan yapılmaktadır. Son olarak, görevi yürütme komutu gönderildiğinde, navigasyon çerçevenizi oluşturmak için initialize_local_frame() işlevini kullanmaktadır. Bu işlev, dron'nun başlangıç konumuna göre yerel referans çerçevesi oluşturmaktadır.

Daha sonra kalkış talep edip (float takeOffHeight) işlevini kullanmaktadır.

```
takeoff(3);
```

GNC API, gnc_api_waypoint yapısını içerir; bu yapı, dron'nuzun konumlarını ve yönelimlerini ayarlamak için kullanabileceğiniz x y z psi değişkenlerini içermektedir. Dron'muzun bir karede uçmasını sağlamak için aşağıdaki yol noktalarını belirtmektedir.

```
//specify some waypoints
        std::vector<gnc_api_waypoint> waypointList;
        gnc_api_waypoint nextWayPoint;
        nextWayPoint.x = 0;
        nextWayPoint.y = 0;
        nextWayPoint.z = 3;
        nextWayPoint.psi = 0;
        waypointList.push_back(nextWayPoint);
        nextWayPoint.x = 5;
        nextWayPoint.y = 0;
        nextWayPoint.z = 3;
        nextWayPoint.psi = -90;
        waypointList.push_back(nextWayPoint);
        nextWayPoint.x = 5;
        nextWayPoint.y = 5;
        nextWayPoint.z = 3;
        nextWayPoint.psi = 0;
        waypointList.push_back(nextWayPoint);
        nextWayPoint.x = 0;
        nextWayPoint.y = 5;
        nextWayPoint.z = 3;
        nextWayPoint.psi = 90;
        waypointList.push_back(nextWayPoint);
        nextWayPoint.x = 0;
        nextWayPoint.y = 0;
        nextWayPoint.z = 3;
        nextWayPoint.psi = 180;
        waypointList.push_back(nextWayPoint);
```

Son olarak kontrol döngü eklenmektedir.

```
//specify control loop rate. We recommend a low frequency to not over load the FCU with messages. Too many messages will cause the dron to be sluggish
        ros::Rate rate(2.0);
        int counter = 0;
        while(ros::ok())
        {
                ros::spinOnce();
```

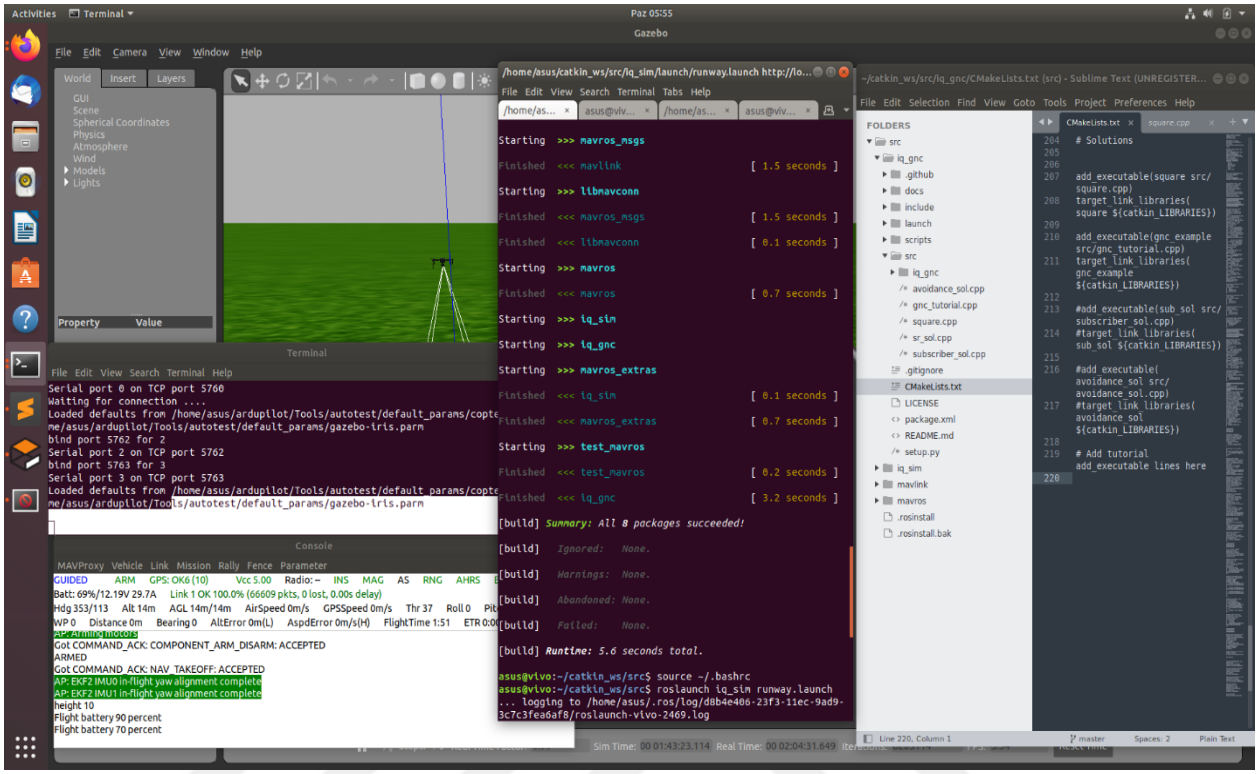
```
                    rate.sleep();
                    if(check_waypoint_reached() == 1)
                    {
                            if (counter < waypointList.size())
                            {
        set_destination(waypointList[counter].x,waypointList[counter].y,waypointList[coun
ter].z, waypointList[counter].psi);
                                    counter++;
                            } else {
                                    //land after all waypoints are reached
                                    land();
                            }
                    }

            }
            return 0;
}
```

Derleme kodu aşağıdaki gibidir.

```
cd ~/catkin_ws
catkin build
source ~/.bashrc
```

Aşağıdaki kod ile örnek çalıştırılmaktadır.

```
roslaunch iq_sim runway.launch
```



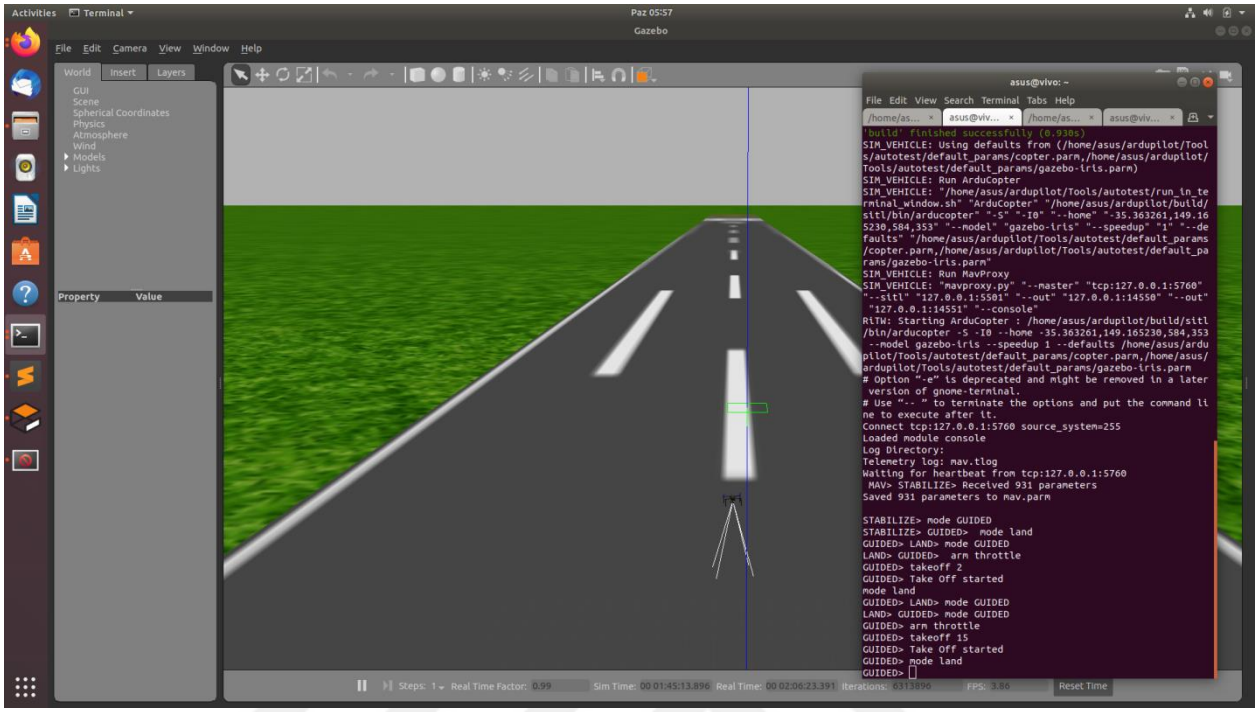
Şekil 3.7.2: SITL'in ve MavProxy'in dronu çalıştırma durumu

```
./startsitl.sh
roslaunch iq_sim apm.launch
rosrun iq_gnc square
```

Ctrl + shift + t tuşlarına basarak gnome terminallerini açıyoruz. Son olarak, uçuş modunu MAVproxy terminalinde rehberli olarak değiştirerek görevi çalıştırılmaktadır.

```
mode guided
```

Artık dron'umuzu kontrol etmek için mevcut olan fonksiyonlar hakkında temel bir anlayışa sahip olmaktadır. Daha karmaşık navigasyon kodu oluşturmamıza yardımcı olması için bu işlevleri kullanabilmektedir.



