



**YILDIZ TEKNİK ÜNİVERSİTESİ  
MAKİNA MÜHENDİSLİĞİ FAKÜLTESİ**

**OTONOM BİR YARIŞ ARACININ  
TASARLANMASI**

15067004 Zeynep Esra İşler

16067023 Mahmut Reyhani

16067604 Alperen Şentürk

**MEKATRONİK MÜHENDİSLİĞİ ANABİLİM DALINDA HAZIRLANAN**

**BİTİRME ÇALIŞMASI RAPORU**

**Proje Danışmanı:** Doç. Dr. Mehmet Selçuk Arslan

İSTANBUL, 2022

# OTONOM BİR YARIŞ ARACININ TASARLANMASI

## İÇİNDEKİLER

Sayfa

REVİZYON TARİHÇESİ .....	v
SİMGELİSTESİ (Alfabetic) .....	1
KISALTMA LİSTESİ .....	4
ŞEKİL LİSTESİ .....	5
TABLO LİSTESİ .....	6
1. GİRİŞ .....	10
1.1 Problem Tanımı .....	10
1.2 Projenin Motivasyonu .....	10
1.3 Projenin Amacı .....	11
1.4 Projenin Hedefleri .....	12
1.5 Literatür Taraması .....	12
1.6 Kabuller ve Varsayımlar .....	13
2. GEREKSİNİM SPESİFİKASYONLARI .....	14
2.1 Paydaş Beklentileri .....	14
2.2 Teknik Gereksinim Spesifikasyonları .....	15
3. TEORİK ALTYAPI BİLGİLERİ VE YÖNTEMLER .....	16
3.1 Geliştirme Ortamları .....	16
3.1.1 MATLAB .....	16
3.1.2 ROS .....	16
3.1.3 CARLA Simülörü .....	18
3.2 Lokalizasyon ve Haritalandırma .....	19
3.3 Aracın Kinematik Bisiklet Modeli .....	19
3.4 RRT Algoritması .....	22
3.5 Pure Pursuit .....	24
4. TASARIM .....	28
4.1 Tüm Sistemin Akış Şeması .....	28
4.2 Donanım .....	29
4.3 Yazılım .....	32
4.3.1 Giriş ve Genel Yaklaşım .....	32
4.3.2 Algılama .....	34
4.3.2.1 LiDAR Tabanlı Algılama .....	34
4.3.2.2 Kamera Tabanlı Algılama .....	35
4.3.3 Hareket Tahmini ve Haritalandırma .....	36
4.3.4 Kontrol .....	37
5. TASARIM DOĞRULAMA .....	39
5.1 Pist Özellikleri .....	39
5.2 Haritalandırma ve Lokalizasyon Simülasyonu .....	42

5.3	İlk Turun Tamamlanması .....	43
5.4	Yarış Turlarının Tamamlanması.....	44
5.5	Alınan Hatalar ve Önlemler.....	44
5.6	İlk Turda (4 m/s) Hızdaki Test Sonuçları .....	45
5.7	17 m/s Hızdaki Test Sonuçları.....	47
6.	ÇALIŞMA TAKVİMİ .....	51
7.	BÜTÇE .....	52
7.1	Malzeme Listesi.....	52
8.	SONUÇLAR VE ÖNERİLER.....	53
	KAYNAKÇA .....	55
	ÖZGEÇMİŞ.....	57

## REVİZYON TARİHÇESİ

Tarih	Rev. No	Tanımı	Yazar(lar)
15/05/2021	1.0	Başlangıç dökümanı	Mahmut REYHANI Zeynep Esra İŞLER Alperen ŞENTÜRK
18/06/2021	1.1	Sponsor görüşmesi için yapılan revize	Mahmut REYHANI Zeynep Esra İŞLER Alperen ŞENTÜRK
21/06/2021	1.2	MST Final Raporu için güncellendi.	Mahmut REYHANI Zeynep Esra İŞLER Alperen ŞENTÜRK
25/06/2021	1.3	Jüri ve danışman değerlendirmeleri göz önüne alınarak güncellendi.	Mahmut REYHANI Zeynep Esra İŞLER Alperen ŞENTÜRK
18/12/2021	1.4	BÇ Final Raporu için güncellendi.	Mahmut REYHANI Zeynep Esra İŞLER Alperen ŞENTÜRK
15/01/2022	1.5	BÇ Final Raporu – İlk Hali için güncellendi	Mahmut REYHANI Zeynep Esra İŞLER Alperen ŞENTÜRK
21/01/2022	1.6	BÇ Final Raporu için güncellendi.	Mahmut REYHANI Zeynep Esra İŞLER Alperen ŞENTÜRK
03/02/2022	1.7	BÇ Final Raporu – Son Hali için güncellendi	Mahmut REYHANI Zeynep Esra İŞLER Alperen ŞENTÜRK

## SİMGE LİSTESİ (Alfabetik)

$C$	Sabit sayı
$F$	Kuvvet [ $N$ ]
$v$	Hız [ $m/s$ ]
$\delta$	Tekerleğin araç doğrultusu ile açısı [ $^{\circ}$ ]
$L$	Aracın Kinematik Bisiklet Modeline göre uzunluğu [ $m$ ]
$I_d$	Bakış mesafesi [ $m$ ]
$\alpha$	Hedeflenen dönüş açısı [ $^{\circ}$ ]
$x_{COG}$	Arabanın enlemesine ekseni
$y_{COG}$	Arabanın boylamasına ekseni
$\alpha_F$	Ön lastik kayma açısı [ $^{\circ}$ ]
$\alpha_R$	Arka lastik kayma açısı [ $^{\circ}$ ]
$R$	Oluşturulan yayın yarı çapı [ $m$ ]
$F_{tR}$	Ön tekerleğin koordinat sisteminin orjine etki eden enine kuvvet [ $N$ ]
$F_{tF}$	Arka tekerleğin koordinat sisteminin orjine etki eden boyuna kuvvet [ $N$ ]
$v_{COG}$	Aracın ağırlık merkezinde oluşan hız
$l_R$	Arka tekerleğin aracın ağırlık merkezine olan uzaklığı
$l_F$	Ön tekerleğin aracın ağırlık merkezine olan uzaklığı
$v_{WF}$	Ön tekerlek hızı [ $m/s$ ]
$v_{WR}$	Arka tekerlek hızı [ $m/s$ ]
$c_R$	Ön tekerlek sabiti
$c_F$	Arka tekerlek sabiti
$\dot{\psi}$	Arabanın açısal hızı [ $m/s^2$ ]

**Yunan Harfleri**

- $\beta$  Araç gövdesi yan kayma açısıdır.
- $\delta$  Direksiyon açısını gösterir.
- $\psi$  Yaw açısını gösterir.
- $\alpha$  Kayma açısını gösterir.

## İndisler

- Birinci türev
- İkinci türev

## KISALTMA LİSTESİ

ROS	Robot Operating System
ESC	Electronic Speed Controller
SAE	Society of Automotive Engineers
LiDAR	Laser Imaging Detection and Ranging
FSAE	Formula Society of Automatic Engineers
SCADA	Supervisory Control And Data Acquisition
SLAM	Simultaneous localization and mapping
TÜİK	Türkiye İstatistik Kurumu
BMS	Battery Management System
DC	Direct Current
EKF	Extended Kalman Filter
GPS	Global Positioning System
IMU	Inertial Measurement Unit
PID	Proportional Integral Derivative
CNN	Convolutional Neural Network
FET	Field Effect Transistör
YOLO	You Only Look Once
ECU	Electronic Control Unit
RRT	Rapidly Exploring Random Trees Algorithm
MPC	Model Predictive Control
DC	Doğru Akım
IEEE	Elektrik ve Elektronik Mühendisleri Enstitüsü
COG	Center of Gravity
2B/2D	İki Boyutlu
3B/3D	Üç Boyutlu
GPU	Grafik İşlemci Birimi
ID	Kimlik Numarası
CPU	İşlem Birimi
RAM	Erişimli Hafıza

## ŞEKİL LİSTESİ

Şekil 2-1 Paydaş Beklentileri .....	14
Şekil 3-1-ROS Mimarisi.....	17
Şekil 3-2 Bisiklet Modeli.....	19
Şekil 3-3 RRT Algoritması ile Yolunu Bulan Bir Araç .....	22
Şekil 3-4 RRT Yaparken Çıkarılan Harita .....	23
Şekil 3-5 RRT Yaparken Çıkarılan Haritaya Devam Etmesi.....	23
Şekil 3-6 Pure Pursuit.....	24
Şekil 3-7 Saf Takip Geometrisi .....	25
Şekil 3-8 Tekerlek Açısı Geometrisi .....	26
Şekil 3-9 Bakış Mesafesi Karşılaştırması.....	27
Şekil 4-1 Sistem Şeması .....	28
Şekil 4-2 NVIDIA TX2 .....	29
Şekil 4-3 RPLidar S1 .....	30
Şekil 4-4 ZED Stereo Camera .....	31
Şekil 4-5 Flipsky FSESC 6.6 VESC 6.....	31
Şekil 4-6 SparkFun 9DoF Razor IMU M0 .....	32
Şekil 4-7 Sensörlerden gelen verilerin işleniş şeması .....	32
Şekil 4-8 LiDAR tabanlı algılama .....	34
Şekil 4-9 Konilerin araç bakışından ve yukarıdan görünümü .....	35
Şekil 4-10 7 noktalı koni .....	36
Şekil 4-11 Stereo kamera ile koni tanımlama ve pozisyonlandırma .....	36
Şekil 4-12 RRT .....	37
Şekil 5-1 Aracın Rviz Simülasyonundan Görüntüsü.....	39
Şekil 5-2 Aracın Pistinin Uzaktan Görünümü .....	39
Şekil 5-3 Pistin MATLAB Üzerinden Oluşturulması .....	40
Şekil 5-4 Pistin MATLAB Üzerinden Oluşturulması .....	41
Şekil 5-5 Haritalandırmanın MATLAB Üzerinden Gösterimi.....	42
Şekil 5-6 Aracın MATLAB Üzerinden Haritasını Oluşturma.....	43
Şekil 5-7 Simülasyonun Adımları .....	44
Şekil 5-8 Aracın 4 m/s'deki Simülasyon Görüntüsü.....	45
Şekil 5-9 Aracın 4 m/s'deki Hız-Zaman Grafiği.....	46
Şekil 5-10 Aracın 4 m/s'deki Merkez Çizgisine Göre Hata Grafiği .....	46
Şekil 5-11 Aracın 4 m/s'deki Direksiyon Açısı Grafiği .....	46
Şekil 5-12 Aracın 17 m/s'deki Simülasyon Görüntüsü.....	47
Şekil 5-13 Aracın 17 m/s'deki Hız-Zaman Grafiği.....	48
Şekil 5-14 Aracın 17m/s'deki Merkez Çizgisine Göre Hata Grafiği .....	48
Şekil 5-15 Aracın 17m/s'deki Direksiyon Açısı Grafiği.....	49
Şekil 5-16 Aracın Hızlı Tur Tamamlama Süreleri .....	49
Şekil 5-17 Look Ahead Proportion Grafiği .....	50

