

Sürü İnsansız Hava Aracı

1.Proje Gereksinimleri

1.1 Donanım Gereksinimleri

- En az 3 adet otonom uçuşa uygun İHA
- Her İHA'da GNSS, IMU, barometre, ve çarpışma önleme sistemleri
- Sadece keşif görevinde kullanılmak üzere kamera modülü
- Yer istasyonu ile haberleşme birimi (müdahale edilmeksizin)
- ArUco işaretçilerini tespit edebilecek çözünürlükte kamera (yalnızca keşif için)
- Formasyon sırasında X metre mesafe, Z metre irtifa

1.2 Yazılım Gereksinimleri

- Tüm görevler için tam otonom sürü algoritması
- Formasyon geçişleri, birey çıkarma/ekleme ve navigasyon senaryolarına uyumlu yazılım
- Haberleşme kaybı sonrası görev tamamlama algoritması
- ArUco işaretçisi tespiti ve hedef nokta iletimi için koordinasyon sistemi
- Tüm sistemlerde kesintisiz görev kontrolü (manüel müdahale olmadan)

1.3 Görev Gereksinimleri

- **3B Formasyon Görevi :** İHA'lar kalkıştan inişe kadar farklı formasyonlara geçerek bu düzeni korumalıdır.
- **Sürüyle Navigasyon Görevi :** Sürü belirli ara noktalardan geçerek formasyonu koruyarak hedefe ulaşır. Görev sırasında yer kontrol bağlantısı kesilir.
- **Birey Ekleme Çıkarma Görevi :** Sürünün içinden bir İHA çıkarılır, yeni bir birey dahil edilir. Sürü düzeni korunarak görev tamamlanır.
- **Keşif Görevi :** Keşif dronları arUco işaretçisini bulur ve konum bilgisini hedef dronlara iletir. Hedef dronlar işaretçiye hassas iniş yaparlar.

2. İHA'ların Özellikleri

2.1 Yazılım Arayüzü

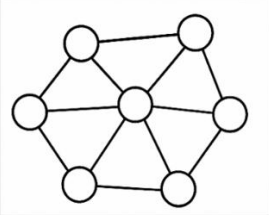
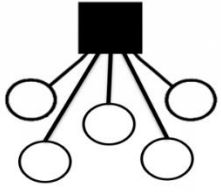
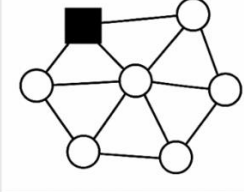
Görevlerin İHA'lara iletilmesi ve sürecin gözlemlenmesi için kullanımı kolay, tasarımı sade ve gerektirdiği işlem gücü düşük olan web tabanlı bir arayüz tasarlanması planlanmıştır.

Teknik açıdan bakıldığında sadece HTML, CSS ve JavaScript ile hızlı ve sade bir arayüz sağlamayı ve kullanıcı girdilerine göre gerekli aksiyonları gerçekleştirmek için de yine aynı prensiplerle geliştirilmiş bir sunucu geliştirilecektir. Arayüz ile sürü arası iletişimin sağlandığı bu sunucu içinde Python veya NodeJS kullanılacaktır.