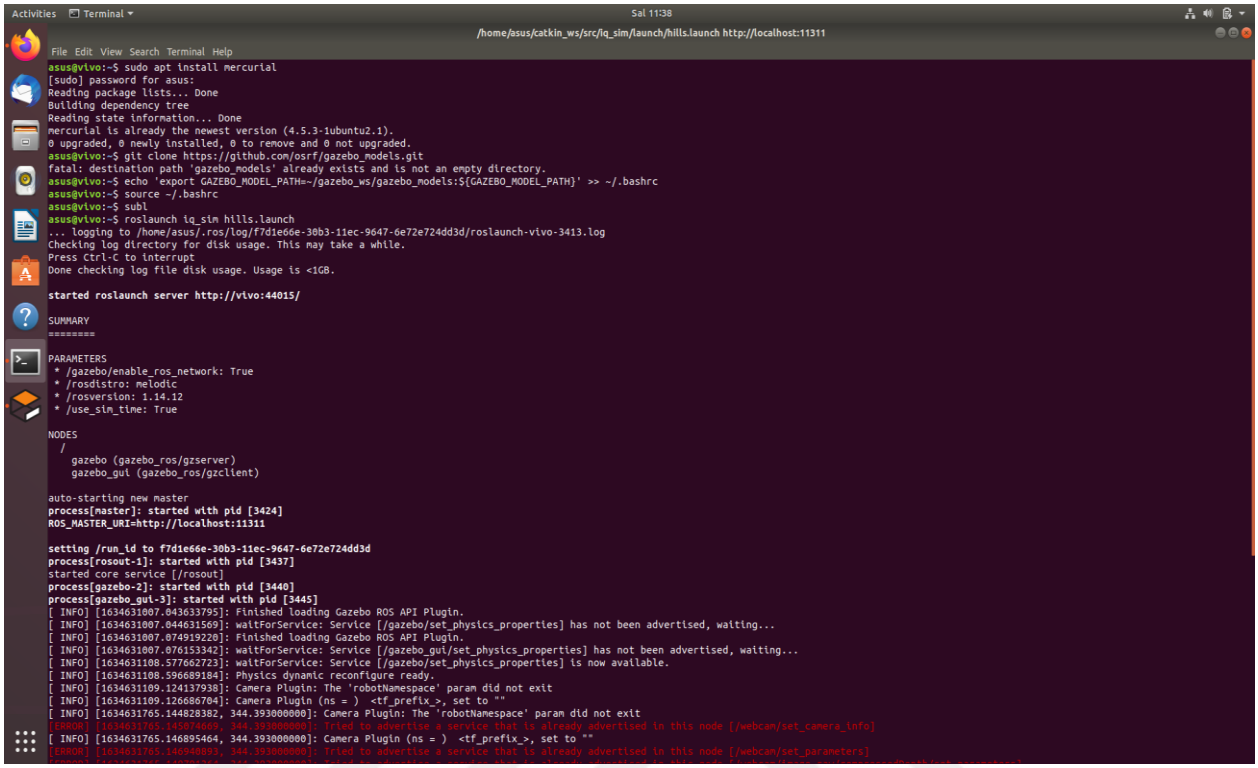
Şekil 3.7.3: Mode Guided ile Ros komutlarının çalışması

Gazebo Modelleri Veritabanına eklenmektedir. Açık Kaynak Robotik Vakfı'ndan (OSRF) bir dizi açık kaynaklı çardak modeli almak için git'i kullanabilmektedir.

```
git clone https://github.com/osrf/gazebo_models.git
```

Bashrc'ye modeller eklenmektedir.

```
Echo'export
GAZEBO_MODEL_PATH=~/gazebo_ws/gazebo_models:${GAZEBO_MODEL_PATH}'
>> ~/.bashrc
source ~/.bashrc
```



Şekil 3.7.4: Ros komutları ile Hills.launch dünyasının Gazebo uygulamasında açılışı

3.9. YENİ BİR DRON SİMÜLASYON DÜNYASI.

~/catkin_ws/src/iq_sim/worlds/ içinde hills.world adında yeni bir dosya oluşturup aşağıdaki satırları dünyaya kopyalayabilmektedir.

```
<?xml version="1.0" ?>
<sdf version="1.6">
 <world name="default">
  <physics type="ode">
   <ode>
    <solver>
     <type>quick</type>
     <iters>100</iters>
     <sor>1.0</sor>
    </solver>
    <constraints>
     <cfm>0.0</cfm>
     <erp>0.9</erp>
     <contact_max_correcting_vel>0.1</contact_max_correcting_vel>
     <contact_surface_layer>0.0</contact_surface_layer>
    </constraints>
   </ode>
   <real_time_update_rate>-1</real_time_update_rate>
   <!-- <max_step_size>0.0020</max_step_size> -->
  </physics>
  <include>
   <uri>model://sun</uri>
  </include>
  <include>
   <uri>model://ground_plane</uri>
  </include>
  <model name="iris">
   <include>
    <uri>model://iris_with_standoffs_demo</uri>
   </include>
   <!-- add new camera -->
   <link name='camera'>
    <pose>0 -0.01 0.070 .8 0 1.57</pose>
    <inertial>
     <pose>0 0 0 0 0 0</pose>
     <mass>0.1</mass>
     <inertia>
      <ixx>0.001</ixx>
      <ixy>0</ixy>
      <ixz>0</ixz>
      <iyy>0.001</iyy>
      <iyz>0</iyz>
      <izz>0.001</izz>
     </inertia>
```

```
      </inertial>
      <visual name='visual'>
        <pose>0 0 0 0 0 0</pose>
        <geometry>
          <cylinder>
            <radius>0.025</radius>
            <length>0.025</length>
          </cylinder>
        </geometry>
         <material>
          <script>
            <uri>file://media/materials/scripts/gazebo.material</uri>
            <name>Gazebo/Grey</name>
          </script>
        </material>
      </visual>
      <sensor name="camera" type="camera">
        <pose>0 0 0 -1.57 -1.57 0</pose>
        <camera>
          <horizontal_fov>1.0472</horizontal_fov>
          <image>
            <width>640</width>
            <height>480</height>
          </image>
          <clip>
            <near>0.05</near>
            <far>1000</far>
          </clip>
        </camera>
        <always_on>1</always_on>
        <update_rate>10</update_rate>
        <visualize>true</visualize>
       <!--  <plugin name="irlock" filename="libArduCopterIRLockPlugin.so">
            <fiducial>irlock_beacon_01</fiducial>
        </plugin> -->
        <plugin name="camera_controller" filename="libgazebo_ros_camera.so">
        <alwaysOn>true</alwaysOn>
        <updateRate>0.0</updateRate>
        <cameraName>webcam</cameraName>
        <imageTopicName>image_raw</imageTopicName>
        <cameraInfoTopicName>camera_info</cameraInfoTopicName>
        <frameName>camera_link</frameName>
        <hackBaseline>0.07</hackBaseline>
        <distortionK1>0.0</distortionK1>
        <distortionK2>0.0</distortionK2>
        <distortionK3>0.0</distortionK3>
```