## **TABLO LİSTESİ**

Tablo 1 Teknik Gereksinim Spesifikasyonları Tablosu .....	15
Tablo 2 Çalışma Takvimi .....	51
Tablo 3 Malzeme Listesi .....	52



## **TEŞEKKÜR**

Bitirme Çalışması için hazırladığımız projemizde bize yardımcı olarak yol gösteren, maddi ve manevi desteğini esirgemeyen proje ve tez danışmanımız Sayın Doç. Dr. Mehmet Selçuk Arslan'a, her türlü desteklerini esirgemeyen ailelerimize teşekkürlerimizi sunarız.

Alperen Şentürk  
Mahmut Reyhani  
Zeynep Esra İşler

## ÖZET

Otonom sistemler günümüzde gittikçe yaygınlaşmaktadır. Endüstriyel alanlardan günlük hayatımıza kadar otonom sistemlerin yaygınlaştığı noktada ulaşım araçları teknolojilerinin etkilenmesi de kaçınılmazdır. Dünya çapında hali hazırda birçok otonom araç projesi tamamlanmış veya sürmektedir. Yine de gelişiminin henüz başlarında olan bu otonom araç sistemlerinin dünyanın her yerinde görülmesi için önünde uzun bir yol vardır. Otonom sistemlerin tam otonom olarak yol araçlarında kullanımına hem yasal hem de teknik endişelerle henüz birçok ülkede başlanmamıştır. Fakat yine birçok ülkede yarı otonom araç kullanımı yollarda görülmektedir. Otonom araç sistemlerinin yarış araçlarında kullanılma fikri de yeni değildir ancak henüz yeterince gelişmiş ve geniş kitlelere hitap etmeye başlamamıştır. Bu projede yapılan otonom bir yarış aracının bilgisayar ortamında gerçekleşmesidir. Bu yarış aracının mekanik tasarım, donanım ve yazılım özelliklerini ile birlikte Mekatronik Sistem Tasarımı ve Bitirme Çalışması raporlarında belirtilmiştir. Söz konusu proje bilgisayar ortamında gerçekleşmiş olsa dahi bu raporda projenin nasıl gerçekleştirildiği; hangi yöntemlerin kullanıldığı, malzeme seçimi ve teknik gereksinimleri gibi konulara da yer verilmiştir. Yazılım hazırlanırken kullanılan yaklaşım ile daha rekabetçi bir araç oluşturulması hedeflenmiştir. Bu sayede otonom yarış aracı sektörüne adım atabilecek ve ileriye hedefleyen uygulamalara bir temel oluşturabilecek araç projesi hazırlanmıştır.

*Anahtar Kelimeler: Otonom, Yarış, Araç, Pist, Koni, Pure Pursuit, Robot Operating System*

## 1. GİRİŞ

### 1.1 Problem Tanımı

Ulaşım ve taşımacılık için kullanılan kara taşıtlarının otonom hale getirilmesinin amaçlanmasındaki en büyük etken, araç kullanımında insan faktörünün yarattığı problemleri en aza indirmektir. TÜİK verilerine göre günümüzde trafik kazalarının %89.3'ü sürücü hatalarından kaynaklanmaktadır. Bu sebepler arasında; direksyon başında uyuyakalma, alkollü araç kullanımı, aşırı hız ve hatalı sollama yer almaktadır.

Otonom araçların Dünya genelinde en çok tercih edilme sebebi, hızlı, güvenilir ve daha ekonomik olmasıdır. Otomotiv sektöründeki birçok teknolojik gelişmenin yarışmacı doğasından ötürü yarış araçları üzerinde geliştirildiği ve daha sonra binek araçlara entegre edildiği gibi bu projenin de yarış aracı olarak tasarılanmasındaki amaç rekabetçi olabilmektir.

Uzun mesafe araç kullanımında ise insani ihtiyaçlar için bir insanın aralıksız araç kullanım süresi ilgili kanunlarca sınırlandırılmıştır. [5]

### 1.2 Projenin Motivasyonu

Projemizi seçerken, otonom teknolojilerin katkılarını günümüz problemlere çözüm sunmasının yanında eğlence sektöründe de kendine yer bulması olarak iki ayrı başlıkta inceledik. Otonom teknolojiler günlük hayatımıza entegre olmaya başladıkça birçok spor ve eğlence sektörünün otonom sistemlere doğru yönelimleri olduğunu görmekteyiz.

Örneğin; 1997 yılında IBM tarafından geliştirilen DeepBlue adında bir satranç motoru, dünya satranç şampiyonu Garry Kasparov'u mağlup etmiş ve bu maç, bir satranç motorunun bir dünya şampiyonunu yendiği ilk maç olmuştur. Günümüzde ise farklı şirketler tarafından geliştirilen satranç motorlarının aralarında gerçekleştirdikleri özel dünya şampiyonası düzenlenmektedir. Bizim otonom yarış aracı tasarıımı projemizin bahsettiğimiz örnekle ilişkisi şu şekildedir: Formula 1 yarışları tüm dünya tarafından ilgiyle takip edilmekte ve bu araçların yapılmasında kullanılan teknolojiler büyük ölçüde yatırımlar almaktadır. Bu yarışlarda da tipki satrançörneğinde olduğu gibi otonom araçların entegre olacağını ve halihazırda var olan otonom araç yarışlarının yakın gelecekte çok daha büyük kitlelere hitap edeceğini inanıyoruz. Biz de proje ekibi olarak, bu sektörün bir parçası olmak ve tasarlayacağımız araç algoritması ile katkıda bulunmak istiyoruz.

## İnsan Faktörünü En Aza İndirgeme

Projenin insan kaynaklı oluşan hataları en aza indirmesini, daha güvenli, daha az hata oranı sağlayan ve daha hızlı bir teknoloji üretmek en önemli motivasyon kaynaklarından biridir. Dünya'nın otonom araçlara yönelmesi, projemizi seçerken bizi de otonom teknolojileri kullanmaya yönelik bir çalışma yapmaya yöneltti.

## Otonom Araçlara Rekabetçi Ortamda Fırsat Verme

Otonom teknolojiler günlük hayatımıza entegre olmaya başladıkça birçok spor ve eğlence sektörünün otonom sistemlere doğru yönelimleri olduğunu görmekteyiz. Örnek olarak SAE'nin düzenlediği Formula Student Driverless yarışması da öğrenci kulüpleri arasında otonom araçlarının yarıştırılması şeklinde gerçekleştirilen bir yarışmadır. Biz de proje ekibi olarak, tasarladığımız araç yazılımı ile bu yarışmada başta okulumuz kulüplerine ve bizim gibi otonom araçlara ilgi duyan mühendis adaylarına katkı sağlamayı arzuluyoruz.

### 1.3 Projenin Amacı

Gelişiminin henüz başlarında olan otonom araç sektörü yenilikçi teknolojilerin bulunduğu, büyük ölçüde yatırımların yapıldığı her geçen gün rekabetin arttığı bir sektördür. Yalnızca fiyat odaklı bir rekabet olmamakla birlikte büyük üretici firmaların, potansiyel müşterileri çekmek için yatırımlarını ve Ar-Ge çalışmalarını bu sektörde yönelttiğini görebiliyoruz. Daha önce bahsettiğimiz gibi fiyat tek parametre olmaktan çıkmış, günümüz bakış açısından insan hayatının öneminin artması, konforun ve verimliliğin daha önemli parametreler haline gelmiş olması bu alanda yapılan çalışmaları büyük ölçüde tetiklemiştir. Büyük ölçüde insan odaklı olan trafik kazalarının önlenmesinin en etkili yolu otonom araçların kullanımının artması ve iyileştirilmesidir. Bu projeyi seçerken, ileride yapılacak olan otonom araçlar için yalnızca rekabetçi bir yapı oluşturmak amaçlanmaktadır. Böylece, trafikte insan hatasına bağlı kaza ve trafik yoğunluklarının düşürülmesi mümkün kılınabilir.

Otomotiv sektöründeki birçok teknolojik gelişmenin, yarışmacı doğasından ötürü yarış araçları üzerinde geliştirildiği ve daha sonra binek araçlara entegre edildiği için SAE tarafından düzenlenen Formula Student Driverless yarışmasının teması üzerinde durularak, bu

yarışmanın zorlayıcılığına uygun bir pist hazırlanmayı ve bu pisti en hızlı şekilde tamamlamayı ve böylece yarışmaya katılacak takımların üzerinde test gerçekleştirebileceği bir araç oluşturmayı amaçlıyoruz.