YAZILIM ARAYÜZÜ

2.2 SÜRÜ İHA UÇUŞ KONTROLÜ YÖNTEMLERİ

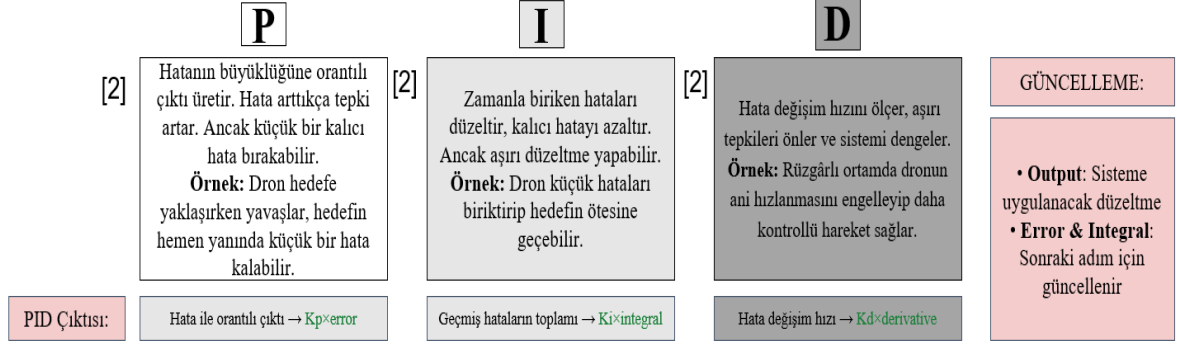
DAĞITIK KONTROL	MERKEZİ KONTROL	HİBRİT KONTROL
 <p>İHA'lar otonom karar alır, yerel iletişim kurar.</p> <p>Avantajlar:</p> <ul style="list-style-type: none"> • Dayanıklılık • Ölçeklenebilirlik • Tepki süresi kısa • Görev esnekliği yüksek 	 <p>Tüm İHA'lar merkezi bir kontrol birimi tarafından yönetilir</p> <p>Dezavantajlar:</p> <ul style="list-style-type: none"> • Tek hata noktası riski. • Yüksek iletişim ve hesaplama yükü • Dinamik görevler için yavaş tepki süresi 	 <ul style="list-style-type: none"> • Merkezi ve dağıtık kontrolün birleşimi. • Görevler merkezi olarak verilir, yerel kararlar otonom İHA'lar tarafından alınır. • Hem ağ merkezi hem otonom sistem elde edilir.

Yarışma görevleri için dağıtık model seçilmiştir. Dağıtık kontrol ; Bu görevlerde hızlı adaptasyon, formasyon esnekliği ve arıza toleransı sağlamaktadır.

2.3 PID Kontrol: İHA'lar için Akıllı Yönlendirme

PID, bir sistemdeki hatayı minimize etmek için kullanılan üç bileşenli bir kontrol yöntemidir. PID kontrolünün sistem üzerindeki etkisi, aşağıdaki matematiksel denklemlerle özetlenir:

$$m(t) = K_p e(t) + K_i \int e(t) + K_d \frac{d}{dt} e(t)$$



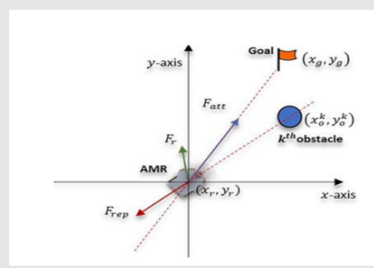
2.4 Yapay Potansiyel Alan (APF)

AMR'nin hedefe ulaşması ve engellerden kaçınması, çekici (F_{att}) ve itici (F_{rep}) kuvvetlerin etkileşimiyle sağlanır. [4]

- F_{att} : Hedeften gelen çekici kuvvet (Gauss fonksiyonu ile).
- F_{rep} : Engellerden gelen itici kuvvet.
- **AMR**, yol planlamasını kolaylaştırmak için noktasal kütle olarak modellenir.

Gauss Fonksiyonunun Özellikleri:

- ✓ **Hedefe Yaklaştıkça:**
d küçülür → Çekici kuvvet (F_{att}) artar → Robot hızlanır
- ✓ **Hedeften Uzaklaştıkça:**
d büyür → Çekici kuvvet (F_{att}) azalır → Robot yavaşlar
- ✓ **Avantajı:** Yumuşak geçişli kuvvet değişimi → Kararlı ve doğal hareket



Hedef (x_g, y_g) ve engel (x_o, y_o) noktaları, Otonom Mobil Robot'a kuvvet uygular. Çekim kuvveti (F_{att}) hedeften, itme kuvveti (F_{rep}) ise engelden kaynaklanır.

