
PROJE RAPORU

MAVI PROJE

TAMAMLANDI

# Göksu (2025)

*Yazar(lar):*

Famura

# İçerikler


1. Proje Tanımı ..... 3
   1.1. Giriş ..... 3
   1.2. Yarışma Detayları ..... 3
      1.2.1. Parkur Detayları ..... 3
      1.1.1. Ana Gereksinimler ..... 4
2. Proje Teknolojileri ..... 4
   2.1. Modüller ..... 4
   2.2. Programlama Dilleri ..... 4
   2.3. Kütüphaneler ve Harici Eklentiler ..... 5
3. Proje Mimarisi ..... 5
   3.1. Proje Mimarisi ..... 5
   3.2. Beden ..... 6
      3.2.1. Girdi ..... 6
      3.2.2. Çıktı ..... 6
   3.3. Zihin ..... 7
      3.3.1. Çalıştırma (Execute) ..... 7
      3.3.2. Bilgi ..... 7
      3.3.3. Dil ..... 7
      3.3.4. Öğrenme ..... 7
4. Proje Analizi ..... 7
   4.1. Tests ..... 7
   4.2. Geliştirilebilir Alanlar ..... 8
5. Project Comments ..... 8
   5.1. Creator Comments ..... 8
6. Credits ..... 8
7. Referanslar ..... 9
8. Disk İçeriği ..... 9

# 1. Proje Tanımı

## 1.1. Giriş

Göksu, Teknofest 2025 İnsansız Deniz Aracı yarışma parkurunu tamamlamak üzere tasarlanmıştır. Ne yazık ki projenin kaynak kodları gizlidir. Bu nedenle bu rapor herhangi bir kod içermemektedir. Ancak sistemi tek başıma (Famura) tasarladığım için mimariyi açıklama ve görselleştirme hakkına sahibim. Bu yüzden projeye ait görseller raporda yer almaktadır. Ayrıca bu rapor yalnızca Göksu'nun **yazılım tarafına** odaklanmaktadır.

Geçen yıldan (Trieme) farklı olarak Göksu, bir **yapay zekâ modeliyle** çalışacak şekilde tasarlanmıştır. Bu yıl projeye ayırabildiğimiz süre 8 aydı ve 4 kişilik bir yazılım ekibiyle çalıştık. Sürecin sonunda projeyi başarıyla tamamladık.

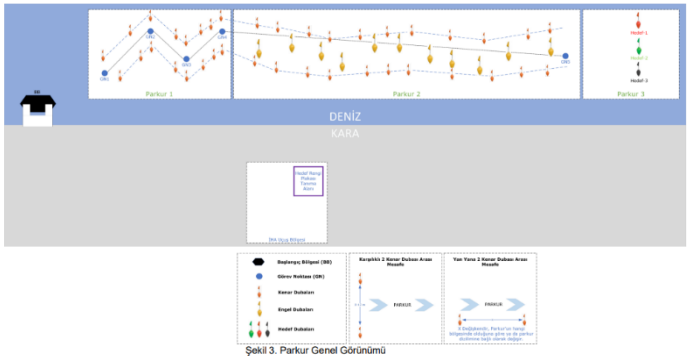
## 1.2. Yarışma Detayları

### 1.2.1. Parkur Detayları

Yarışma parkuru bu yıl üç ana bölümden oluşmaktadır:

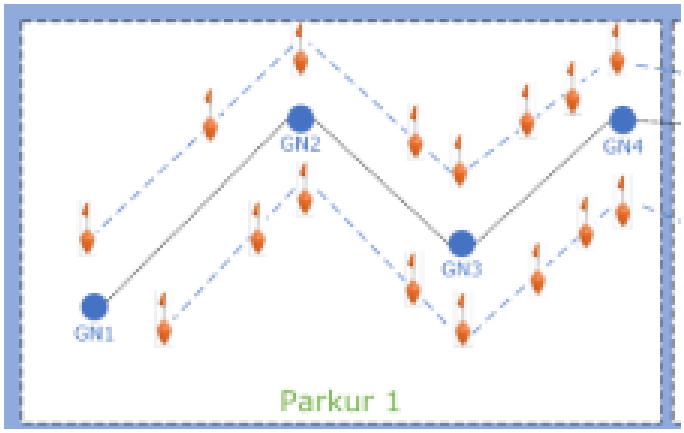
- Parkur 1
- Parkur 2
- Parkur 3

Gemi, bu parkurların tamamını **tek ve kesintisiz bir denemede** tamamlamak ve ardından geri dönmek zorundadır. Tüm parkur için iki deneme hakkı verilmiştir. Parkur sırasında geminin herhangi bir nesneye temas etmesi ya da parkur dışına çıkması yasaktır. Bu durumlardan herhangi biri ceza puanı ile sonuçlanır. Ceza puanlarının tekrarı diskalifiye anlamına gelir.



Şekil 3. Parkur Genel Görünümü

Aşağıda parkurların detaylı açıklamaları yer almaktadır.




#### *1.2.1.1. Parkur 1*

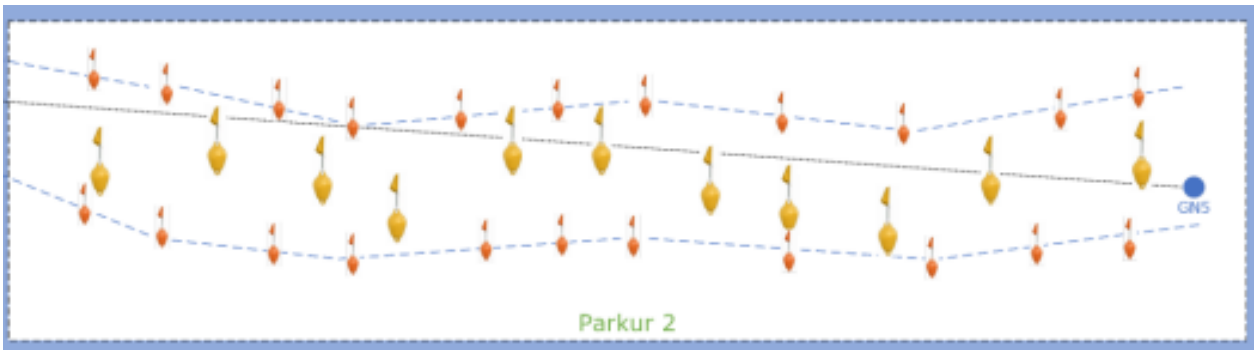
Parkur 1, geminin rota bulma ve görüntü işleme algoritmalarını test etmek amacıyla eklenmiştir. Parkur boyunca sınır engelleri bulunmaktadır. Bu parkurda hem sol hem de sağ sınır engelleri **turuncu renktedir**, bu da sınırların sınıflandırılmasını zorlaştırmaktadır.

Gemi, sınır engelleri arasında kalarak ve hiçbir nesneye çarpmadan parkuru tamamlamak zorundadır.

#### *1.2.1.2. Parkour 2*

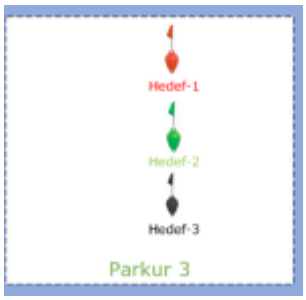
Parkur 2, rota bulma, görüntü işleme ve engelden kaçınma algoritmalarını test etmek amacıyla eklenmiştir. Bu parkurda da sol ve sağ sınır engelleri yine **turuncu renktedir**, bu da sınır ayrımını zorlaştırır. Ayrıca parkurun ortasında **sarı renkte kaçınılması gereken engeller** bulunmaktadır.

Gemi, sınırların arasında kalarak ve hiçbir nesneye çarpmadan parkuru tamamlamak zorundadır.




