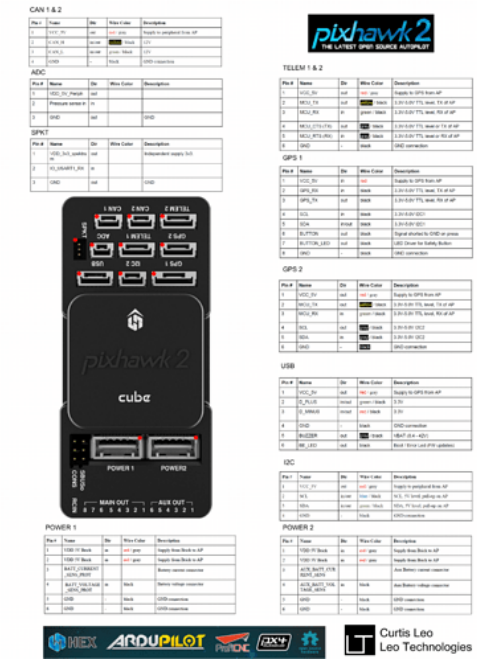
Yoda İHA Yazılım Ödevi 1 **Meryem KAYA - 24014116**

Savaşan İHA kategorsindeki NEÜ Kapsül Göktürk Ekibi , ALGAN UAV ve Hüma Takımlarının KTRlerini inceleyerek başladım.

Uçuş Kontrol Kartları

Uçuşun gerektirdiği kararlılık için önemli olan geri besleme sinyallerini üretmek, uçuş mekaniğinin ve görevinin gerektirdiği yazılımı barındırmak, Savaşan İHA' nın bileşenleri arasında iletişimi sağlama ve uçuş için gerekli sensör verilerini toplayıp anlamlı hale getirilmesini sağlamaktadır.İncelemiş olduğum raporlarda kullanılan ve karşılaştırılması yapılan kartlar Pixhawk Cube 2.1 Orange, Pixhawk 2.4.8 ve Pixhawk Cube Black ön plandaydı. Black Cube tercihinde IMU (her iki sensörden gelen verileri birleştirerek CPU'nun (merkezi işlemci) konum, hız ve tutum gibi 3 boyutlu uzaydaki yönelim bilgilerini ve hareket yönünü hesaplamasına olanak tanır) adedine göre gyro, ivmeölçer ve pusula verilerini yüksek hassayitte ölçmesi ve taşıyıcı sayesinde I2C, UART, SBUS giriş çıkış birimleri bulunması ve harici sensörler ile haberleşmesine dikkat edilmiştir. Cube Orange da ise dahili 3 manyometre(manyetik alanı veya manyetik dipol momentini ölçen bir cihazdır), 3 ivmeölçer, 2 barometre ve 3 jiroskop gelişmiş sensörlere sahip olması, PPM, SBUS, UART vb. sinyal giriş çıkış yerlerine sahip olması ,diğer Cube serilerine kıyasla Dahili ADS-B Anteni barındırması, (ADS-B havada bulunan uçakların konumlarını çok daha kesin olarak görülebildiği bir iletişim yöntemidir.) Kullanılan kart yazılımı ile uyumlu çalışması göz önünde bulundurulmuştur.



Uçuş bilgisayarı türleri Raspberry Pi 4B, NVIDIA Jetson Nano

kameradan alınan görüntüyü, görüntü işleme algoritmaları ile çalıştıracaktır ve haberleşme sistemi ile veri akışı sağlayacaktır. Bu işlemleri performanslı yapabilmesi nedeniyle Nvidia Jetson Nano yardımcı bilgisayarı seçilmiştir. Jetson Nano 128 Cuda çekirdeği ve 4 GB RAM hızına sahiptir. Jatson Nano ile

birlikte kullanılacak kamera FPS değerini etkilemektedir. Bu nedenle Jatson Nano ile uyumlu IMX219-77 kamera kullanılmıştır.

Uçuş Bilgisayarı: Belirlenen görevde İHA'da kullanılacak olan otonom kontrol algoritmasının tam performanslı ve kullanılması planlanan güç dağıtım kartı ile uyumlu çalışması, 125Mbps Ethernet bağlantı hızı ile yer istasyonuna daha hızlı veri ve görüntü aktarılması istendiği için Raspberry Pi 4B tercih edilecektir.