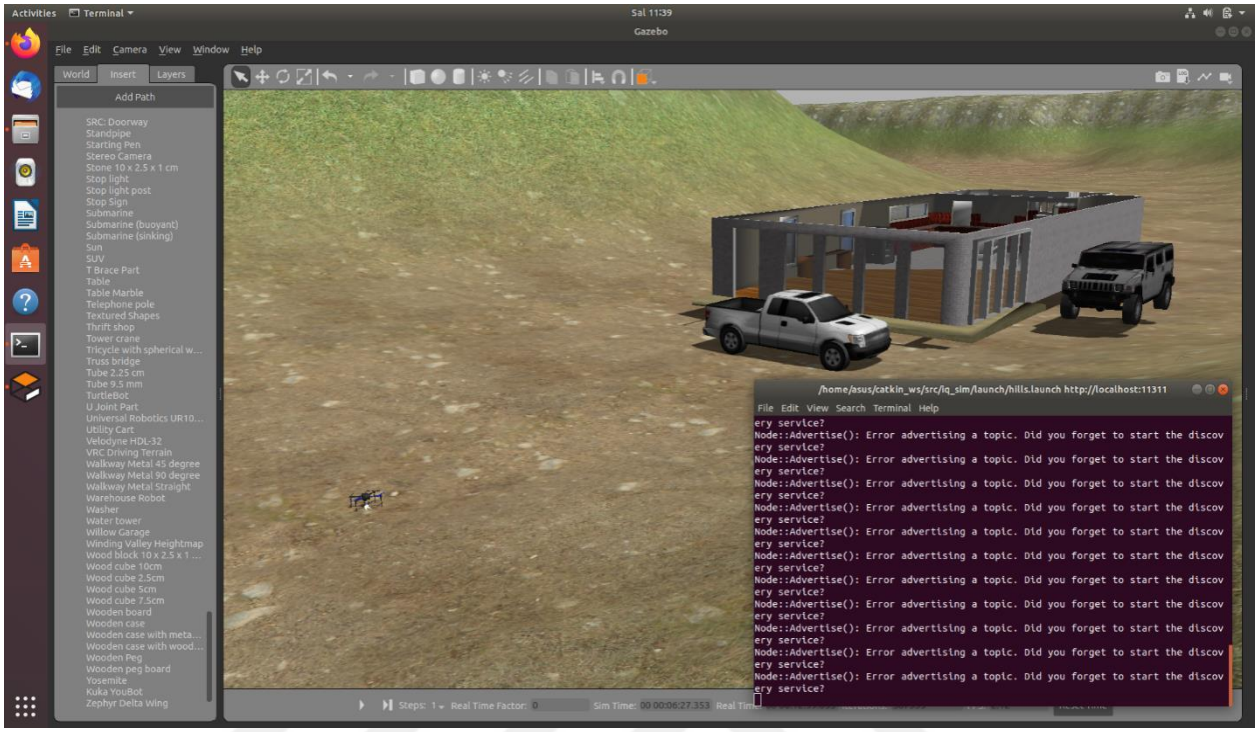
```
        <distortionT1>0.0</distortionT1>
        <distortionT2>0.0</distortionT2>
      </plugin>
      </sensor>
    </link>
    <!-- attach camera -->
    <joint type="revolute" name="base_camera_joint">
      <pose>0 0 0.0 0 0 0</pose>
      <parent>iris::iris_demo::gimbal_small_2d::tilt_link</parent>
      <child>camera</child>
      <axis>
        <limit>
          <lower>0</lower>
          <upper>0</upper>
        </limit>
        <xyz>0 0 1</xyz>
        <use_parent_model_frame>true</use_parent_model_frame>
      </axis>
    </joint>
  </model>
 </world>
</sdf>
```

Yukarıdaki kod sdf kodudur. Gazebo'ya özel bir biçimlendirme dilidir. Yukarıdaki satırlar, iyi bir gazebo dron simülasyon temeli oluşturmaktadır. Yukarıdaki fizik etiketi, dron'un uçuşunu en iyi şekilde simüle etmek için uyarlanmaktadır. Simülasyonun bir sonraki kısmı, dron'umuz olan iris. İris model etiketinin içindeki kod, kamera sensörümüz için gerekli olan koddur. Gelecekteki derslerde sdf kodlamasını gözden geçmektedir. ROS Eklentilerini Kolaylaştırmak için Başlatma Dosyası Ekleyip ~/catkin_ws/src/iq_sim/launch içinde hills.launch adında bir dosya oluşturup sonra aşağıdaki satırları eklenmektedir.

```
<launch>
  <!-- We resume the logic in empty_world.launch, changing only the name of the world to be
launched -->
  <include file="$(find gazebo_ros)/launch/empty_world.launch">
    <arg name="world_name" value="$(find iq_sim)/worlds/hills.world"/>
    <!-- more default parameters can be changed here -->
  </include>
</launch>
```

Yeni Dünya Başlatılmaktadır.

```
roslaunch iq_sim hills.launch
```



Şekil 3.7.1: Hills.launch dünyasına eklediğimiz nesnelerin modeli

3.10. GAZEBO ROBOTUNA SENSÖR EKLEMEK.

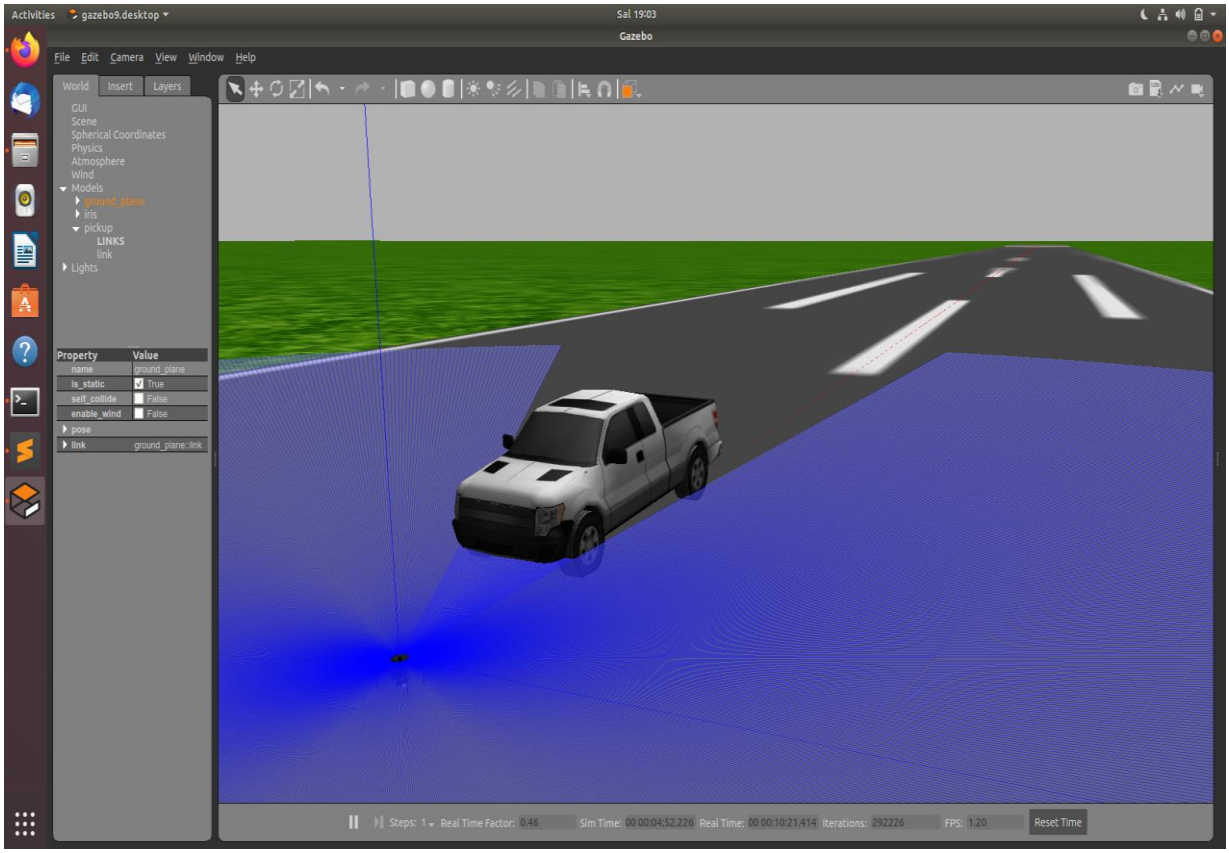
Dron Modelimize 2d LiDAR sensörü eklenebilmektedir. Bir modele sensör eklemenin 2 yolu vardır. Birincisi, bir modelin sdf dosyasını değiştirmektir. İkincisi, model etiketleriyle sensörü dünya dosyasının yanına eklemektir. İkinci yöntem, bugün üzerinde duracağımız şeydir. iq_sim/wolrds klasöründeki flye runway.world dosyasını açıp, sonra satıra ilerleyebilmektedir.

```
<model name="iris">
```