#### *1.2.1.3. Parkur 3*

Parkur 3, haberleşme ve rota bulma algoritmalarını test etmek için tasarlanmıştır. Otonom bir drone havalanarak mevcut üç hedeften hangisinin seçileceğine dair bilgiyi toplar. Ardından bu bilgiyi gemiye iletir ve gemi belirtilen hedefe yönelik **kamikaze görevini** gerçekleştirir.




### 1.1.1. Ana Gereksinimler

Yarışmaya katılabilmek için yazılım ekibi tarafından karşılanması gereken temel gereksinimler şunlardır:

- Bir Yer Kontrol İstasyonu (GCS) bulunmalıdır
- Gemi GCS üzerinden kontrol edilebilmelidir
- Tüm telemetri verileri GCS üzerinden erişilebilir olmalıdır
- Gemi çevresini haritalandırmalı ve bu harita GCS üzerinden gerçek zamanlı görüntülenebilmelidir
- Gemi hem GCS hem de herhangi bir RC kontrolcü ile manuel olarak kontrol edilebilmelidir
- Gemi üç moda sahip olmalıdır: durma, manuel ve otonom
- Kullanıcılar bu modlar arasında istedikleri zaman geçiş yapabilmelidir

# 2. Proje Teknolojileri

## 2.1. Modüller

- **Windows (Windows 10):**
  GCS, Windows üzerinde çalışacak şekilde geliştirilmiştir. Casper Excalibur G770 üzerinde kullanılmıştır.
- **Ubuntu (Ubuntu 22.04):**
  Göksu, Ubuntu üzerinde çalışacak şekilde geliştirilmiştir. Jetson Orin Nano kullanılmıştır.
- **Rasbery Pi OS:**
  Drone, Raspberry Pi OS üzerinde çalışmaktadır. Raspberry Pi 4 kullanılmıştır.

## 2.2. Programlama Dilleri

- **Python - Pycharm**
  Projenin tamamı Python ile yazılmıştır ve Python 3 üzerinde çalışacak şekilde ayarlanmıştır. Tüm sanal ortamlar BIOS seviyesinde servis edilmekte ve ilgili elektronik modül güç aldığında otomatik olarak başlatılmaktadır.

### 2.3. Kütüphaneler ve Harici Eklentiler

- **PySerial:**
  **Elektronik modüller arası iletişimi sağlar. Telemetri modülleri, Wi-Fi ve mobil hat frekanslarıyla çakışmaması için elektronik ekip tarafından seçilmiştir.**
- **YOLO:**
  **Görüntü işleme için kullanılmıştır.**
- **Threading:**
  **Paralel iletişim süreçlerinin yönetimi için kullanılmıştır.**
- **PyGame:**
  **GCS üzerinde telemetri verilerinin ve canlı haritanın gösterimi için kullanılmıştır.**
- **PyMavlink:**
  **Jetson Orin ile Pixhawk modülü arasındaki bağlantı için kullanılmıştır.**
- **TensorFlow:**
  **Aelita'nın öğrenme ve karar verme modüllerini destekler. Rota tahmini ve adaptif davranışlar için sinir ağlarının eğitimi ve çıkarımı bu yapı üzerinden yapılır.**
- **WhisperDock:**
  **Sesli girdi işleme için kullanılır. Radyo üzerinden iletilen sesli komutları metne çevirerek Aelita'nın yarışma sırasında komut almasını sağlar.**
- **XTTS-v2:**
  **Sesli çıktı üretimi için kullanılır. Aelita, komutlara yanıt verirken ve durum bilgisi paylaşırken sentezlenmiş konuşma kullanır.**
- **Unity (C#):**
  **Simülasyon testleri için kullanılmıştır. Fiziksel testlerden önce Aelita'nın davranışlarını gözlemlemek amacıyla özel sanal ortamlar oluşturulmuştur.**
- **Robot Operating System (ROS):**
  **Alt sistemler arası modüler iletişim için entegre edilmiştir. Özellikle çok iş parçacıklı yapılarda mesajlaşma ve sensör füzyonunu destekler.**