#### 1.4 Projenin Hedefleri

- Araç, taşınabilir ve maliyeti düşük olması adına 1:10 ölçüğünde tasarılanmalıdır.
- Araç, sürücü olmaksızın çevresini algılayabilmeli yani haritalandırma yapıp kendisini bu haritada başarı ile konumlandırmalıdır.
- Araç, pisti ilk turunda haritalandırmalı ve kalan 9 turu en hızlı şekilde tamamlamalıdır.
- Araç, pistteki turlarını hızlı bir şekilde tamamlayabilmek için yol planlaması yapmalıdır.
- Araç, planladığı yolda etrafındaki konileri devirmeden, 60 kilometre/saat maksimum anlık hızza ulaşabilmelidir.

#### 1.5 Literatür Taraması

Otonom yarış araçları ile ilgili incelenen çalışmalar şu şekildedir:

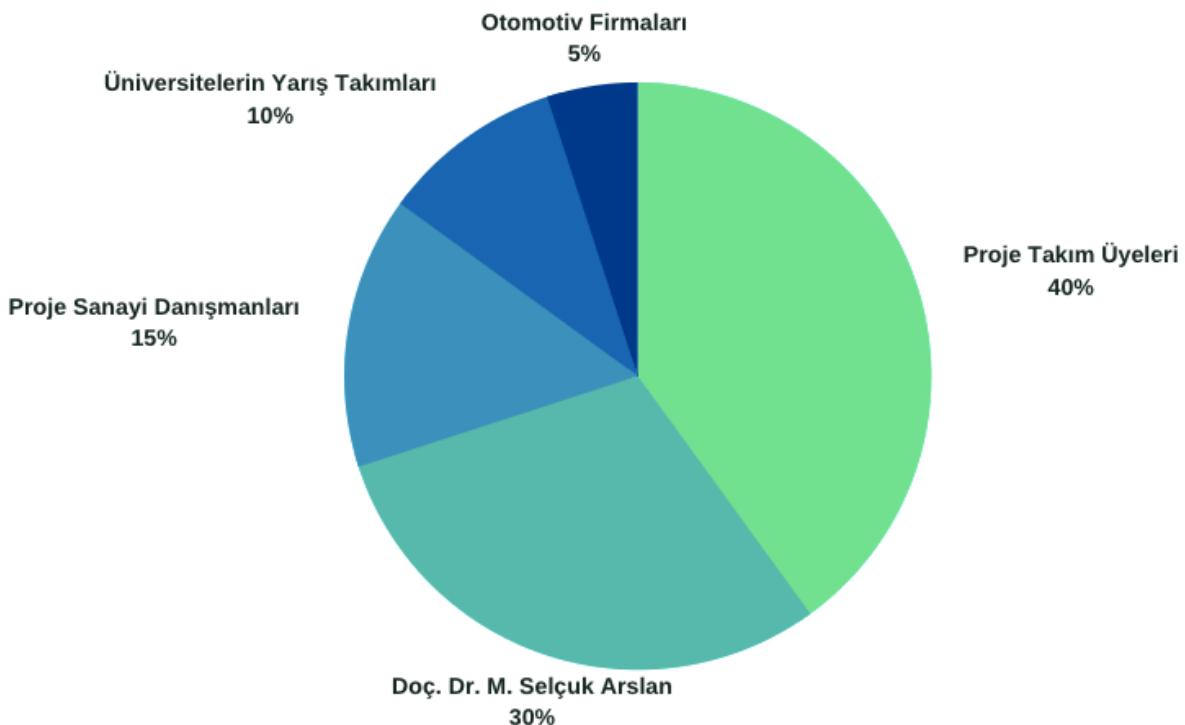
- “AMZ Driverless: The Full Autonomous Racing System” isimli çalışmada otonom tam ölçekli bir aracın LiDAR ve kamera kullanarak yarış pistindeki sarı ve mavi renkli konilerden veriler elde etmişlerdir. Elde edilen verileri SLAM aracılığıyla optimize edip aracın haritalandırmasını tamamlanmıştır. Bu çalışmada algı, planlama ve kontrol olarak 3'e ayrılmış ve her bir bölüm kendi için detaylandırılarak aracın en optimal şekilde tasarlanması hedeflenmiştir. [2]
- “Teaching Autonomous Systems at 1/10th-scale: Design of the F1/10 Racecar, Simulators and Curriculum ”, 1/10 ölçekli otonom yarış aracı tasarımının anlatıldığı sistem integrasyonunda elektornik hız kontrolcüsü, SLAM ve Nvidia Jetson TX1/TX2 kullanılmıştır. Hipotezleri, 1/10 ölçekli otonom araçların, gerçek otonom araçların ölçeklendirilmiş dinamiklerini, algılama yöntemlerini, karar vermesini ve risklerini yeterince yakaladığını ancak otonom sistemlerin temellerini öğretmek için güvenli ve erişilebilir bir platform olması amaçlanmıştır. [3]
- “Design, build, and test drive a FSAE electric vehicle” isimli çalışmada; mekanik dizayn, aktarma organları, SCADA ve BMS olarak 4 alt başlıkta otonom bir yarış aracı tasarlanması, test edilmesi ve sistem integrasyonu oluşturulması amaçlanmıştır.[4]

## 1.6 Kabuller ve Varsayımlar

- (i) Elde edilen verilerden gelen gürültü yok varsayılmaktadır.
- (ii) Çalışmamızda servo motorun başarılı bir şekilde çalıştığı kabul edilmektedir.
- (iii) Araç düz bir yüzey üzerinde hareket ediyor,
- (iv) Yük aktarımı ihmali edilebilir,
- (v) Kombine kayma ihmali edilebilir
- (vi) Boyuna aktarma organları kuvvetleri ağırlık merkezine etki eder.

## 2. GEREKSİNİM SPESİFİKASYONLARI

### 2.1 Paydaş Beklentileri



**Şekil 2-1 Paydaş Beklentileri**

Şekil 2.1'de dağılımı gösterilen proje paydaşları:

- Proje Takım Üyeleri
- Doç. Dr. M. Selçuk Arslan
- Üniversitelerin Yarış Takımları
- Proje Sanayi Danışmanları
- Otomotiv Firmaları şeklinde belirlenmiştir.

Paydaşların bekłentileri ise;

1. Çağdaş mekatronik tanımına uyması
2. Aracın pist öğelerini tanımması
3. Aracın pisti en kısa sürede tamamlaması
4. Aracın taşınabilir olması
5. Farklı sensörlerden veri alınması
6. Maliyetin düşük olması
7. Tüm projenin 2022 Ocak ayında teslim edilmesi şeklindedir..

## 2.2 Teknik Gereksinim Spesifikasyonları

**Tablo 1 Teknik Gereksinim Spesifikasyonları Tablosu**

Paydaş Beklentileri	Teknik Gereksinimler	Açıklama
4,3	1. Araç 25km/h maksimum hızı sahip olmalıdır.	Araç için seçilen DC motor bu ihtiyacı karşılayabilmektedir.
4	2. Araç kütlesinin 6 kilogramı geçmemesi gerekmektedir.	Tüm donanım kütleleri toplandığında 5-6kg kütleye sahip bir araç ortaya çıkacaktır.
1,2,4	3. Araç 1:10 oranında tasarılanmalıdır.	Ölçeklendirilmiş şasi tasarımına gidilmiştir.
1,3	4. Maksimum 50,000 rpm, 3500 kV, 2.5 N.m tork değerinde firçasız doğru akım motor kullanılmalıdır.	Aracın hız hedefini karşılaması için talebe göre motor seçiminde bulunulmuştur.
1,3,5	5. İşlemci çift çekirdek 2.3GHz üzeri işlemci gerekmektedir.	Talebe uygun işlemci seçilmiştir.
1,3	6. Manevra kabiliyeti için yüksek keskinlikte servo motor kullanılmalıdır.	Seçilen servo motor yüksek tork ve hassasiyete sahiptir.
1,2,5	7. Minimum 20m menzilli, minimum örnekleme hızı 8kHz, 2B nokta bulutu oluşturma kabiliyetine sahip LiDAR seçilmelidir.	40m menzilli, kısa mesafe LiDAR seçilmiştir.
1,2,5	8. 100 Hz poz güncelleme oranı, 4.416x1.242 piksel, gerçek zamanlı derimnlik tabanlı görsel odometri ve SLAM kabiliyetine sahip kamera gerekmektedir.	Gerekli tüm kabiliyetlere sahip çift lensli, stereo derinlik kamerası seçilmiştir.

### 3. TEORİK ALTYAPI BİLGİLERİ VE YÖNTEMLER

Bitirme Çalışması boyunca, yapılan tüm teorik hesaplamaları ve yaklaşımın hangi kaynaklar kullanılarak tamamlandığı, bu yaklaşımı tamamlamak için kullanılan yöntemlerin anlatıldığı bölümüdür.

#### 3.1 Geliştirme Ortamları

Tüm geliştirme ortamları Bitirme Çalışması kapsamında kullanılmıştır. Çalışmaların sonucunda elde edilen çıktılar 5.TASARIM DOĞRULAMA bölümünde detaylarıyla aktarılmıştır.