Bu hat, dron'umuzun belirtildiği yerdir. Dron'a zaten bağlı bir kamera görebilmektedir. 2d lidarımızı eklemek için benzer bir yöntem yapacağız. Daha sonra kamera için /joint etiketinin altına aşağıdaki satırları eklenmektedir.

```
    <link name="hokuyo_link">
      <pose>0 0 0 0 0 0</pose>
      <collision name="collision">
        <pose>0 0 0.3 0 0 0</pose>
        <geometry>
          <box>
            <size>0.1 0.1 0.1</size>
          </box>
        </geometry>
      </collision>
      <visual name="visual">
        <pose>0 0 0.27 0 0 0</pose>
        <geometry>
          <mesh>
            <uri>model://hokuyo/meshes/hokuyo.dae</uri>
          </mesh>
        </geometry>
      </visual>
      <inertial>
        <mass>0.016</mass>
        <inertia>
          <ixx>0.0001</ixx>
          <ixy>0</ixy>
          <ixz>0</ixz>
          <iyy>0.0001</iyy>
          <iyz>0</iyz>
          <izz>0.0001</izz>
          <!-- low intertia necessary to avoid not disturb the dron -->
        </inertia>
      </inertial>
      <sensor type="ray" name="laser">
        <pose>0 0 0.3 0 0 1.57</pose>
```

```
      <visualize>true</visualize>
      <update_rate>10</update_rate>
      <ray>
       <scan>
        <horizontal>
         <samples>1024</samples>
         <resolution>1</resolution>
         <min_angle>-3.141593</min_angle>
         <max_angle>3.141593</max_angle>
        </horizontal>
       </scan>
       <range>
        <min>0.1</min>
        <max>30</max>
        <resolution>0.1</resolution>
       </range>
       <!-- <noise>
        <type>Gaussian</type>
        <mean>0.0</mean>
        <stddev>0.01</stddev>
       </noise> -->
      </ray>
      <plugin name="hokuyo_node" filename="libgazebo_ros_laser.so">
       <robotNamespace></robotNamespace>
       <topicName>/spur/laser/scan</topicName>
       <frameName>/hokuyo_sensor_link</frameName>
      </plugin>
     </sensor>
    </link>
    <joint name="hokuyo_joint" type="fixed">
     <pose>0 0 0 0 0 0</pose>
     <parent>iris::iris_demo::iris::base_link</parent>
     <child>hokuyo_link</child>
    </joint>
```



Şekil 3.10.1: Gazebo ortamında dünyamıza eklediğimiz pikap nesnesine karşı Lidar Dron sensörünün algılama durumu

Resimde görüldüğü gibi Lidar sensörü dron'un etrafını taramaktadır, önüne gelen herhangi bir nesne algılandığı zaman lidar sensörü senaryoda tasarladığımız şekilde etrafındaki nesnenin bulunduğu konumdan belli bir mesafeye kaçmaya çalışır. 'Safelists' ekleyerek, engelden kaçınma programı oluşturmaktadır. iq_gnc/src içinde avoidance. cpp adlı bir dosya oluşturulmaktadır. Ardından aşağıdakileri iq_gnc CMakeLists'e eklenmektedir.

```
add_executable(avoidance src/avoidance.cpp)
target_link_libraries(avoidance ${catkin_LIBRARIES})
```

Bir Genaric C++ ROS düğümü kurulmaktadır.

```
#include <ros/ros.h>
#include <gnc_functions.hpp>
int main(int argc, char **argv)
{
        //initialize ros
        ros::init(argc, argv, "avoidance_node");
        ros::NodeHandle n;
        //add rest of code here
        return 0;
}
```

Önce kullanacağımız lidarın mesajı için include dosyası eklenmektedir.

```
#include <sensor_msgs/LaserScan.h>
```

daha sonra ana işlevimizde ros eklentisini şu şekilde tanımlanmaktadır.

```
ros::Subscriber collision_sub = n.subscribe<sensor_msgs::LaserScan> ("/spur/laser/scan", 1,
scan_cb);
```

Lidar verilerine erişmek için bir geri arama işlevi kullanmaktadır. Bunu içerir ve ana işlev arasına eklenmektedir.

```
Void scan_cb(const sensor_msgs::LaserScan::ConstPtr& msg)
{
}
```

Kalkış ve kontrol döngüsü eklenmektedir.

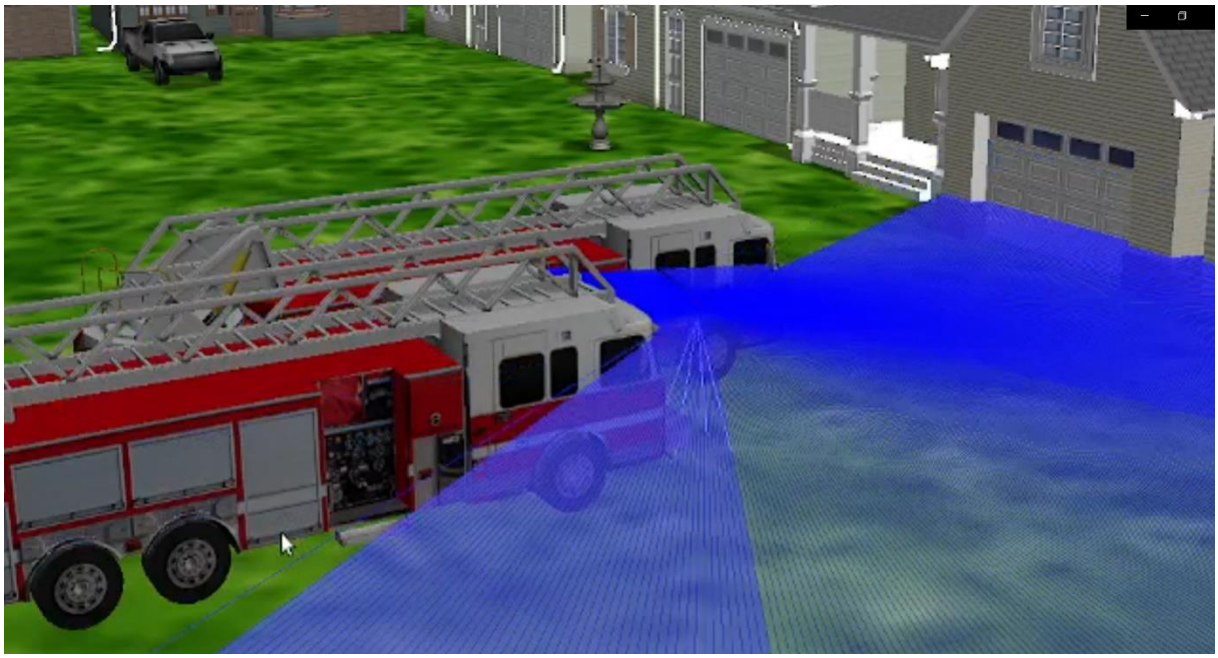
```
//initialize control publisher/subscribers
init_publisher_subscriber(n);
// wait for FCU connection
wait4connect();
//wait for user to switch to mode GUIDED
wait4start();
//create local reference frame
initialize_local_frame();
//request takeoff
takeoff(2);
set_destination(0,0,2,0);
ros::Rate rate(2.0);
int counter = 0;
while(ros::ok())
{
        ros::spinOnce();
        rate.sleep();
}
```

Bu, dronumuzu havalandıracak ve pozisyon tutmaktadır. Lidar verilerini ayrıştırmak için lidarın dönüşlerini inceleyeceğiz ve dron'nun manevra yapacağı bir yön ve büyüklük oluşturmaktadır. Burada görülen potansiyel alan yönteminin bir versiyonu kullanmaktadır.

```
sensor_msgs::LaserScan current_2D_scan;
current_2D_scan = *msg;
float avoidance_vector_x = 0;
float avoidance_vector_y = 0;
bool avoid = false;
for(int i=1; i<current_2D_scan.ranges.size(); i++)
{
        float d0 = 3;
        float k = 0.5;
        if(current_2D_scan.ranges[i] < d0 && current_2D_scan.ranges[i] > .35)
        {
                avoid = true;
                float x = cos(current_2D_scan.angle_increment*i);
                float y = sin(current_2D_scan.angle_increment*i);
                float U = -.5*k*pow(((1/current_2D_scan.ranges[i]) - (1/d0)), 2);
                avoidance_vector_x = avoidance_vector_x + x*U;
                avoidance_vector_y = avoidance_vector_y + y*U;
        }
}
```