# 3. Proje Mimarisi

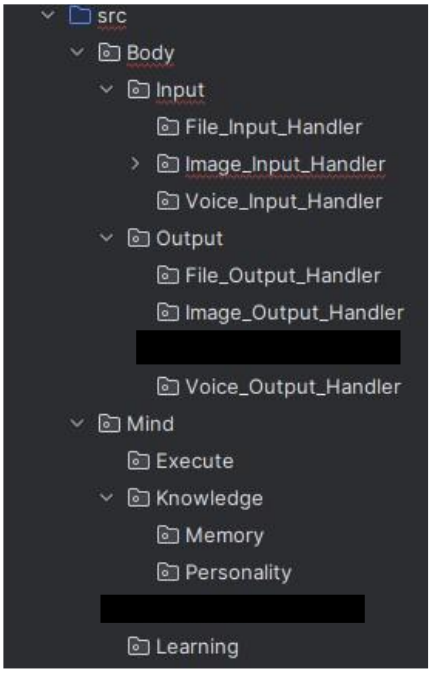
## 3.1. Proje Mimarisi

Proje iki ana bölümden oluşmaktadır: **beden** ve **zihin**. Beden, dış çevreyle etkileşimi sağlayan arayüzleri temsil eder; görüntü, ses ve sistem dosyaları gibi girdileri işler ve bunlara karşılık gelen çıktıları üretir. Zihin ise içsel bilişsel süreçleri kapsar; girdileri analiz eder, bilgiyi depolar, iletişim kurar ve dinamik şekilde öğrenir. Bu iki yapının sorunsuz entegrasyonu, Aelita'nın zorlu deniz senaryolarında güvenilir ve tutarlı bir şekilde çalışmasını sağlar.

Ekip yaklaşımı, insan vücudu ve beyninin birlikte çalışma prensiplerinden ilham almıştır; ancak doğrudan herhangi bir mevcut yapay zekâ mimarisini taklit etmez. Aelita'nın işlevselliği; gerçek zamanlı karar verme, hata yönetimi ve kendi kendini düzeltme yetenekleri üzerine kuruludur ve küçük ölçekli deniz araçlarını stabil şekilde yönlendirebilecek biçimde tasarlanmıştır. Makine öğrenmesi için TensorFlow, elektronik rota yönetimi için Pixhawk ve haberleşme için MAVLink gibi temel araçlar bu geliştirme yaklaşımını desteklemektedir.

Donanım ve yazılım kütüphaneleriyle yüksek uyumluluğu nedeniyle Python'un tercih edilmesi, geliştirme sürecini sadeleştirmiş ve sistem entegrasyonunu verimli hâle getirmiştir. Proje ilerledikçe yapay zekâ yapısında yapılan iteratif iyileştirmeler, sistemin yeni zorluklara uyum sağlamasını ve yarışma beklentilerini karşılamasını mümkün kılmaktadır.

Önerilen plan kapsamında Aelita (dolayısıyla tüm çalışma), farklı alt bölümlere ve farklı yaklaşımlara ayrılmıştır. Bu ayrım tamamen yazılım ekibimiz tarafından yapılmış olup herhangi bir mevcut yapay zekâdan kopyalanmamıştır. Ancak doğal olarak, insan beyninin ve insan vücudunun çalışma biçimlerinden ilham alınmıştır.




Solda görüldüğü üzere Aelita'yı iki ana bölüm ve bu bölümlerin alt parçaları olacak şekilde ayırdık. Bu yapının çalışma mantığını, Java'daki sınıf (class) yapısına benzetmek mümkündür.

**Beden** kısmında Aelita'nın çevresiyle etkileşime girmesini hedefliyoruz. Motor ve kontrol bileşenleri henüz tamamlanmadığı için, bu bileşenlere ait handler'lar şu an için oluşturulmamıştır.

**Zihin** kısmında ise Aelita'nın kendisini inşa etmeyi amaçlıyoruz. Bu bölüm, onun beyni gibi çalışacaktır; karar verme, öğrenme ve farklı dilleri anlama yetenekleri bu katmanda yer alacaktır.