##### 3.1.1 MATLAB

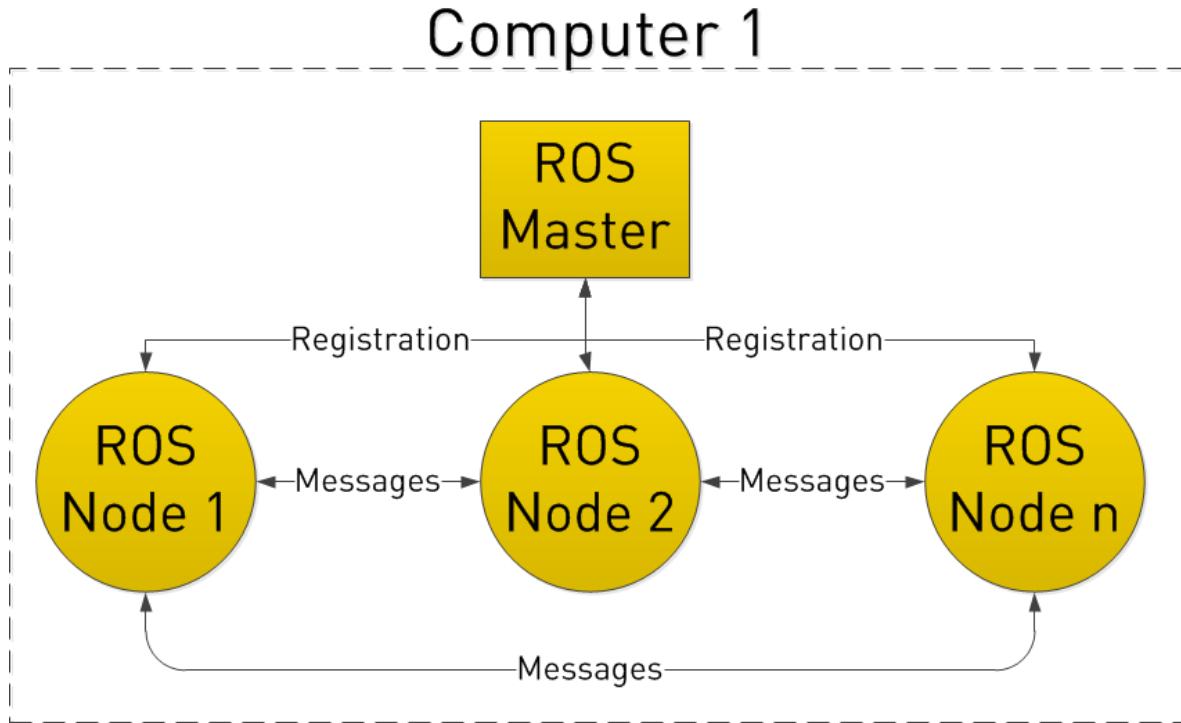
MATLAB (matris laboratuvarı), dördüncü nesil bir programlama dili ve çok paradigmal sayısal hesaplama yazılımıdır. MathWorks, tescilli bir programlama dili olan MATLAB'ı oluşturur. Kullanıcı, MATLAB kullanarak matrisleri manipüle edebilir, fonksiyonları ve verileri çizebilir, algoritmalar uygulayabilir, kullanıcı arayüzleri oluşturabilir ve C, C++, Java ve Fortran dahil olmak üzere diğer dillerde yazılmış programlarla iletişim kurabilir.

Bitirme çalışması kapsamında MATLAB kütüphanelerinden ve geometrik görselleştirici paketlerinden faydalanyanmıştır.

##### 3.1.2 ROS

Robot İşletim Sistemi (ROS), robot yazılım geliştiricileri için bir yazılım platformudur. ROS HAL (Düşük seviyeli cihaz kontrolü, en çok kullanılan fonksiyonlar, veri paylaşımı ve paket yönetimi) gibi ortak bir işletim sistemi sunar. ROS tabanlı devam eden süreçleri temsil etmek için mimari grafikler kullanılır. Bu mimari grafiklerde çok parçalı sensörler, kontrol mekanizmaları, konum, aktuatör ve diğer süreçlerden gelen veriler görüntülenir. Robot kontrolünün düşük bir gecikme süresi vardır, bu nedenle gerçek zamanlı bir işletim sistemi değildir. ROS ise eşzamanlı kodlarla birlikte kullanılabilir.

ROS yapısını biraz daha anlamak için içerisindeki alt fonksiyonların yürütülmesini sağlayan ilişki ağını tanımk önemlidir.



**Şekil 3-1-ROS Mimarisi**

**Düğüm:** Bir ROS düğümü, görevleri ve hesaplamaları gerçekleştirmekten sorumlu bir süreçtir. Konular veya diğer daha karmaşık araçlar kullanılarak birbirleriyle birleştirilebilirler.

**Konu:** Konular, düğümler arasında tek yönlü çalışan bilgi kanalları olarak tanımlanabilir. Bu, tek yönlü bir iş akışı olarak kabul edilir, çünkü düğümler konulara abone olabilir, ancak bir konu hangi düğümlerin kendisine abone olduğunu bilemez.

**Yönetici:** ROS yöneticisi, kalan düğümlere bir ad ve kayıt sağlayan bir hizmettir. Ana işlevi, tek tek düğümleri, birbirlerinin yerini bulabilmeleri ve birbirleri arasında iletişim kurabilmeleri için etkinleştirilmektir.

**Paket:** Paketler, ROS organizasyonunun özüdür. Bu paketlerde, bir robotik uygulaması oluşturmak için düğümler, kitaplıklar, veri kümeleri veya faydalı bileşenler bulabilirsiniz.

**Yığın:** Bir ROS yiğini, hep birlikte bazı işlevler sağlayan bir dizi düğümdür. Geliştirilecek işlevsellik çok karmaşık olduğunda görevleri düğümler arasında bölmek için yararlı olabilir.

ROS kurulumu için Linux işletim sisteminin olması yeterlidir. ROS kullanım alanları ve kullanılan işletim sistemlerine göre farklı dağıtımlara sahiptir:

- Ubuntu 16.04(LTS)+ROS Kinetic (Kullanılan)
- Ubuntu 18.04(LTS)+ROS Melodic
- Ubuntu 20.04(LTS)+ROS Noetic

ROS içerisinde dahili olarak çalışabilen RVIZ ve Gazebo ortamları, ROS üzerinde inşa edilen robotun görselleştirilmesi ve testlerinin yapılması için kullanılan faydalı programlardır. Gazebo, ücretsiz ve açık kaynak kodlu bir 3D robot simülatördür. Donanımlı tasarımlı ile gerçekçi sahneler oluşturabilmektedir. Robotların oluşturulmasını, algoritma testlerini, regresyon testlerini ve yapay zeka sistemlerinin eğitimini sağlar. RVIZ, ROS uygulamaları için bir 3d görselleştirme aracıdır. Robot modelinizin bir görünümünü sağlar, robot sensörlerinden sensör bilgilerini yakalar ve yakalanan verileri yeniden yürütür. Resimler ve nokta bulutları dahil olmak üzere kameradan, lazerlerden, 3D ve 2D cihazlardan gelen verileri görüntüleyebilir.

Bitirme çalışması kapsamında iyileştirilmiş simülasyon çalışmaları dahil pek çok çalışma RVIZ ve Gazebo ortamlarında oluşturulmuş ve test edilmiştir.

### 3.1.3 CARLA Simülatörü

CARLA, otonom sürüş araştırması için açık kaynaklı bir simülatördür. CARLA, sıfırdan otonom sürüş sistemlerinin geliştirilmesini, eğitimini ve onaylanması desteklemek için geliştirilmiştir. CARLA, açık kaynak kod ve protokollerine ek olarak, bu amaçla oluşturulmuş ve serbestçe kullanılabilen açık dijital varlıklar (kentsel yerleşimler, binalar, araçlar) sağlar. Simülasyon platformu, sensör takımlarının ve çevresel koşulların esnek özelliklerini destekler.

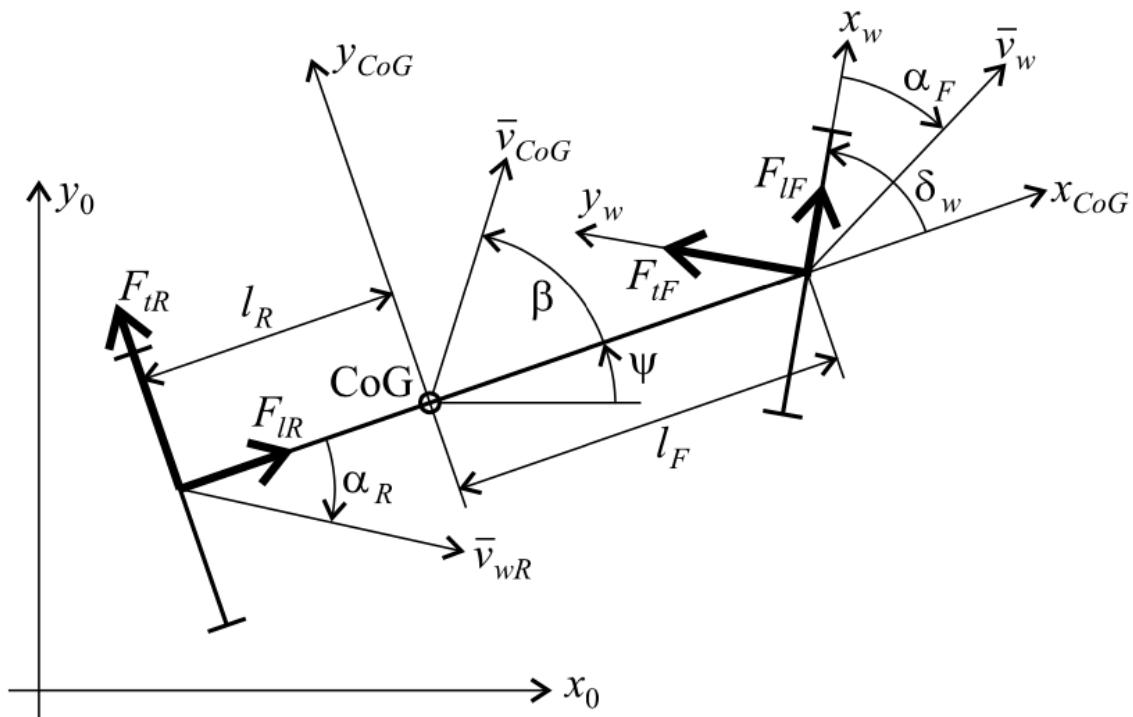
### 3.2 Lokalizasyon ve Haritalandırma

Eşzamanlı yerelleştirme ve haritalama (SLAM), bilinmeyen bir ortamın haritasını oluştururken veya güncellerken aynı anda bir aracının içindeki konumunu takip eden hesaplama sorunudur. Bu başlangıçta bir tavuk-yumurta sorunu gibi görünse de, belirli ortamlar için en azından yaklaşık olarak izlenebilir sürede çözmek için bilinen birkaç algoritma vardır. Popüler yaklaşık çözüm yöntemleri arasında parçalıkfiltresi, genişletilmiş Kalmanfiltresi, kovaryans kesesi ve GraphSLAM bulunur. SLAM algoritmaları, hesaplamalı geometri ve bilgisayarla görme kavramlarına dayanır ve sanal gerçeklik veya artırılmış gerçeklik için robot navigasyonu, robotik haritalama ve odometride kullanılır.