Yol planlama problemini basitleştirmek için, AMR (Autonomous Mobile Robot) bir noktasal kütle olarak kabul edilir ve çekim kuvveti (F_{att}), aşağıdaki gibi Gauss fonksiyonu kullanılarak ifade edilir. [4]

$$F_{att} = A * e^{-\frac{d^2}{2\sigma^2}}$$

$$d = \sqrt{(x - x_{target})^2 + (y - y_{target})^2}$$

Parametreler:

- A:** Maksimum kuvvet büyüklüğü
- σ (sigma):** Kuvvetin yayılımını kontrol eder
- d:** AMR ile hedef arasındaki mesafe

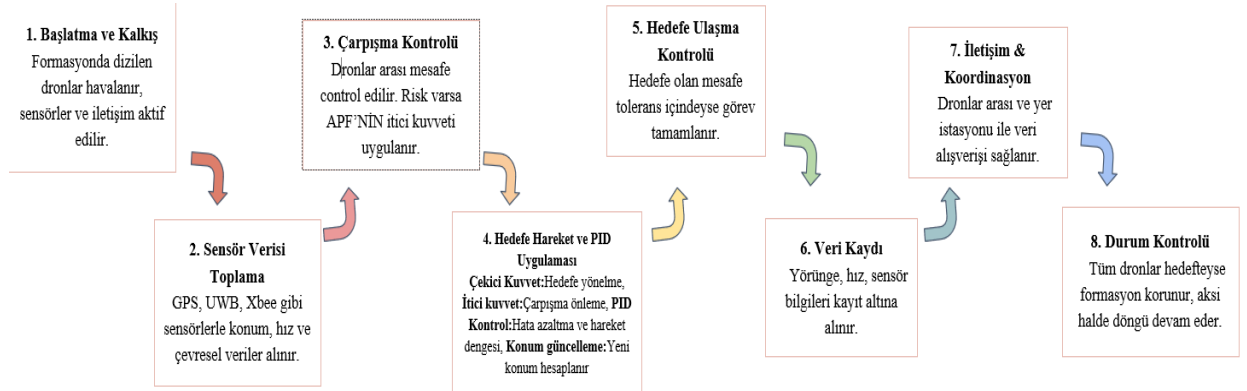
2.5 PID Destekli Yapay Potansiyel Alan

Neden PID Destekli APF?

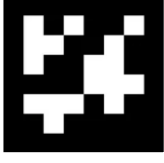
- **Gerçek Zamanlı:** APF, düşük hesaplama maliyetiyle dinamik ortamlarda hızlı tepki verir.
- **Hassas Kontrol:** PID, daha kararlı ve akıcı hareket sağlar.
- **Alternatiflere Göre Üstünlük:** PSO, ACO, ABC gibi yöntemler yüksek hesaplama gerektirir ve dinamik sistemlerde yetersiz kalabilir.

PID Destekli APF'nin Kullanım Alanları

- Dron yarışları
- Arama-kurtarma
- Askeri görevler
- Gerçek zamanlı sürü hareketi ve çarpışma önleme



2.6 ArUco İşaretçisinin Hazırlaması Ve Tespiti



İşaretçi, tespit için stabil olmalı ve rüzgâr, güneş gibi etkenlerden minimum etkilenmeli. Yüksek kontrastlı, mat kâğıda çıktı alınıp sabitlenmeli veya ağır bir plakaya yerleştirilmelidir.

Minimum ve maksimum boyutlar arasındaki işaretçilerin avantajları ve dezavantajları vardır. Raspberry Pi kamerasında boyutlar arasındaki genişlik, aşağıdaki formül ile piksel cinsinden hesaplanabilir.

ppi: pixel per inches (inç başına düşen piksel)

İşaretçinin inç cinsinden kenar uzunlukları arttıkça, tespit edilen piksel sayısı da artar, bu da tespiti kolaylaştırır.

Ayrıca, dictionary boyutu önemlidir; iniş noktası tespitinde yüksek boyutlu bir grid gereksizdir. "DICT_4X4_50" daha yüksek piksel vererek bu görev için yeterlidir.

Yaptığımız performans testlerinin sonuçlarına, görevin engelsiz bir ortamda yapılacağına ve işaretçi için verilen şartlara göre bir karar alındığında aşağıdaki yollar izlemeye karar verdik. İşaretçinin Hazırlanmasında İzlenecek Yol: 80x80 boyutta, "DICT_4X4_50" dictionary ile mat bir kâğıda veya yüksek kontrast ile çıktı alınıp bir mermer üzerine yapıştırılarak rüzgar vb. etkenlere karşı hazırlanacaktır.