Şimdi bu alt bölümlerin her birine tek tek bakalım:

## 3.2. Beden

### 3.2.1. Girdi

1) **File_Input_Handler:**
   Aelita'nın sistem içi girdileri okumasını sağlar. Sistem loglarını okuyabilir. Ayrıca sesli iletişimin kesildiği durumlarda .txt dosyaları üzerinden iletişim kurmak için kullanılır.
2) **Image_Input_Handler:**
   Aelita'nın görüntü işleme süreçlerinden sorumludur.
3) **Voice_Input_Handler:**
   Aelita ile sesli iletişimi sağlar. Bu modülde WhisperDock kullanılmıştır. Yarışma sırasında radyo üzerinden Aelita'ya komut vermemizi mümkün kılar.

### 3.2.2. Çıktı

1. **File_Output_Handler:**
   Aelita'nın loglarını .txt formatında oluşturur ve gerektiğinde farklı dosya türleri üretmesini sağlar.
2. **Image_Output_Handler:**
   Görüntü işleme sonuçlarını dışa aktarır. Görüntü işleme bu projenin en zor kısımlarından biri olduğu için, bu modül bir kontrol noktası olarak tasarlanmıştır. Sürekli çalışmaz; yalnızca istendiğinde çıktı üretir. Bu sayede hatanın nerede olduğunu kolayca tespit edebiliriz.
3. **Voice_Output_Handler:**
   Aelita'nın sesli geri bildirim vermesini sağlar. Bu modülde XTTS-v2 kullanılmıştır.

### 3.3. Zihin

#### 3.3.1. Çalıştırma (Execute)

Bu bölüm, Aelita'nın beden ile bağlantı kurmasını sağlar. Hangi çıktının hangi beden modülü üzerinden verileceğine karar verir. Gerekirse Öğrenme modülüyle iletişime geçerek hangi yolun seçileceğine karar verir. Girdiler için bu yapı gerekli değildir; zihin modülleri doğrudan ilgili beden modüllerini kullanır.

#### 3.3.2. Bilgi

Aelita'nın veri depolamasını sağlar. İki ana parçaya ayrılır:

- **Hafıza:** Tüm verilerin saklandığı alan
- **Kişilik:** Karar verme ve risk alma süreçlerini kolaylaştırmak için kullanılır

Kişilik oluşturulurken üç figürden ilham alınmıştır:

- **Niki Lauda:** Mühendis yaklaşımı
- **Margaret Thatcher:** Güçlü ve cesur duruş
- **Misato Katsuragi:** Kadın bakış açısı

#### 3.3.3. Dil

Aelita'nın Türkçe ve İngilizce iletişim kurmasını sağlar. Ancak bu yapıdan vazgeçilmesi ve yalnızca İngilizce iletişime geçilmesi planlanmaktadır.

#### 3.3.4. Öğrenme

Makine öğrenmesi ve karar verme yeteneklerini içerir. Projenin en kritik bölümüdür. TensorFlow bu modülde kullanılmaktadır. Zamanla bu yapı **karar verme** ve **öğrenme** olarak ikiye ayrılacaktır. Şu an yeterli veri olmadığı için net bir sınıflandırma yapılmamıştır.

## 4. Proje Analizi

### 4.1. Tests

Proje hem fiziksel hem de simülasyon ortamlarında test edilmiştir. Model Unity (C#) üzerinde ve gerçek koşullarda test edilmiştir. Ayrıca Aelita, **Need for Speed Underground 2** ve **Trackmania Nations Forever** oyunları kullanılarak test edilmiştir.

NFSU2'de otoyollarda net sınırlar bulunduğu için Aelita'ya bu sınırlar arasında kalması ve trafikten kaçınması öğretilmiştir.


Trackmania'da ise pistlerin çoğunda net sol ve sağ engeller bulunmaktadır. Bu yapı da aynı amaçla kullanılmıştır.