SLAM algoritmaları mevcut kaynaklara göre uyarlanmıştır, dolayısıyla mükemmelliği değil, operasyonel uyumluluğu hedefler. Yayınlanmış yaklaşımlar sürücüsüz arabalarda, insansız hava araçlarında, otonom sualtı araçlarında, gezegen gezicilerinde, daha yeni yerli robotlarda ve hatta insan vücutunun içinde kullanılmaktadır.

### 3.3 Aracın Kinematik Bisiklet Modeli

Kinematik bisiklet modeli, iki arka tekerleğin ve iki ön tekerlein sırasıyla bir arka tekerlek ve ön tekerlekte toplandığı kinematik dört tekerlekli modelin basitleştirilmiş halidir.



**Şekil 3-2 Bisiklet Modeli**

$x_{COG}$  arabanın uzunlamasına aksı olsun,  $y_{COG}$  sola doğru yönlendirilir ve  $z_{COG}$  yukarı doğru yönlendirilir. Tekerleklerin ağırlık merkezine (CoG) olan mesafesini sırasıyla  $l_F$  ve  $l_R$  olarak gösterilmiştir. F (front) arabanın ön tekerleklerini ifade ederken, R (rear) arabanın arka tekerleklerini ifade etmektedir. Arka tekerlek hızı  $v_{wR}$ 'dir ve  $x_{COG}$  eksenine olan açı  $\alpha_R$ 'dır.  $\beta$ ,  $\alpha_F$ ,  $\alpha_R$  açıları genellikle sırasıyla araç gövdesi yan kayma açısı, ön lastik yan kayma açısı ve arka lastik yan kayma açısı olarak adlandırılır.

Ön tekerlek, arabanın uzunlamasına  $x_{COG}$  eksenine göre sola yönlendirilirse,  $v_{COG}$  ve  $v_{WF}$  vektörleri  $x_{COG}$  ekseninin sol tarafında,  $v_{wR}$  hız vektörü ise  $x_{COG}$  ekseninin sağ tarafındadır. Bu durumda tekerlein dönüş açısı ve kayma kayma açısı pozitif olarak kabul edilir.

Ön tekerlein koordinat sisteminin orijine etki eden kuvvetler boyuna kuvvet  $F$  ve enine kuvvet  $F_{tF}$ 'dir. Enine bileşenin

$$F_{tF} = c_F + \alpha_F$$

olduğu varsayılmıştır, burada  $c_F$  sabittir. Benzer şekilde, arka tekerlein çerçevesinin orijininde etki eden kuvvetler, boyuna kuvvet  $F_{tR}$  ve enine kuvvet  $F_{tR}$ 'dir.

$$F_{tR} = c_R + \alpha_R$$

burada  $c_R$  sabittir.

Ön tekerlein hızı, arabanın açısal hızına ( $\dot{\psi}$ ), yani dairesel hareketteki noktanın yarıçapındaki hızının etkisiyle artan COG hızına eşittir. Hız vektörünün mutlak değeri  $v_{COG}$  ile gösterilecektir. Arka tekerleklerdeki hız değerinin denklemi,

$$\begin{aligned} v_{WF} &= \sin(\delta_{W-} \alpha_F) = l_F \dot{\psi} + v_{COG} \sin(\beta) \\ v_{WF} &= \cos(\delta_{W-} \alpha_F) = v_{COG} \cos(\beta) \end{aligned}$$

şeklinde gösterilir. Eğer bu iki denklemi birbirine bölersek  $\tan(\delta_{W-} \alpha_F)$ ,

$$\tan(\delta_{W-} \alpha_F) = \frac{l_F \dot{\psi} + v_{COG} \sin(\beta)}{v_{COG} \cos(\beta)}$$

denklemi elde edilir. Arka tekerleklerde olduğu gibi ön tekerleklerin hızı da aynı şekilde elde edilebilir.

$$\begin{aligned} v_{WR} &= \sin(\alpha_R) = l_R \dot{\psi} - v_{COG} \sin(\beta) \\ v_{WR} &= \cos(\alpha_R) = v_{COG} \cos(\beta) \end{aligned}$$

Eğer bu iki denklemi birbirine bölersek  $\tan(\alpha_R)$ ,

$$\tan(\alpha_R) = \frac{l_R \dot{\psi} - v_{COG} \sin(\beta)}{v_{COG} \cos(\beta)}$$

denklemi elde edilir. Denklemleri basitleştirmek için bazı kabuller yapılması gerekmektedir.  $\tan(\delta_W - \alpha_F) \approx \delta_W - \alpha_F$ ,  $\tan(\alpha_R) \approx \alpha_R$ ,  $\sin(\beta) = S_\beta \approx \beta$  ve  $\cos(\beta) = C_\beta \approx 1$  şeklinde kabul edilirse,

$$\alpha_F = \delta_W - \beta - \frac{l_F \dot{\psi}}{v_{COG}}, \quad \alpha_R = -\beta + \frac{l_R \dot{\psi}}{v_{COG}}$$

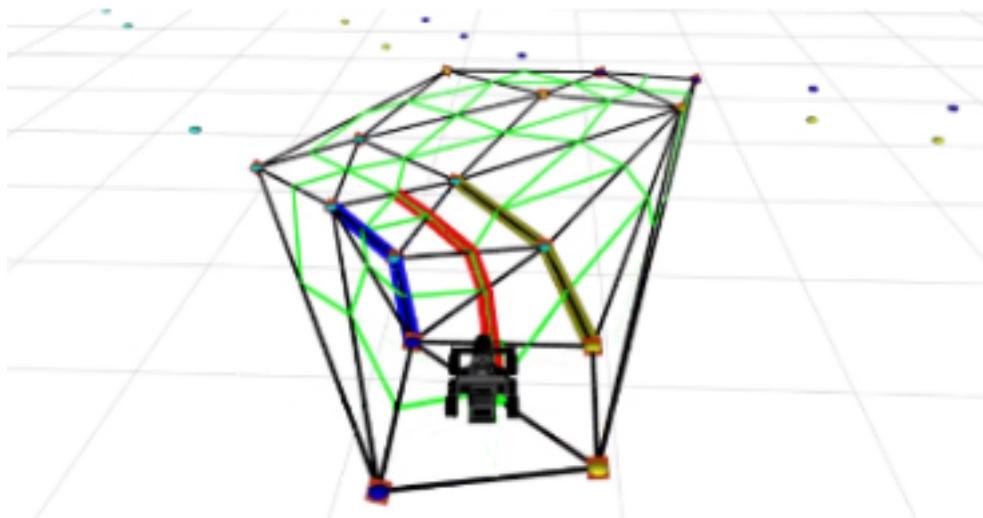
$$\bar{v}_{COG} = (C_\beta S_\beta 0)^T v_{COG}$$

$$\bar{\alpha}_{COG} = v_{COG} \dot{v}_{COG} + \omega \times \bar{v}_{COG}$$

$$\bar{\alpha}_{COG} = \begin{bmatrix} C_\beta & -v_{COG} S_\beta \\ S_\beta & v_{COG} C_\beta \\ 0 & 0 \end{bmatrix} \begin{pmatrix} \dot{v}_G \\ \dot{\beta} \end{pmatrix} + \begin{pmatrix} -S_\beta \\ C_\beta \\ 0 \end{pmatrix} \dot{\psi} v_{COG}$$

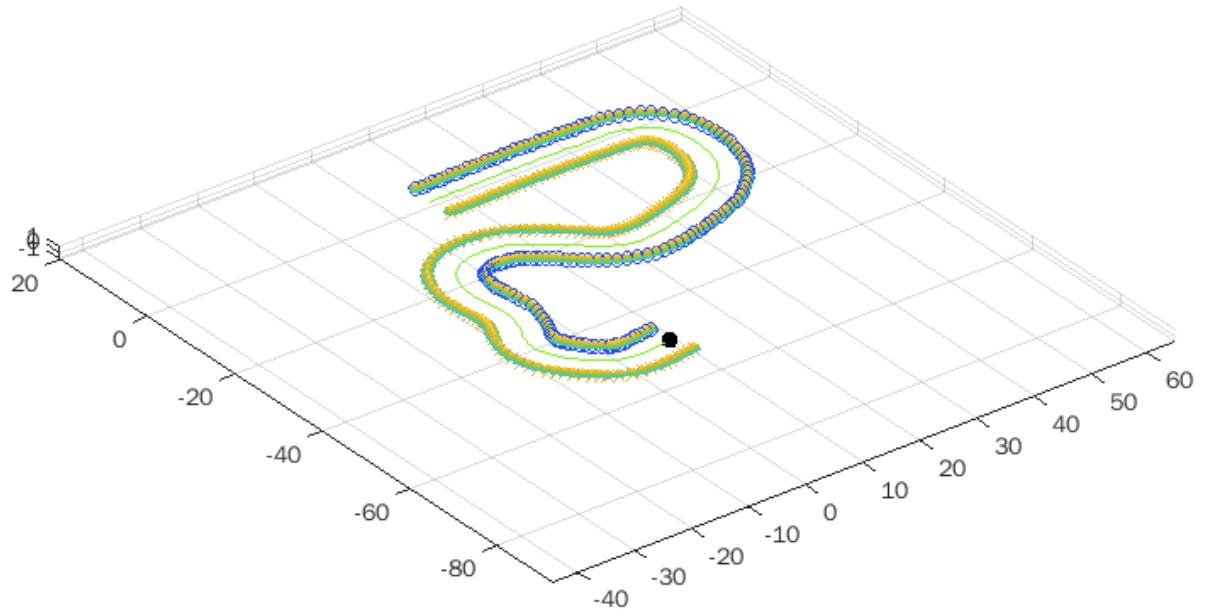
Bir aracın lineer olmayan modeli bu şekilde çıkarılır. [5]