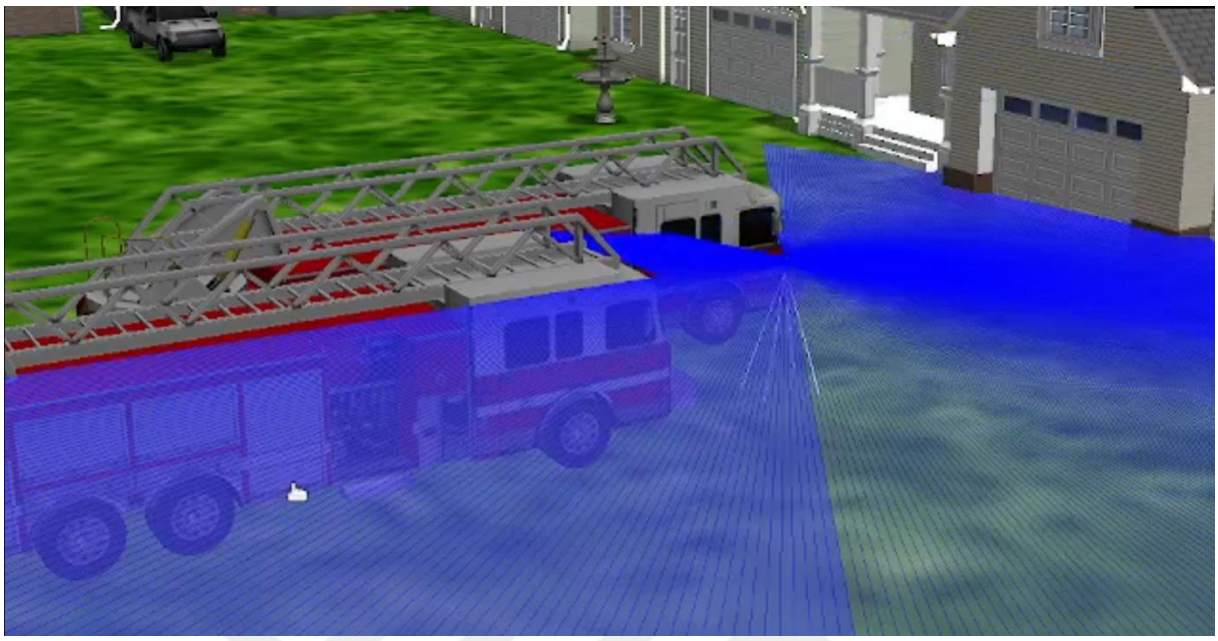
Aşağıdaki kod, doğru referans çerçevesinde kaçınma yol noktasını oluşturmakta ve ölçmektedir.

```
float current_heading = get_current_heading();
float deg2rad = (M_PI/180);
avoidance_vector_x = avoidance_vector_x*cos((current_heading)*deg2rad) -
avoidance_vector_y*sin((current_heading)*deg2rad);
avoidance_vector_y = avoidance_vector_x*sin((current_heading)*deg2rad) +
avoidance_vector_y*cos((current_heading)*deg2rad);
if(avoid)
{
        if( sqrt(pow(avoidance_vector_x,2) + pow(avoidance_vector_y,2)) > 3)
        {
                avoidance_vector_x = 3 *
(avoidance_vector_x/sqrt(pow(avoidance_vector_x,2) + pow(avoidance_vector_y,2)));
                avoidance_vector_y = 3 *
(avoidance_vector_y/sqrt(pow(avoidance_vector_x,2) + pow(avoidance_vector_y,2)));
        }
        geometry_msgs::Point current_pos;
        current_pos = get_current_location();
        set_destination(avoidance_vector_x + current_pos.x, avoidance_vector_y +
current_pos.y, 2, 0);
}
```



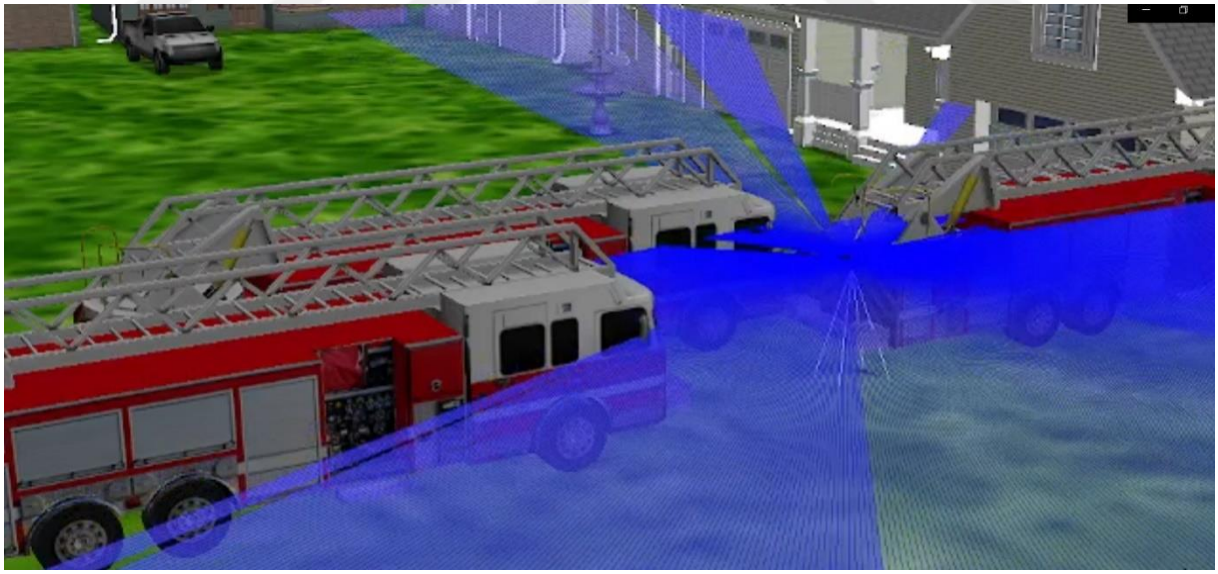
Şekil 3.10.2: Kaçınma yol noktasını oluşturup ölçmesi

Resimde Lidar sensörü ile donatılmış dron'umuza, 2 tane yakın mesafede itfaiye arabası eklenmektedir, eklenen itfaiye arabaları Lidar algıladığı zaman, otomatik şekilde belli bir uzaklığa kaçmaktadır.



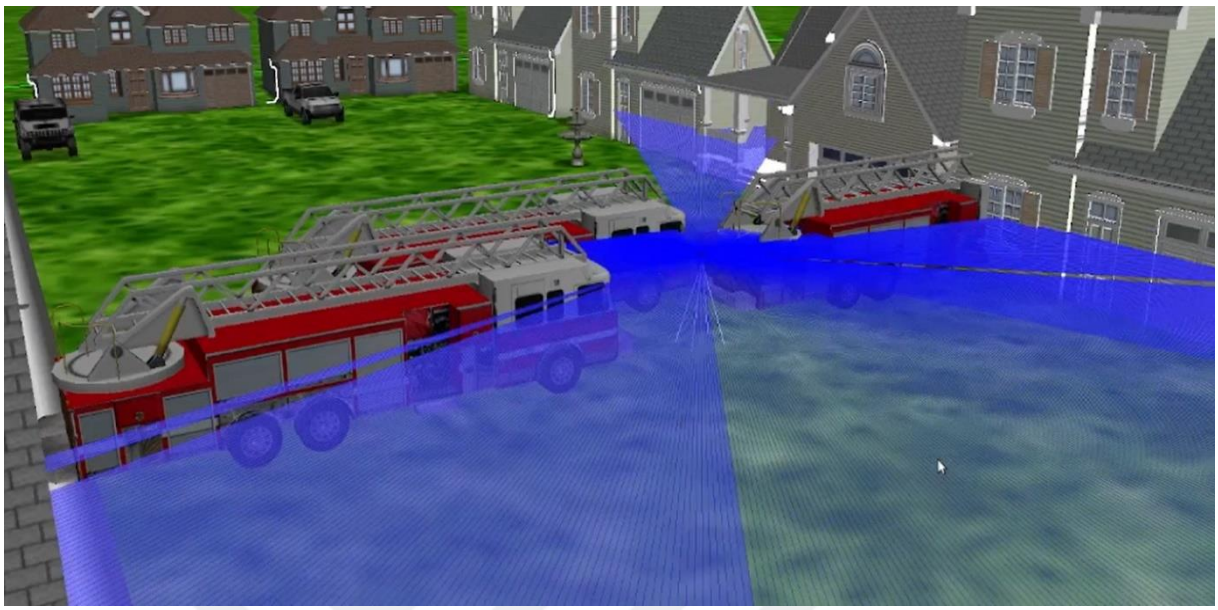
Şekil 3.10.3: Konum değiştirmesi

Yukarıdaki resimde dron'umuz bir nesneyi tespit ederek eski konumdan belli bir mesafeye hereket etmektedir.