Keşif sürecinde izlenecek yol:

1. Görev alanının iki eş parçaya bölünmesi
2. Dronların kendi parçalarını grid tabanlı bir arama yöntemiyle keşfetmesi
3. İşaretçinin tespit edilmesi
4. İşaretçi konumunun 3. Drona gönderilmesi
5. 3. Dronun gönderilen konuma hareket edip iniş yapması

2.7 Kullanılan Ana Elektronik Birleşenler

- **Pixhawk Cube Orange**

Üçlü IMU yedekleme ile yüksek kararlılık, H7 işlemci desteğiyle otonom kontrol imkanı sunar.

- **Cube Pilot Pixhawk Here 3 GPS + RTK Base Modülü**

RTK teknolojisiyle santimetre hassasiyetli GNSS verisi ve sekronize uçuşa olanak sağlar. [8] [9]

- **XBee 3 Modülleri**

2.4 GHz frekans bandında mesh yapı üzerinden, MAVLink protokolü ile güvenilir bir haberleşme sağlanmaktadır. [10]

- **UWB Modülü (DWM)**

GNSS'in yetersiz kaldığı senaryolarda, kısa mesafede yüksek doğruluklu konumlandırma imkanı sunulmaktadır. [7]

NOT: UWB ihtiyaç halinde kullanıma karar verilecektir. Kesin bir kullanım söz konusu değildir.



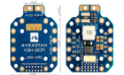
- **Skywalker ESC 30A & SunnySky 2216 980KV Motorlar**

Düşük hızlarda yüksek tork üreterek verimli bir itki performansı sağlar.



- **Matek PDB-HEX Güç Dağıtımı**

Stabil ve güvenli enerji yönetimi ile 12S desteğiyle yüksek güç çıkışı sağlanmaktadır.



- **Raspberry Pi 4 + Kamera V2/V3**

Görüntü işleme ve ArUco tespiti için güçlü bir altyapı sunarak görev yönetimine etkin destek sağlar.



2.8 Sürü Görevlerinin İHA'lara Uygulanması İçin Altyapı ve Yazılım Çözümleri

Geliştirdiğimiz sürü droneleri kontrol sistemi, **modüler ve ölçeklenebilir** bir mimariye sahiptir. Yazılımın temel bileşenleri şunlardır:

- Yazılım Dili:** Python (iletişim ve görev yönetimi)
- Haberleşme Protokolü:** MAVLink (Micro Air Vehicle Link) üzerinden UDP/TCP protokolleri ile droneler arası ve yer kontrol istasyonu iletişimi
- Kontrol Katmanı:** MAVLink tabanlı MAVProxy arayüzü, ROS 2 ile entegre edilerek insansız hava araçlarının (İHA) programlanması ve kontrol edilmesi mümkün hale gelmektedir.
- Simülasyon Ortamı:** Gazebo ile hem fiziksel modelleme hem de sürü senaryoları test edildi.
- Görev Dağıtımı:** Her bir İHA'ya görevlerin dağıtık algoritmalarla atanması. Kullanılan algoritmalar arasında:
 - Birey Çıkarma Görevi İçin : Potansiyel Alan + PID
 - Sürü Halinde Navigasyon Görevi İçin : PID Destekli APF
 - Sürü Keşif Görevi İçin : Potansiyel Alan + PID
 - Alan Tarama İçin : Flood Fill Algoritması

#Potansiyel Alan Yöntemi (Potential Field)

```
force = Vector(0, 0)
for obstacle in obstacles:
    force += repulsive_force(drone, obstacle)
force += attractive_force(drone, goal)
drone.move(force)
```