### 4.2. Geliştirilebilir Alanlar

İlerleyen versiyonlarda Aelita'nın öğrenme ve karar verme yetenekleri, pekiştirmeli öğrenme teknikleriyle daha da geliştirilebilir. Sesli iletişim sistemi çok dilli destek sunacak şekilde genişletilebilir. Ayrıca GCS tarafına daha gelişmiş bir 3D görselleştirme sistemi entegre edilerek operatörler için daha iyi durumsal farkındalık sağlanabilir.

## 5. Project Comments

### 5.1. Creator Comments

2025 yarışması bana, 2024'te öğrendiklerime ek olarak çok şey kattı. Bu sefer bir **veteran** olmayı, bir **lider** olmayı öğrendim. Sadece birlikte çalıştığın insanları yönetmeyi değil, seninle aynı hedefi isteyen ama sana güvenen insanların sorumluluğunu taşımayı da. Hata yapmanın sorumluluğu ve bunun sürekli baskısı altında çalışmak…

**Hooah**

**– Famura**

## 6. Credits

- Famura – Yazılım Departmanı Lideri
- Musab Kılıç – Yazılım Ekip Üyesi (Görüntü İşleme)
- Ali Haydar Sucu – Yazılım Ekip Üyesi
- Abdullah Aksoy – Yazılım Ekip Üyesi

## 7. Referanslar


- MathWorks. (n.d.). *Dijkstra's algorithm*. In *MATLAB & Simulink documentation*. Retrieved October 18, 2025, from https://www.mathworks.com/help/matlab/ref/graph.shortestpath.html
- OpenCV. (n.d.). *YOLO object detection with OpenCV*. Retrieved October 18, 2025, from https://docs.opencv.org/
- PyGame Community. (n.d.). *PyGame documentation*. Retrieved October 18, 2025, from https://www.pygame.org/docs/
- PyMavlink Developers. (n.d.). *pymavlink: Python MAVLink library*. Retrieved October 18, 2025, from https://www.ardusub.com/developers/pymavlink.html
- Python Software Foundation. (n.d.). *Python 3 documentation*. Retrieved October 18, 2025, from https://docs.python.org/3/
- Raspberry Pi Foundation. (n.d.). *Raspberry Pi OS documentation*. Retrieved October 18, 2025, from https://www.raspberrypi.com/software/
- Robot Operating System (ROS). (n.d.). *ROS documentation*. Retrieved October 18, 2025, from https://www.ros.org/
- Serial Communication Library (PySerial). (n.d.). *PySerial documentation*. Retrieved October 18, 2025, from https://pyserial.readthedocs.io/
- Unity Technologies. (n.d.). *Unity documentation*. Retrieved October 18, 2025, from https://docs.unity.com/
- Google. (n.d.). TensorFlow: An end-to-end open source machine learning platform. Retrieved October 18, 2025, from https://www.tensorflow.org/
- NVIDIA Corporation. (n.d.). Jetson Orin Nano developer kit. Retrieved October 18, 2025, from https://developer.nvidia.com/embedded/jetson-orin
- ElevenLabs. (n.d.). XTTS-v2: Cross-lingual text-to-speech model. Retrieved October 18, 2025, from https://elevenlabs.io/
- WhisperDock Developers. (n.d.). WhisperDock: Voice recognition and processing toolkit. Retrieved October 18, 2025, from https://github.com/openai/whisper
- Unity Technologies. (n.d.). Unity real-time development platform. Retrieved October 18, 2025, from https://unity.com/
- Electronic Arts. (2003). Need for Speed: Underground 2 [Video game]. Electronic Arts.
- Nadeo. (2006). TrackMania Nations Forever [Video game]. Ubisoft.


## 8. Disk İçeriği

- Göksu.pdf
- Göksu.docx
- Ekip_Fotoğrafı.jpg
- Otonom_Testi.mp4
- Image_Proccessing_Test.mp4
- Testing_Photo.jpg
- Uluslararası_Kurula_Sunum.mp4
- Canon In D by Evangelion Series