### 3.4 RRT Algoritması

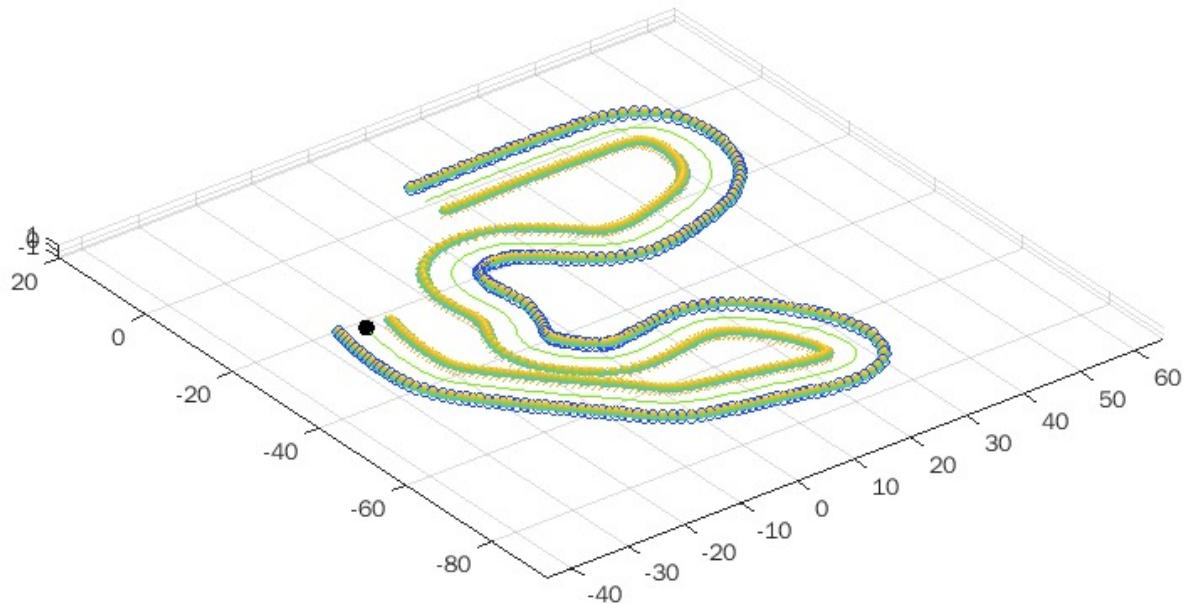


**Şekil 3-3 RRT Algoritması ile Yolunu Bulan Bir Araç**

RRT(Rapidly-exploring Random Tree) yani Hızlı-Keşfedilen Gelişigüzel Ağacılar algoritmasıdır. Bu algoritma S. LaValle ve J. Kuffner isimli iki bilim insanı tarafından 1998'de icat edilmiştir. RRT algoritması, geleneksel ve sezgisel yaklaşımın çok fazla bilgi işlem gücüne ihtiyaç duyduğu ve ortam verilerinin bulunmadığı durumlarda mobil robotlar için olasılıksal yol planlama çözümleri sağladığı için son yıllarda popülerlik kazanmıştır. Bu yaklaşım, tüm bölgeyi hızlı bir şekilde tarar ve özellikle hareket alanının geniş olduğu ve diğer yöntemlerin zor olduğu durumlarda ağaç yapısı şeklinde bir haritalama üzerinde en uygun yoldan gitme stratejisi ile hedefe ulaşmayı sağlar. Bu stratejinin amacı, baştan rastgele noktalar seçerek dallar oluşturmak ve daha sonra bu dallara dayalı hedefe ulaşmak için en uygun yolu oluşturmaktır. Bu amaçla çeşitli yerlerde rastgele konumlar seçer, ancak seçilenlerin dışında yeni bir nokta seçip hedefe yakınsama işlemi kolaylaştırır. RRT yaklaşımı, geleneksel tek yönlü uygulamaya ek olarak, çift yönlü, çok ağaçlı organize ve farklı şekilde sınırlandırılmış görünülmektedir.

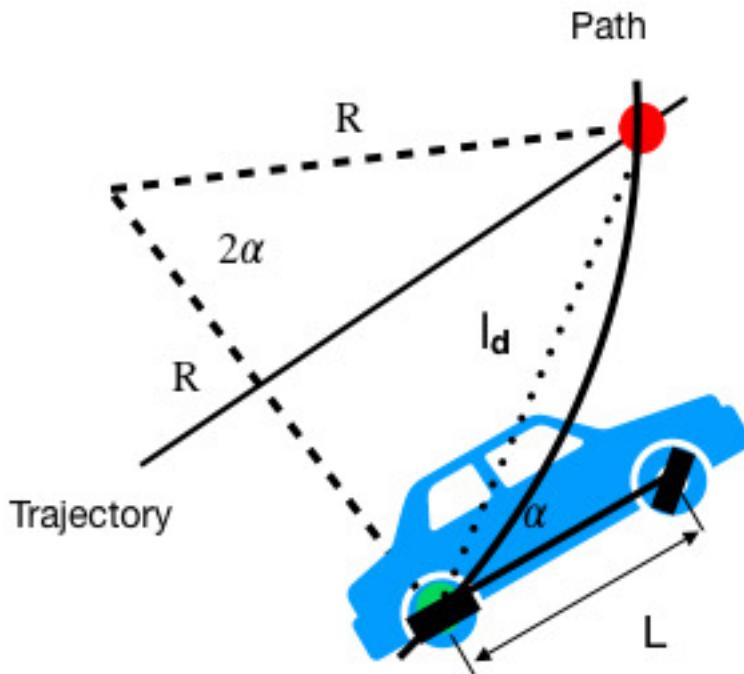


**Şekil 3-4 RRT Yaparken Çıkarılan Harita**



**Şekil 3-5 RRT Yaparken Çıkarılan Haritaya Devam Etmesi**

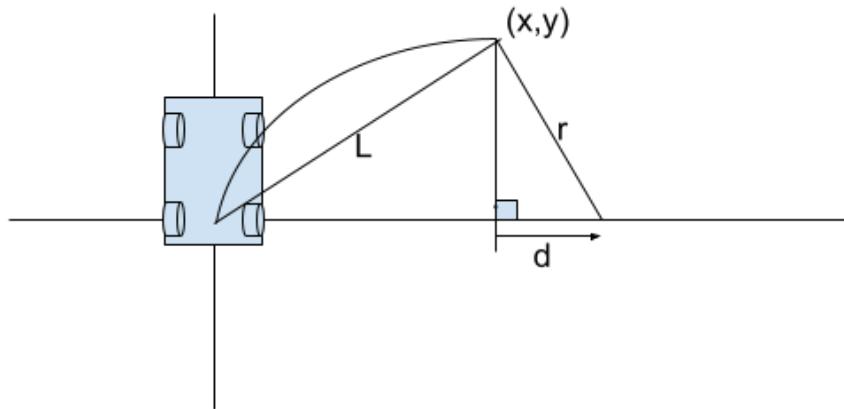
### 3.5 Pure Pursuit



**Şekil 3-6 Pure Pursuit**

Saf Takip (Pure Pursuit) bir aracın hareketi sırasında aracın ilerisinde anlık olarak nokta ve o noktaya doğru eğrilik hesaplayarak aracın hedef konumunu belirleyen bir algoritmadır. Noktanın anlık olarak belirlenmesinin altı çizilmelidir çünkü algoritma sürekli olarak mevcut yol dahilinde önünde bir nokta belirler ve o noktaya doğru hareket eder. Yolun kıvrımlarından dolayı takip noktası yolun durumuna ve aracın hareketine göre anlık olarak değişir.

Hedef noktası, aracımızın arka aksın orta noktasından bir bakış mesafesi(look ahead distance) uzaklıktadır. Daha sonra bahsettiğimiz hedef noktası ile aracın mevcut noktası arasında bu naktaları birleştiren bir yay(eğri) oluşturulur. Yayı oluştururken (yayın)kiriş uzunluğu olarak bakış mesafesi alınır.



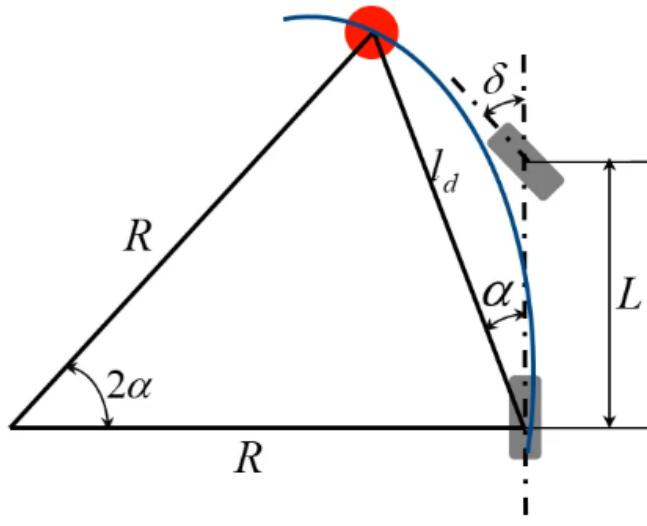
**Şekil 3-7 Saf Takip Geometrisi**

Yukarıdaki şekilde aracın arka aksının orta noktasından koordinat düzlemindeki  $(x,y)$  noktasına(bakış noktası) kadar bir doğru parçası çizilir. Yayın oluşturacağı hayali çemberin x-ekseni ile kesiştiği noktadan  $(x,y)$  noktasına çizilen doğru parçası ‘r’ ile ifade ediliyor bu da çemberin yarıçapına tekabül etmektedir. Burada amaç, o noktaya varabilmek için gerekli olan yayı çizebilmektir. Bu durumda;

$$\begin{aligned}
 x^2 + y^2 &= L^2 \\
 x + d &= r \\
 (r - x)^2 + y^2 &= r^2 \\
 r^2 - 2rx + x^2 + y^2 &= r^2 \\
 2rx &= L^2 \\
 r = \frac{L^2}{2x} &
 \end{aligned}$$

Denklemleri elde edilebilir. Bu, Saf Takip'in basitçe gösterimidir.