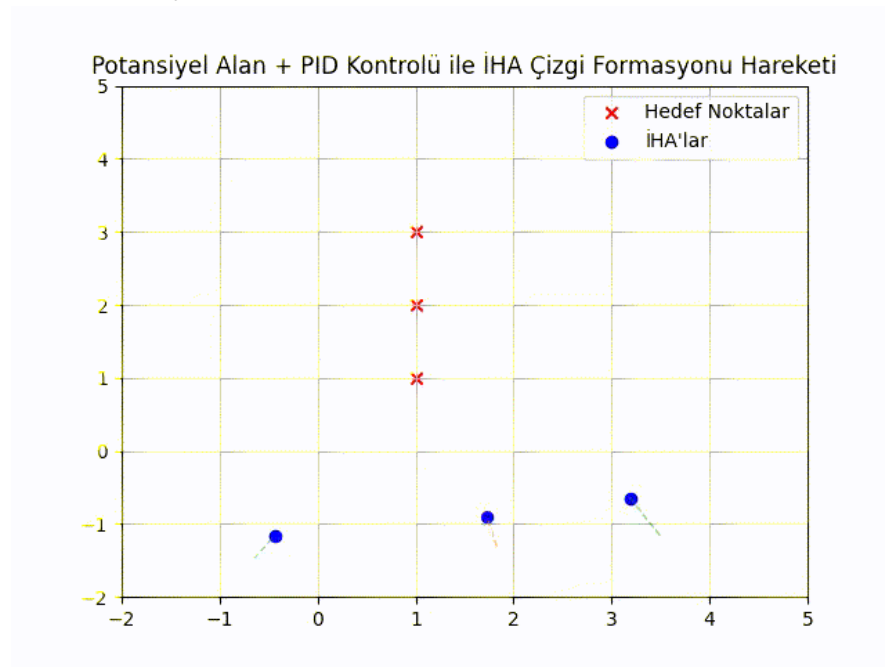
#PID Kontrol Örneği

```
error = setpoint - current_value
integral += error * dt
derivative = (error - prev_error) / dt
output = Kp * error + Ki * integral + Kd * derivative
```

#Karma Kullanım (Potansiyel + PID Örneği)

```
force = potential_field(drone, goal, obstacles)
desired_angle = math.atan2(force.y, force.x)
angle_error = desired_angle - drone.angle
drone.rotate(PID(angle_error))
drone.forward()
```

3 .Formasyon ve Uçuş



3.1 Pseudo kod ve Akış Diyagramı

init():

```
drone_pos = start_positions  
errors, integrals = zero_arrays()
```

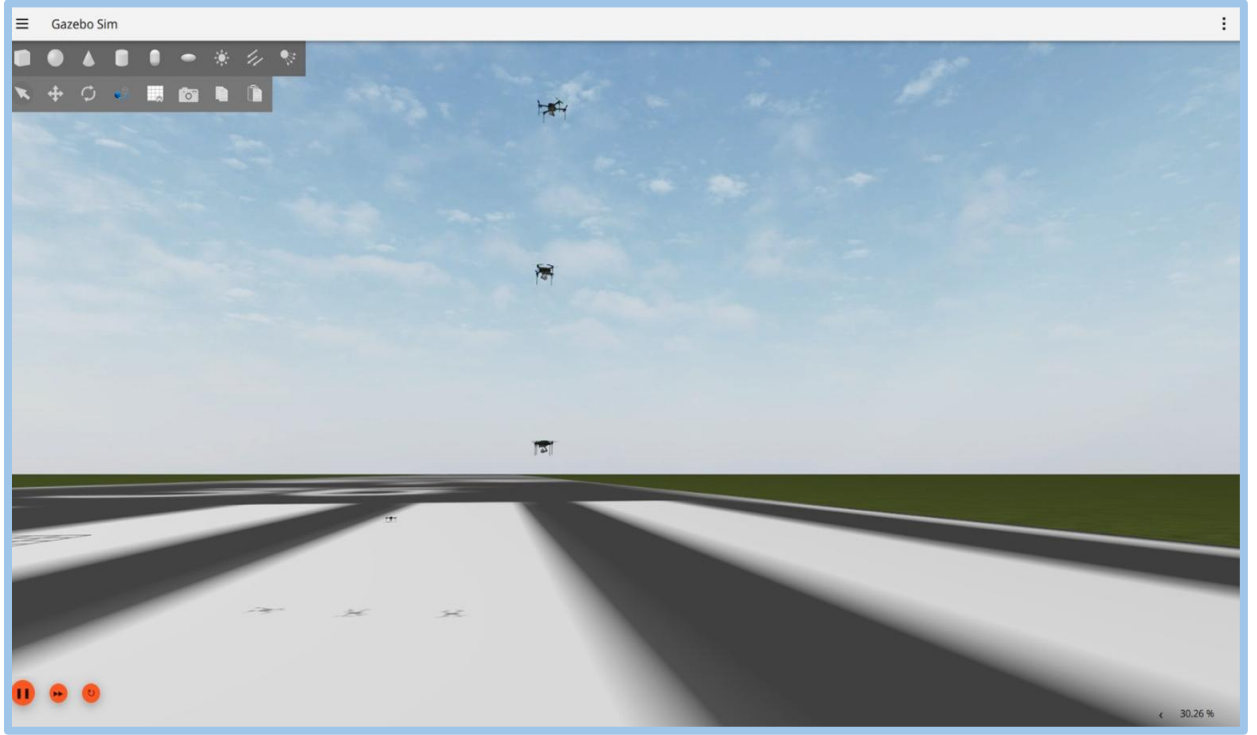
update():

```
for each drone i:  
    F_attr = attraction(drone_pos[i], target[i])  
    F_rep = repulsion(drone_pos[i], others[i])  
    F_pid, errors[i], integrals[i] = pid_control(drone_pos[i], target[i],  
    errors[i], integrals[i])  
    drone_pos[i] += F_attr + F_rep + F_pid  
    check_goal(drone_pos[i], target[i])
```