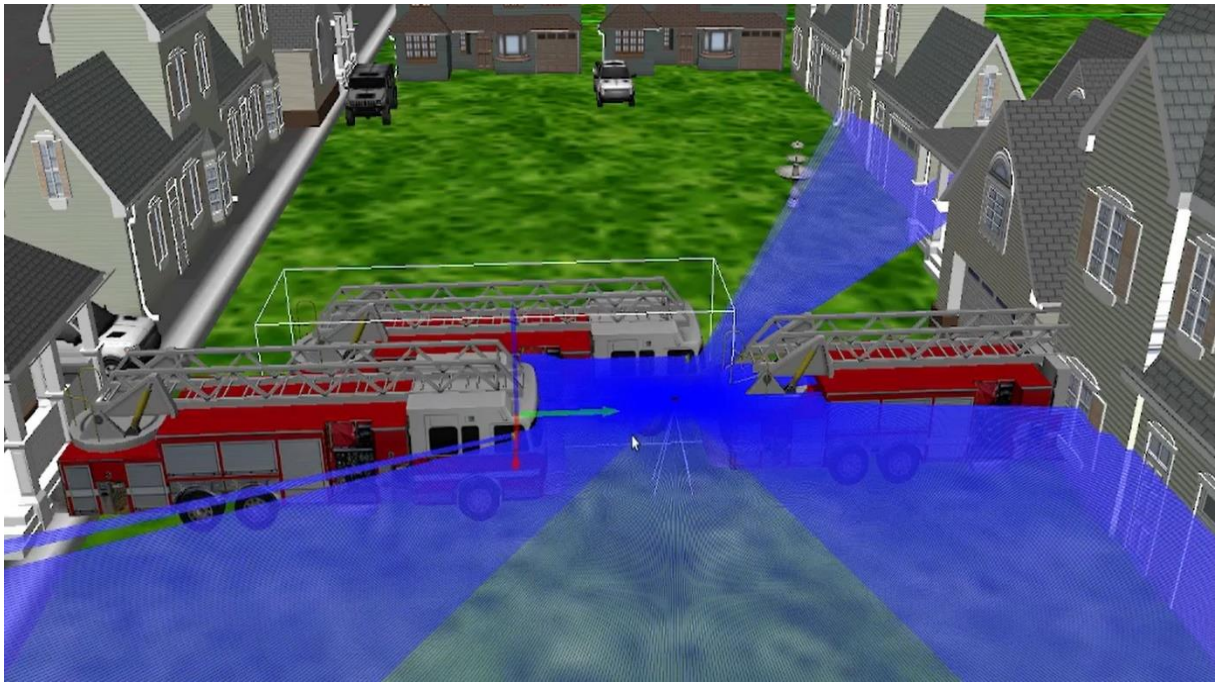
Şekil 3.10.4: Gazebo dünyamıza ek olarak engel eklemek

Gazebo dünyamıza ek olarak üçüncü itfaiye arabasını eklemek durumundayız, aynı şekilde dron lidar sensörü sayesinde yeni bir nesnenin etrafında bulunduğunu algıladığı görünmektedir.



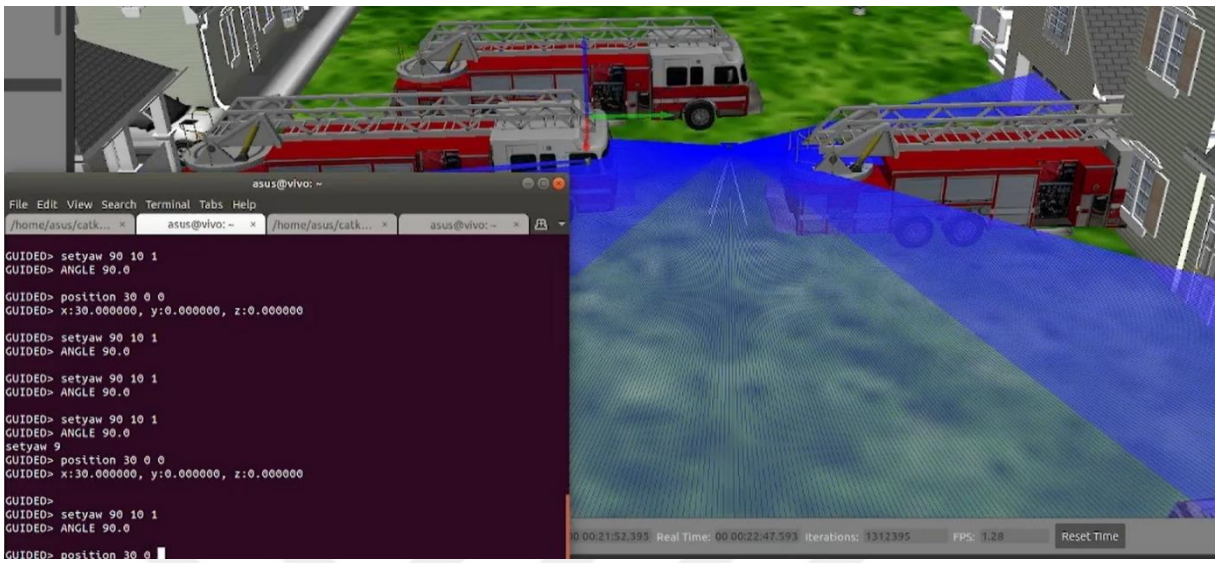
Şekil 3.10.5: Dron'un etrafındaki engellerden kaçınması

Yeni durum için ön, sol ve sağ taraftan eklenmiş olan nesneleri güzel bir şekilde algılayan lidar sensörü, tasarlanmış olan kod üzerinden otomatik belli bir mesafeye kaçtığı görülmektedir.



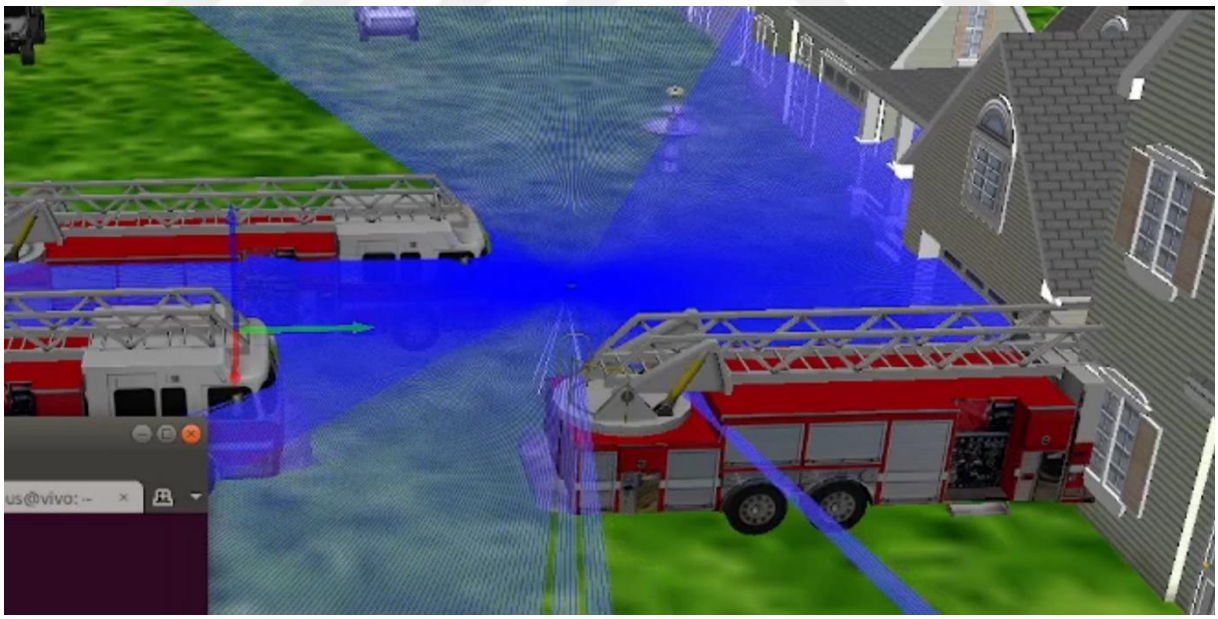
Şekil 3.10.6: Dron'un etrafındaki engellerden kaçınması

Dronun önündeki itfaiye arabasını ileriye sürüklemeyerek konumunu değiştirmektedir.