Otopilot Yazılımı

Pixhawk ile uyumlu çalışan PX4 oto pilot yazılımı seçilmiştir. PX4 seçim kriterleri: Açık kaynaklı bir yazılıma sahip olması sayesinde uçuş modu eklenebilmesi, Yaygın olarak kullandığından döküman bilgisinin fazla olması, Modunu Auto-tune modu sayesinde rahatlıkla ayarlanabilmesi, Fail-safe uçuş moduna sahip olmasıdır.

İHA'da bulunacak kontrol yazılımı dronekit kütüphanesi aracılığıyla yazılmıştır. Bu kütüphanenin özelliklerinden biri ise İHA'nın kontrol yüzeylerine RC Channellar ile komut verilebildiğinden dolayı zorlanılmadan istenilen manevraların yapılabilmesidir. Bu özelliğe ArduPilot yazılımının sahip olmasından dolayı simülasyon testlerinde ArduPilot yazılımının kullanımına karar verilmiştir. ROS – Gazebo – Ardupilot arasındaki haberleşme ise Ardupilot dökümantasyonunda belirtilen 'plugin' ile gerçekleştirilir . Simülasyon testleri için hazır model olarak simülasyon ortamına eklenen İHA'lardan biri ALGAN İHA Takımına ait olmasıyla birlikte diğer iki İHA ise rakip olarak belirlenmiştir.. Hazırlanan simülasyonda farklı senaryolar denenerek otonom uçuş algoritması mümkün olduğunca stabil hale getirilmeye çalışılmıştır. Otonom Uçuş: Yarışma sırasında görevlerin gerçekleştirilebilmesi için İHA'nın stabil uçması gerekmektedir. Stabil uçuş için ise simülasyon ortamı üzerinde İHA üzerinde 'autotune' modu aktifleştirilerek uygun PİD ayarları yapılmıştır. Sonraki aşamada dronekit kütüphanesi aracılığıyla yazılan Python yazılımları ile gerekli testler gerçekleştirilmiştir.

Nesne Takibi ve Tespit Algoritmaları

Otonom kilitlenme görevi için rakip aracın görüntüsünün kamera görüntü işleme algoritmasıyla tanınıp ona göre aracın yönlendirilmesini gerektirir. Bu görevde takibin aralıksız devam etmesi için algoritmanın gerçek zamanlıya yakın bir hızda çalışması gerekir. Algoritma seçiminde belirtilen hızı sağlayabilen 2 tane ulaşılabilir yazılımı vardır. Bunlardan biri SSD diğeri ise YOLO'dur. SSD algoritması sadece convolutional neural network kullanırken YoloV5s algoritması görüntünün içindeki potansiyel objeyi edge detection yöntemiyle önceden tespit edip sonrasında CNN modelini uygular. YoloV5s'in yaptığı bu optimizasyon ona büyük avantaj kazandırır. Bu yöntemin getirdiği en büyük dezavantaj görüntüdeki bölünmüş her karenin işlenmemesidir. Bu YoloV5 algoritmasının isabetlilik olarak geride kalmasına neden olur ve küçük ve çoklu cisimlerin algılanamamasına neden olur. Fakat bu dezavantaj yarışmanın gereklilikleri göz önüne alındığında pek bir önem arz etmemektedir. Yarışmada bizden tespit edilmesi istenen hedefler rakip İHA'lar ve önceden belirlenmiş QR kodlar; bu hedefler algoritmalar için ayırt edilmesi çok zor hedefler değiller ve sadece tek bir sınıf için eğitim yapılması gerekir.

YoloV5 Algoritması

YoloV5s'in üzerine kurulduğu framework olan PyTorch; bu alanda önde gelen bir yazılımdır. Yapay zekâ adına yapılan bilimsel çalışmaların salt çoğunluğunda kullanılması bunun bir sonucudur. PyTorch temelde C++ tabanlı olması sebebiyle yapılan çalışmalar C++'a hızlıca aktarılabilir. Aynı zamanda yüzeyde ise python kullanması sebebiyle yazılan projeler her zaman daha kolay anlaşılabilirdir. Bu özellikler yapay zekâ gibi karmaşık alanlarda fazlasıyla işe yarar.