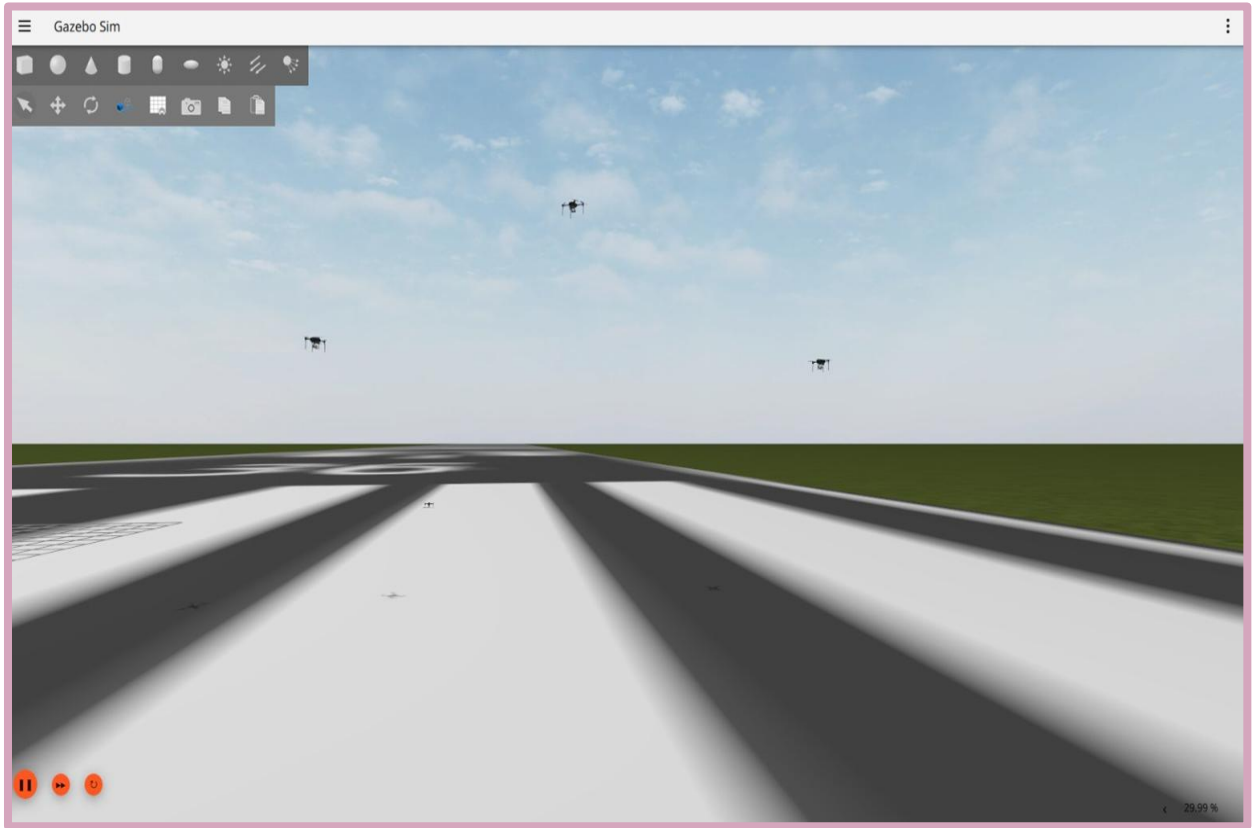
main():

```
init()  
while not all_reached:  
    update()
```