Şekil 3.10.7: Dron'un 30 metre ileriye ilerlemesi

Position(Konum) x y z komutu ayarlamasıyla ileride bulunan itfaiye arabasına x ekseni üzerinden Konum 30 0 0 komutu ile dronumuzu 30 metre ileriye yürütmektedir.



Şekil 3.10.8: Dron'un engelden kaçınması

İleriye x ekseni üzerinden giden dron önündeki engelden kaçınarak sağ tarafta müsait alanın olduğunu algılayıp konumunu o yöne doğru değiştirmektedir.

4. BULGULAR

4.1. SON-ADIM TESLIMAT VE DRON KULLANIMI

Yaşadığımız dönemde kırsal alanlardan kente doğru bir göç hareketi mevcuttur. Bunun neticesinde, değişen tüketici davranışlarına artan odaklanma, son adım lojistiğinin ehemmiyetini de arttırmaktadır. Artık son-adım lüksten çok bir beklenti haline gelmiştir. BM, 2050 yılına kadar insanların %60'inin şehirlerde yaşayacağını tahmin etmektedir (Shaikh. K, 2017). Bundan dolayı şehirlerde normalin üstünde bir yaşam kalitesinin olması ve problemlerin ortadan kaldırılması önemlidir. Son dönemde şehirlerde araç trafiğinin fazla olması ve popülasyonun artması dron'ların daha hızlı ve güvenilir olmasını gerekli hale getirmiştir. Dron kullanımı acil teslimatlardı normal araçlara göre tercih edebilmektedir. Önemi, fazla maliyeti ve karmaşıklığı nedeniyle nakliyenin son aşamasında bilgi işlem teknolojileri, Endüstri 4.0 ve yeni nakliye araçlardan söz edebilmektedir. Teknolojinin ilerlemesi ile son-adım teslimatta dron'lar sorunları çözmek için önemli bir araç olarak görülmektedir.

4.2. DRON AVANTAJLARI

Dron'lar diğer nakliye taşıtlarıyla kıyaslandığında yakın aralıkta daha az maliyet, ulaşımda kolaylık ve insana ihtiyaç duymaması açısından avantajları vardır. Taşıma araçlarında bir yol üzerinde gitme zorunluluğu mevcutken dron ise istediği yön ve yere doğru hamle yapabilir ve ulaşabilir. Bundan dolayı başlangıç ve ulaşma noktası arasındaki mesafe kısalarak hem zamandan hem de harcanacak efordan tasarruf edilmiş olur. Trafik ve kalabalık içine girmemesi de bir artı olarak söylenebilir. Müşteri açısından dron kullanımının güvenlik ayağı oldukça önemlidir. Ayrıca GPS özelliğiyle dron takibi kolayca yapılabilmektedir.

4.2.1. Dron Dezavantajları

Garcia K. ve Santoso J (2019), dron taşımacılığının zorluklarını üç ana alanda ele alıyor:

Hukuki Hususlar: Mesela ABD yönetmeliklerine göre hükumet kurumları ve kişiler üzerinden uçamazlar, havalimanlarının yakınında uçabilirler, gece uçamazlar, 120 metrenin altında uçmaları gerekir, 160 km/s'yi geçmemelidir.

Dron yük taşıma kapasitesi 5-7 kg arasındadır, taşıma mesafesi ise: 15-20 km ile sınırlıdır. Ayrıca dron'ların batarya teknolojisi kısıtlı olduğundan çok uzun bir mesafe taşıma imkânı yoktur. Taşıma araçlarına göre daha az ağırlık taşıyabilmesi ve sadece hafif malzemeleri yüklenebilmesi de bir dezavantaj olarak söylenebilir. Dron çoğu konuda tercih edilse de güvenlik, gizlilik ve ülkelerin yasal düzenlemelerine tabi tutulduğu için bazı kısıtlamalar dron kullanımında kısıtlama yaratabilmektedir.

4.3. DRON'UN UÇUŞ SÜRESINI HESAPLAMAK.

Dron için Lipo Pil Seçimi ve Uçuş Süresi hesaplama aşamasında Lipo Pil, Hücre Sayısı 2S, 3S, 4S tercih etmemize karar vermemiz gerekmektedir.

İkinci aşamada Lipo pilimizin ağırlığını belirlememiz gerekmektedir. Hangi aralıkta lipo pil bizim multikopterimize için uygun veya uygun değil bunu belirlememiz gerekmektedir.

Üçüncü sırada Lipo pilimizin depo değeri olan mAh gelmektedir. En son C değerini belirlemektedir.

İlk aşamada Hücre sayısını belirlememiz için motorumuzun ve ESC'mizin kaç S lipo pil desteklediği ve bu lipo pillerdeki verimlilik değerlerini bakmamız gerekmektedir. Hücre sayısını belirlememiz bizim diğer değişkenlerimize yetkilememektedir.

Ağırlığı belirlememiz için itki küvvetine ve dronun toplam ağırlığına ihtiyacımız var. Burda Lipo pilin kaç gram olması gerektiğini bir aralık olarak belirledikten sonra mAh geçe bilmektedir, çünkü mAh değerini belirleyebilmemiz için ilk gereken değişkenimiz ağırlık. Dolaysıyla ağırlıkta yapacağımız hesap mAh yetkilemektedir. mAh ağırlıkla beraber belirledikten sonra Hover(Uçuş) süresini belirlememiz gerekiyor. Hover dronun havada askıda kaldığı anı ifade ediyor. Uçuş süresini hesaplamak zor, uçuk kişiden kişiye göre değişir (Manevralar, hızlı kullanmak, eğim açısını belirlemek). Seçtiğimiz mAh değeri ve sistemimizin ihtiyaç duyduğu akımı hesaplayarak C değerimizi belirlememiz gerekmektedir.

Pervane için ML2212 motorun kullanmaktadır. ML2212 motoru 920KV, 230 watt çıkışlı fırçasız bir motordur. 709 ile 1550 gram ağırlığındaki özellikle spor uçaklar için kullanmaktadır.

Tablo 4.3-1: ML2212 920KV motorun teknik veri tablosu

ML2212 MOTOR								
ltem NO.	Volts (V)	Prop	Throttle	Amps (A)	Watts (W)	Thrust (g)	Efficiency (g/W)	Operating temperature(℃)
ML2212 920KV	11.1V	DJI9.4*4.3	50%	1.8	20.0	230	11.5	37℃
			65%	2.8	31.1	310	10.0	
			75%	3.9	43.3	410	9.5	
			85%	5.5	61.1	480	7.9	
			100%	7.6	84.4	610	7.2	
		APC10*4.5	50%	2.6	28.9	290	10.0	55℃
			65%	5.1	56.6	460	8.1	
			75%	7.4	82.1	590	7.2	
			85%	10.1	112.1	730	6.5	
			100%	13.4	148.7	860	5.8	
	14.8V	DJI9.4*4.3	50%	2.7	40.0	350	8.8	52℃
			65%	4.4	65.1	490	7.5	
			75%	6.3	93.2	640	6.9	
			85%	8.3	122.8	790	6.4	
			100%	11.5	170.2	990	5.8	

Karşılaştırma yaparsak tablodan APC10*4,5 tam yük altında 860 gram kaldırma küvveti uygularken DJI9,4*4,3 lipo pil yine tam yük altında 990 gram kaldırma küvvetini uygulamaktadır.

Bu deneyde kullandığımız fırçasız motor yüksek miktarda akıma ihtiyaç duyduğundan 5000mAh 11.1V 3 hücreli Li-Po (Lityum Polimer) pil kullanmaktadır. Sürekli olarak yaklaşık 3A akım sağlayabilmektedir.

Şekil 4.3.1: 5000mAh 3 hücreli Li-Po pil

Şimdi hücre seçimindeki ikinci kriteri ESC 2 ve 3S lipo piline destek vermektedir. Pil ağırlığını hesaplamamız için motorların toplam gücü (itki küvveti) = Toplam kalkış ağırlığı x 2

ML 2212 920KV dronun teknik tablosundan 3S lipo pil, 10*4,5 pervane için tam yük altında 1 motorumuz 860 gram kaldırma küvvetini uygulaya bilmektedir.

860 gram ve toplam 4 tane motorumuz olduğu için toplam sistemimizin kaldırma küvveti 3440 gram olarak çıkmaktadır. Toplam ağırlık Pil ve Dronun ağırlığından oluşur. Bizim durumumuzdaki dronun ağırlığı 1300 gram.

3440 gram = (Pil + 1300 gram)*2

Pil = 420 gram

Bu pil değerini faydalanarak mAh hesabını yapabilmektedir.