Algoritmanın eğitilmesi için gerekli veri seti Kaggle'dan alındı. Bu veri seti Roboflow'da etiketlendi. Roboflow'da bu hazırladığımız veri setiyle bize gerekli train validate ve test setlerinin hazırladı. Bu aşamadan sonra Eğitim aşaması fazla uzun ve donanım açısından pahalı olduğu için Google Colab'de eğitim algoritmanın eğitimini yaptık. Bu aşamada elimizde rakip İHA'yı tanımak için gerekli yazılım sağlanmış oldu. Bundan sonraki aşamalarda bu hedefe yönelme sağlanacaktır.(**bu kısımı yeterince araştırmadığım için birçok kavramı algılayamadım.**)

Edge Detection

Edge detection agloritması, görüntü üzerindeki piksellerin renk değerlerinin birbirinden farklılaşasından yola çıkarak görüntü üzerindeki kenarları belirler.

CNN (Convolutional Neural Network)

Bu aşamadan sonra YoloV5'in şemasında da görüldüğü gibi CNN aşmasına girilir. Convolutional neural network, görüntüdeki kareleri boyutları önceden kararlaştırılmış bir çerçeve içine alır ve bu çerçeve içindeki verileri modelde işleyip tek bir veri halinde sıkıştırır. Bu sıkıştırma işlemi bittikten sonra ise önceden belirlenmiş adım sayısı kadar kareyi atlar ve oraya denk gelen çerçevedeki karelere aynı sıkıştırma işlemini uygular. Bu küçültme işlemi belirtilen sayıda tekrarlanır. Bunlardan elde edilen veriyle nihai tahmine varılır.

Non Max Supression

Bu aşamanın sonunda YoloV5 yazılımının içinde Non Max Supression adı altında geçen fonksiyon çağırılır bu fonksiyon aşağıda görülene benzer hataları düzeltir. Bu hatalar CNN'in çalışma doğası gereği birden fazla karenin birbirinin üstüne binerek aynı cismin birden çok kez tanınmasına sebep olabilir. Bu fonksiyon bunun önüne geçer. Bir diğer görevi ise eğer tahminde güven oranı yeterince yüksek değilse bu etiketlemeyi elemektir.

Sonucun Aktarılması Bu aşamada cismin sınırları; orta nokta, dikey ve yatay uzunluk şeklinde elde edilir. OpenCV ile de bu cismin etrafına sınır kutusu çizilir. Bizim için önemli olan cismin sınırlarının hangi piksellerde olduğudur. Bu piksellerin konumundan ve kameranın görüş açısının bilgileri ile takip edilmesi istenen cismin konumunu elde edebiliriz. Yer istasyonunda tanımlama işlemi bittikten sonra bu veriler otonom kontrol için yönlendirme algoritmasına aktarılır. Burada zaman farkı çok önemli olduğundan işlenen her kare için ayrı ayrı zaman damgası vurulur.

Derin öğrenme tabanlı hedef tespit algoritmalarından üç algoritma üzerinde araştırma yapılmıştır ve karşılaştırılmıştır. Bu algoritmalar sık kullanılan Single Shot Detector (SSD) , You Only Look Once (YOLO) ve Faster Recurrent Neural Network (Faster-RCNN) algroitmalarıdır. Kullanılıcak algoritmada doğruluk oranı (mAP) ve FPS değeri önemlidir. FPS değeri saniyede alınan görüntü sayısıdır. Düşük doğruluk değerleri yanlış hedefleri tespit etmemize neden olabilir. Gerçek zamanlı görüntü işleyebilmek için FPS değeri yüksek olmalıdır. Yardımcı bilgisayarın işlemcisi ve hızı, kullanılacak

algoritmaların hızı FPS değerini doğrudan etkilemektedir. Bu nedenle algoritmalar bu iki parametre ile kıyaslanmıştır. Derin öğrenme tabanlı algoritmaların çalışma hızı ve doğruluğu göz önünde bulundurulduğunda YOLO algoritmasının kullanıma daha uygun olduğuna karar verilmiştir. YOLO, yapay sinir ağları kullanarak nesne tespiti yapan bir algoritmadır. YOLO modellerinde versiyonlarına göre katman sayıları değişmektedir.