3.2 Simülasyon



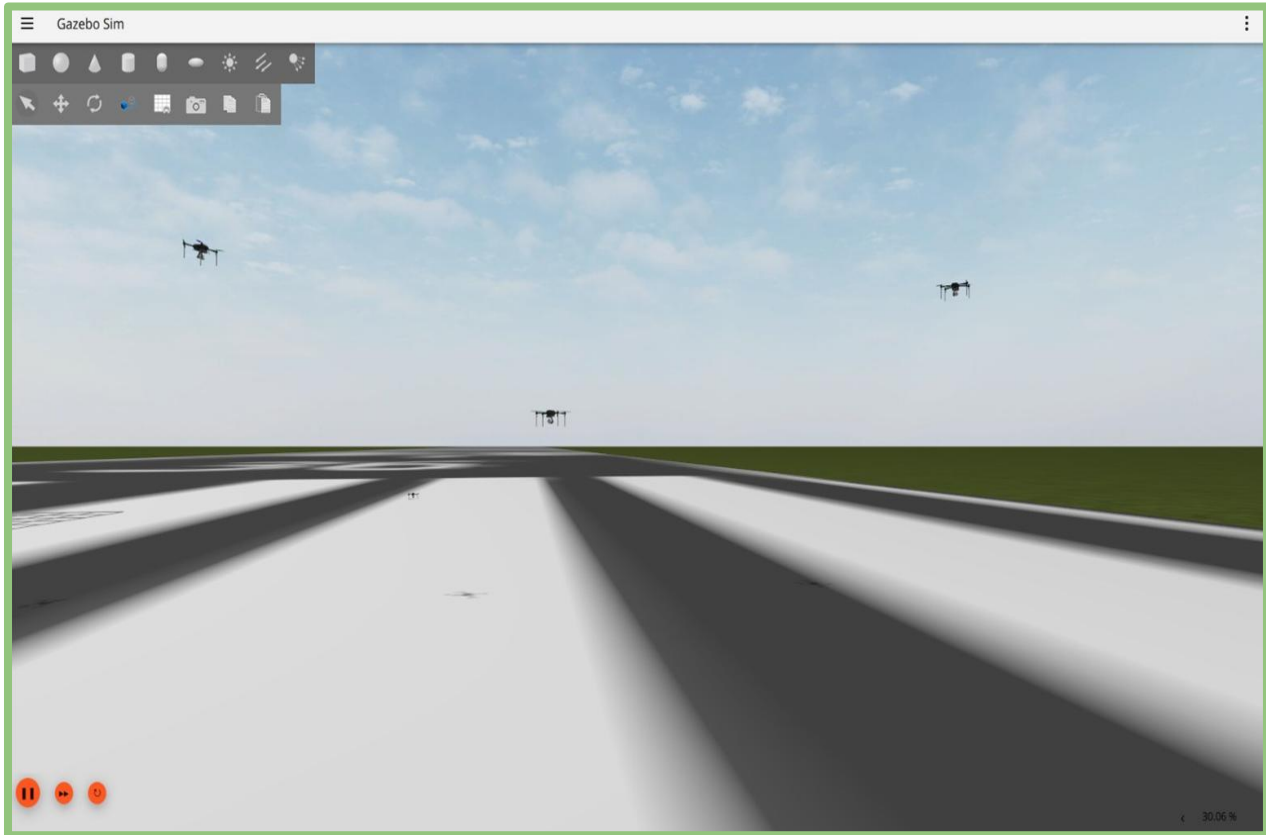
Çizgi Formasyonu



Ok Başı Formasyonu



Serbest Formasyon



V-Formasyonu

223908046 Kutay Paça