Elimizdeki 3S lipo pil ve 420 gram pil ağırlığı bilgisini internetten araştırarak bu aralıkta olan lipo pillerimizi baktığımızda 6000mAh var olduğunu gördük. Sürekli olarak 25C üzerinden akım sağlıyor. Bu değerleri faydalanarak Hover süresini hesaplamaktadır.

Hover Süresi = Pilin 1 dakikada verebildiği akım / 4 motorun toplamda talep ettiği akım.

Bu durumda Pil 6000mAh

1mA = 1A denk geldiği için 6000mAh = 6A

1 saat boyunca 6A akım suna bildiğine göre 6Ax60=360, yani 1 dk boyunca 360A akım suna bildiğini söyleyebiliriz. 360 A akımın sadece %80 kullanacağımızı düşünerek depomuzun toplamda 1 dk boyunca 288A/dk suna bileceğini sonucunu çıkarmaktadır.

360 * %80 = 288A/dk

İlk aşamamız tamamlandı, şimdi motorların talep ettiği akımı hesaplamamız için, ilk olarak toplam kalkış ağırlığını hesaplamamız gerekiyor. Dronumuz 1300gram, kullandığımız 6000 lipo pil 468grama denk geliyor dolayısıyla bizim toplam kalkış ağırlığı 1768 gram.

1300 + 468 = 1768 gram

1768 gram toplamda 4 motor kaldıracağı için motor başına düşen yükü hesaplamak için bu değeri 4'e bölüyorum ve motor başına 442 gramlık bir yük kaldığı ortaya çıkmaktadır.

$$1768 / 4 = 442 \text{ gram}$$

Bizim motorumuzun dengede kalması için uygulaması gereken kaldırma küvveti 442 gram anlamına gelmektedir. Teknik tabloya bakarak 442 gram kaldurma küvvetine 1 motorun uygulaması için 5A ihtiyacımız var. Bu 5A toplam çeken 4 tane motorumuz olduğu için

$$5A x 4 = 20A$$

yani bizim motorlarımızın anlık talep ettiği akım 20A olarak karşımıza çıkmaktadır.

$$288/20=14.4dk$$

yaklaşık olarak hover süresini tanıyacağını anlaya bilmektedir.

Lipo pilin C değeri bizim sistemdeki ihtiyacını karşılıyor mu bunu hesaplayacağım. Bunun için sistemin ihtiyaç duyduğu akımı hesaplamamız gerekyor.Bunun için teknik veri tablosuna bakarak 11V, 10*4,5 pervane tam yük altında 13,4 A akım çektiğini görüyoruz. Bu 1 motorun çektığı akım, ve bunu 4e çarptığımızda bizim sistemimizin ihtiyaç duyduğu akım 53,6 A olarak karşımıza çıkmaktadır.

$$13,4 A x 4 = 53.6A$$

Şimdi Lipo pilimizin bize bu akım sağlayıp sağlayamadığını hesaplamamız gerekyor. Lipo pilimiz 6000 mAh = 6Ah , lipo pilin verdiğı anlık akımı 1 saat boyunca vere bileceğı akımla seçtığımız lipo pilin C değerini çarparak bula bilmektedir.

$$6A x 25 = 150A$$

Burdan çıkan sonuç lipo pilin anlık olarak 150A akım sunabileceğini bize göstermektedir.

5. SONUÇ VE TARTIŞMA

Günümüzde en yaygın teknolojisi olan İHA'lar artık kargo taşımacılığı da yapabilmektedir. Son zamanlarda nakliyenin ehemmiyetini ele alarak verimli, sağlam, seri ve düşük maliyetlerle en ehemmiyeti de ortama hassas bir şekilde kargo İHA'lar için hizmete başlamaktadır. Nakliyenin en temel maksadı en az vakit içerisinde ve en düşük maliyetle gerçekleştirmek olmasıyla küreselleşen yeryüzünde teknolojinin artması ile ulaştırma sorununun giderek artması ve bu konuda yer alan aksilikleri engellemek maksadıyla kargo İHA'lar ehemmiyet kazanmaktadır. Kargo İHA'lar nedeniyle siparişleri şehrin bir tarafından öbür tarafına kolayca gönderebilmektedir.

Bu çalışmada MavProxy yazılımını destekleyen Ardupilot ortamında ve Gazebo simülatöründe kullanılan Lidar dron önüne çıktığı nesneleri kurduğumuz algoritma ile tespit ederek engelden kaçınmaktadır. Gazebo simülatörü kapalı ve açık mekânlar için tasarlanan, birden fazla robotların 3 boyutlu ortamda test yapabildiği bir programdır. Dron'un kontrolünü Robot İşletim Sistemi ile gerçekleştirmektedir. ROS sistemi ile kontrol edilen robotta algılayıcı sayesinde seslenme, görüntüleme benzeri dışarıdan eklenen veriler ROS ile bilgisayara ulaşmaktadır. Bilgisayarda algoritma ve kullanım alanına göre yapılan veri alıcı konumunda robota komut olarak geri dönebilmektedir.

Tez çalışmamızda kullanılan yazılım araçları Ardulilot MavProxy ROS ve SITL test için kullanılan dron'umuzu sanal ortamda denemek için maliyet açısından ve gerçek hayata tatbik edebilmemiz için büyük katkı sağlamaktadır. Her türlü büyük çalışmalar ilk önce teori öğrenilip sonra pratiğe döktüğümüz gibi bu çalışmada da yapmak istediğimiz amacımızı ilk önce test ortamında yani simülasiyon ortamında deneyerek, başarısız sonuçları simülasyonda görerek düzeltme yapabilme maliyet açısından çok fayda vermektedir. Dronumuz yük taşıma kapasitesi 5-7 kg arasındadır, taşıma mesafesi ise: 15-20 km ile sınırlıdır. Dron'un kaldırma kapasitesi ve uçma mesafesi seçtiğimiz motora ve pilimize bağlıdır.

Sonuç olarak kullandığımız Gazebo programında farklı alan modelleyerek test dron'umuzun 3 boyutta etrafına farklı açılarından engel nesneleri koyarak kurduğumuz engelden kaçınma algoritmasın %90 başarılı bir şekilde hedefe ulaşıldığı görülmektedir.

Bu tez çalışmasında yapılan simülasyon çalışması için hava koşulları (yağmurlu, kar, vb) durumunda dikkate alınmamıştır. Geleceğe yeterli çalışmalar olarak dron simülasyonu için bu koşullarda dikkate alarak yeniden değerlendirilmesi yapılabilir. Yapılan yeni denemelere göre hedefe varma başarı oranı test edilebilir.

KAYNAKLAR

Ahmet. K (2018). The Importance and Future of Drone in Cargo Transportation.

M. Castillo-Lopez; S. Jose; M. Miguel. (2018). Model Predictive Control for Aerial Collision Avoidance in dynamic environments.

E. Devos, E. Ebeid, P. Manoonpong (2018). Development of Autonomous Drones for Adaptive Obstacle Avoidance in Real World Environments.

Jesus. T, Jonathan. P. (2021). PANTHER: Perception-Aware Trajectory Planner in Dynamic Environments.

S. Kawabata, J. Hoon Lee, S. Okamoto, (2019). Obstacle Avoidance Navigation Using Horizontal Movement for a Drone Flying in Indoor Environment.

R.Azim Bappy, S.Islam, A.Sajjad, N.Imran. (2017). Design and Development of Unmanned Aerial Vehicle (Drone) for Civil Applications.

D.Valencia, D.Kim, (2018). Quadrotor Obstacle Detection and Avoidance System Using a Monocular Camera.

Kellner, Madachy, Raffo (1998).

Keller. R. (2019).

Hürriyet Haber. (2017).

Jeff. B. (2019).

ÖZGEÇMİŞ

Kişisel Bilgiler	
Adı Soyadı	Nauryzkhan Derbishev
Doğum Yeri	
Doğum Tarihi	
Uyruğu	❒ T.C. ☑ Diğer:
Telefon	
E-Posta Adresi	
Web Adresi	

Eğitim Bilgileri	
Lisans	
Üniversite	L.N.Gumilyov Eurasian National University
Fakülte	Faculty of İnformasion System
Bölümü	Automation and Control
Mezuniyet Yılı	2017

Yüksek Lisans	
Üniversite	İstanbul Üniversitesi
Enstitü Adı	Ulaştırma ve Lojistik
Anabilim Dalı	Akıllı Ulaşım Sistemleri Anabilim Dalı
Programı	Akıllı Ulaşım Sistemler Programı