Bir de aracın dönüş açısının zamana bağlı fonksiyonunu elde etmek için bir şekil oluşturalım ve denklem kuralım.



**Şekil 3-8 Tekerlek Açısı Geometrisi**

Burada Kinematik Bisiklet Modeline dayanarak dört tekerli aracımız iki tekerlek ile modellenmiştir.

$\delta$  = Tekerleğin araç doğrultusu ile açısı

$L$  = Aracın Kinematik Bisiklet Modeline göre uzunluğu

$I_d$  = Bakış mesafesi

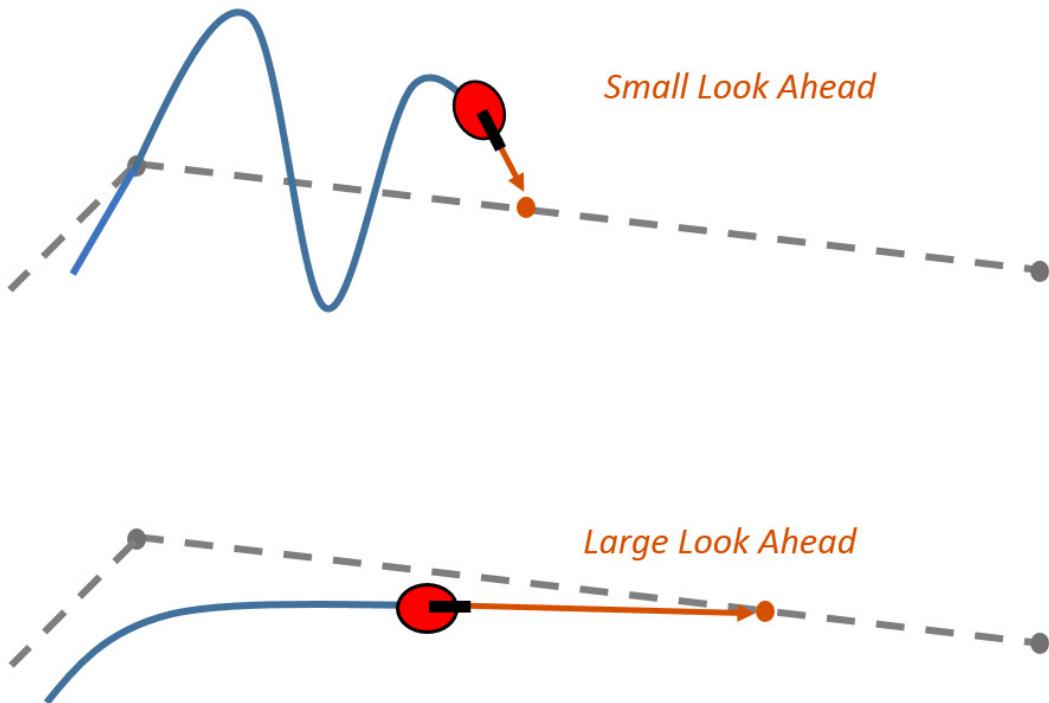
$\alpha$  = Hedeflenen dönüş açısı

$R$  = Oluşturulan yayın yarı çapı

Bu durumda anlık teker açısının denklemini kuralım.

$$\delta(t) = \tan^{-1}\left(\frac{2 \cdot L \cdot \sin \alpha}{I_d}\right)$$

Açı denklemimizi de ifade ettikten sonra Saf Takip için ihtiyacımız olanı elde etmiş bulunuyoruz.

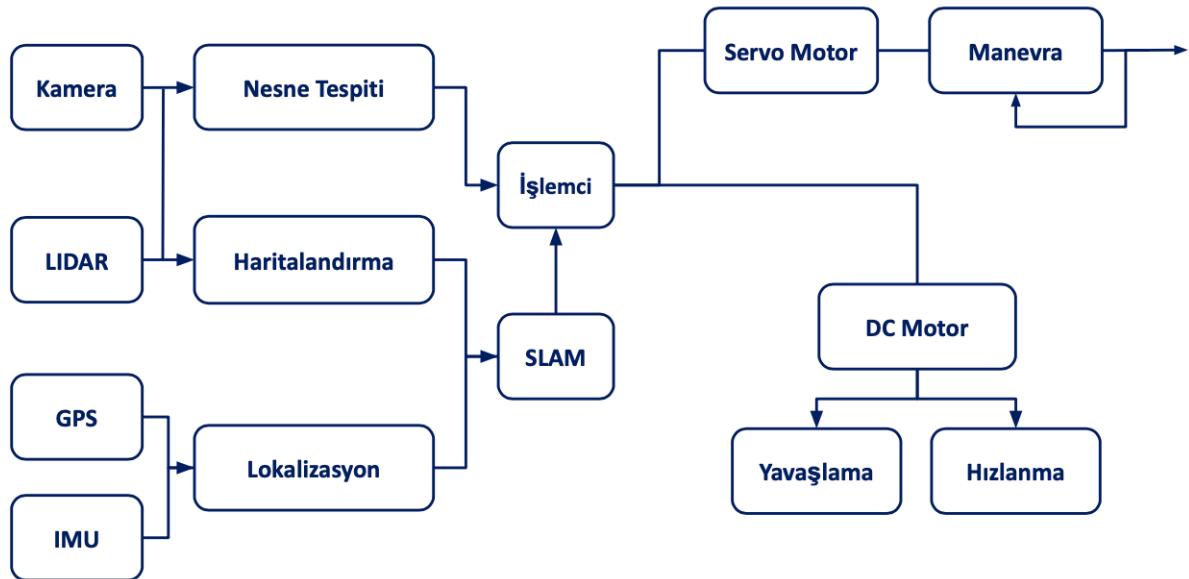


**Şekil 3-9 Bakış Mesafesi Karşılaştırması**

Bakış mesafesinin uzunluğunu artırrarak Saf Takip algoritmasının yol takibindeki yapacağı hatalar düşürülebilir. Bu sayede araç takip edeceği yola dik bir şekilde giriş yapmamış ve daha pürüzsüz bir sürüş gerçekleştirilmiş olur. Bakış mesafesinin çok yüksek tutulması durumunda ise yoldan çıkma durumları yaşanabilir.

## 4. TASARIM

### 4.1 Tüm Sistemin Akış Şeması

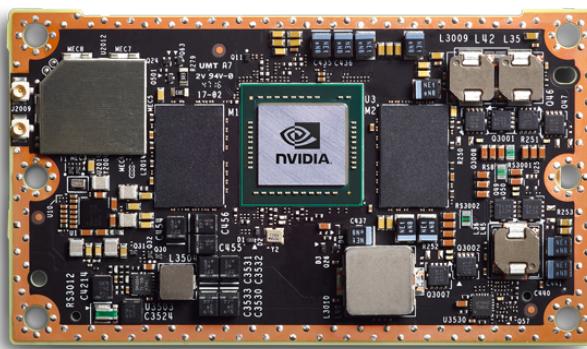


Şekil 4-1 Sistem Şeması

Bütün sistem bir şema ile anlatılacak olursa (Şekil 4.1.1) sistem 4 ana bölümde incelenebilir. Bunlar *Ham Data*, *Çevre Tanıma ve Anlamlandırma*, *Güzergah Planlama* ve *Araç Kontrol* olarak ayrılmıştır. *Ham Data* kısmında sensörlerden gelen veriler farklı alanlarda işlenmek üzere *Çevre Tanıma ve Anlamlandırma* kısmına geçecektir. Burada gerçekleşecek olan *Koni Tespiti*, *Lokalizasyon* ve *Haritalandırma*, *Duruş Tahmini* sonrası *Yerel Model* ve *Global Model* oluşturulacaktır. Daha açık ifade etmek gerekirse *Yerel Model* aracın o anki durumu ve konumu, *Global Model* aracın içinde üzerinde hareket ettiği bütün pisttir. piste entegre edilmiş halidir. Bir analogi kurmak gerekirse bunu pilotlu bir yarış aracı olarak düşünübiliriz. *Yerel Model* pilot ve aracı olurken, *Global Model* pilotun ekibini ve tüm yarış pisti olur. Daha sonra bu modeller *Güzergah Planlama* işlemine geçirilir ve burada iki model birleştirilir. *Davranış Planlama* kısmında aracın tüm bu durumlara göre nasıl hareket edeceği belirlenir ve *Araç Kontrol* aşamasına geçilir. Bu aşamada ise aracın pisti istenen şekilde takip etmesi için gereken *Engel Reddetme* sağlanır. Engel olarak tanımlanan koniler belirli bir mantık ile reddedilmelidir. Araç konilere çarpmadan ilerlerken bir yandan düzgün bir yol tutuşa sahip olmalı, diğer yandan da pisti en hızlı şekilde tamamlamak için virajlara giriş ve çıkış anları ve düz yol gibi durumlarda hızını kontrol etmelidir.

## 4.2 Donanım

Projede gerçekleştirilen hedeflenen araç ölçeklendirilmiş olduğu için(1:10) proje kapsamı ve odağı gereği araç şasisi ve şasi dinamiği çalışmaya dâhil edilmemiştir. Projenin kıtaslarına uygun olarak tasarlanmış basit bir araç şasisi üzerinde süspansiyon elemanları ile birlikte 1 adet fırçasız DC motor ve manevra kabiliyeti için ön iki tekeri kontrol edecek bir servo motor düşünülmüştür. Aracın hareket kabiliyeti mil tahrikli 4 tekere doğrudan tahrik ile tek bir fırçasız DC motordan sağlanacaktır. İşlemci üzerinde gerçekleştirilecek algoritmaların elektronik hız kontrolcüsüne iletilmesi ile aracın o anki durumuna göre vermesi gereken hareket tepkisinin bu motor sistemi ile gerçekleştirilmesi hedeflenmiştir.



**Şekil 4-2 NVIDIA TX2**

Yapılan araştırmalar sonucunda NVIDIA TX2 (Şekil 4.2.1) model işlemcide karar kılınmıştır. Benzer kapsamdaki pek çok otonom projede yaygın olarak kullanılan TX2, kredi kartı kadar küçük boyutu, 256 çekirdekli NVIDIA Pascal mimarisini ve 8 GB belleği ile geliştiricilerin vazgeçilmez seçimi haline gelmiştir. Tüm bu özellikler arasında düşük güç ihtiyacı seçilme nedeni arasında en çok öne çıkan unsur olmuştur. Sadece 7.5 watt ile günümüz masaüstü bilgisayarlarından 25 kat fazla enerji verimliliği sağlayabilmektedir. Küçük boyutuna rağmen yüksek işlem hızı ile, yarış konseptindeki problemlerin başında gelen veri gecikmelerini minimize edeceği düşünülmüştür.

Daha önce de bahsedildiği üzere aracın herhangi bir kullanıcı kontrolüne ihtiyaç duymadan %100 otonom olabilmesi için sensörlerle donatılmıştır. Bu sensörler,

- RPLidar S1
- Zed Stereo Camera
- Flipsky FSESC 6.6 VESC 6
- SparkFun 9DoF Razor IMU M0

şeklinde düşünülmüştür. Aracın ön kısmına yerleştirilmesi planlanan LiDAR ile 180° tarama yapılması öngörlülmüştür. Bu sayede araç tamamlaması gereken pisti oluşturan tüm konileri tespit edebilecektir. Aracın ilerleyeceği yolu tüm sürüş boyunca izleyecek olan derinlik kamerası konilerin tahmini uzaklıklarını belirleyebilecek, oluşturulan görüntü işleme algoritmaları ile koniler renklerine göre sınıflandırabilecektir. Standart bir otonom araç projesinde, araç yalnızca LiDAR ya da yalnızca derinlik kamerası ile görüş sağlayabilir ve fonksiyonel olarak çalışabilir. Ancak projenin odaklandığı yarış konsepti beraberinde bir meydan okumayı da getirmiştir. LiDAR kamera ikilisinden sağlanan yedekli veri sayesinde olası bir sensör arızası halinde araç fonksiyonlarını yine de yerine getirebilecektir. Aynı zamanda bu yedekli sensör mekanizması LiDAR'ın topladığı verilerin tekrarını azaltmak için de kullanılacaktır. Kullanılması planlanan LIDAR'ın modeli aşağıda görseli verilen RPLidarS1 (Şekil 4.2.2)'dır



**Şekil 4-3 RPLidar S1**

İkincil sensör, LiDAR ile senkron şekilde çalışması planlanan kamera olacaktır. İkili kamera modeli ile sağlanan görüşte bir derinlik söz konusu olmaktadır. Aracın yarışacağı pisti oluşturan farklı renkli konileri (sarı-mavi) algılaması için kameradan alınan veri görüntü işleme teknikleri ile sınıflandırılacak ve daha sonrasında harita oluşturma kısmında kullanılacaktır. Aracın ölçeklendirilmiş olması beraberinde görece daha küçük bir pisti getirmiştir. Bu sebeple seçilen kameradaki derinlik menzilinin 50 m altında olması bile bir sorun teşkil etmeyecek, aksine daha kesin sonuçlar verebilecektir. Derinlik kamerası olarak ZED Stereo Camera (Şekil 5.2.3) tercih edilmiştir. 110 derecelik görüş alanına sahip modelin

sensörleri arası 120 mm genişliğindedir, bu da aracın 1.5 m ile 20 m arası bir derinlikte görüş alabilmesine olanak sağlamaktadır.



**Şekil 4-4 ZED Stereo Camera**

Aracın hızını kumanda etmek için kilit rol oynayan donanım elektronik hız kontrolörüdür. Seçilen elektronik hız kontrolcüsünün modeli Flipsky FSESC 6.6 VESC 6 (Şekil 4.2.4)'dır.



**Şekil 4-5 Flipsky FSESC 6.6 VESC 6**

İçeriğindeki FET yapısı sayesinde motorun hızını ve yönünü düşük hızlarda bile hassas olarak ayarlayabilmesi motor sürücüsü olarak seçilmesindeki en geçerli sebeptir. İşlemciden gelen kontrol bilgisini FET'ler yardımıyla yorumlayarak motora iletecektir. Bu sayede araç, hareketini istenen doğrultuda sağlayabilecektir.

Düzen bir sensör donanımı ise ataletsel ölçüm birimidir. İçerisinde 3 adet üç eksenli sensöre sahiptir. Bunlar ivmeölçer, jiroskop ve manyometredir. Projede, IMU (Inertial Measurement Unit) aracın o anki ivmesinin tespit edilmesini ve bu veriyi diğer sensörlerden gelen verilerle birlikte EKF'ye sokup kesin bir sonuç elde edilmesini sağlar. Seçilen IMU SparkFun 9DoF Razor IMU M0 (Şekil 4.2.5)'dır.

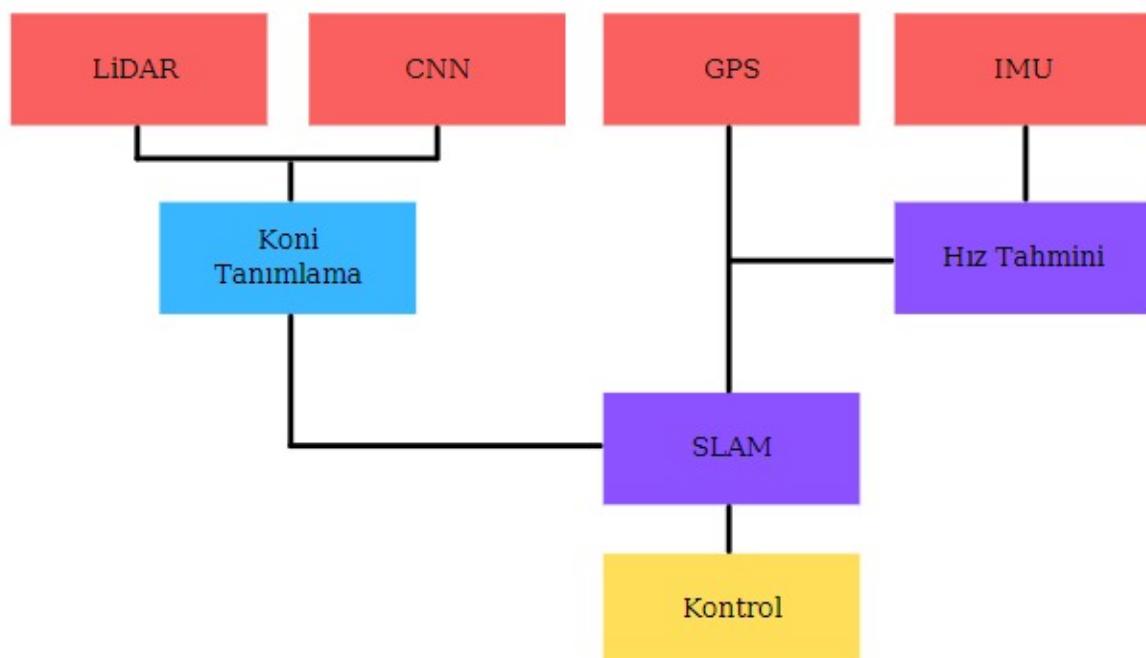


Şekil 4-6 SparkFun 9DoF Razor IMU M0

### 4.3 Yazılım

Gerekli mekanik aksam ve donanımlar sağlandığında bile araç henüz hareket etmeye hazır değildir. Aracın istenen tüm fonksiyonları karşılayabilmesi için bir yazılım sisteme ihtiyaç duyulmaktadır. Bu bölümde yazılım sistemi tüm alt birimleri ile detaylı şekilde incelenmiştir.

#### 4.3.1 Giriş ve Genel Yaklaşım



Şekil 4-7 Sensörlerden gelen verilerin işleniş şeması

Projemiz olan otonom bir yarış aracını yapabilmek için yaklaşımımızdan ve ihtiyaçlarımızdan bu rapor içerisinde bahsedilmiştir. Temel olarak bahsedilenler mekanik tasarım, donanım yapısı ve yazılım yapısı şeklinde dir. Mekanik tasarım yapılırken hedeflenen, kullanılacak donanımların fiziksel olarak taşınabilmesi; birbirleri arasında fiziksel bağlantı kurabilmeleri, aracın dört tekerlekli bir araç olarak rahatça hareket edilebilmesi ve kullanılacak bağlantı elemanlarının belirlenmesi yönünde olmuştur. Bu donanımların seçilirken gözetilen şeylerin başında ise maliyet ve kullanılacak algoritma şemalarına uygunluk vardır. Bu durumda mekanik tasarım, donanım yapısını donanım yapısı ise yazılım yapısını en uygun şekilde gerçekleştirebilmek üzerine oluşturulmuştur. Çünkü projemiz otonom bir yarış aracı olarak proje hedeflerinde bahsedildiği gibi asıl odaklandığı yeri yaratıcı bir yazılım teması oluşturabilmek ve uygun algoritmalar ile pisti optimum sürede bitirebilmektir.

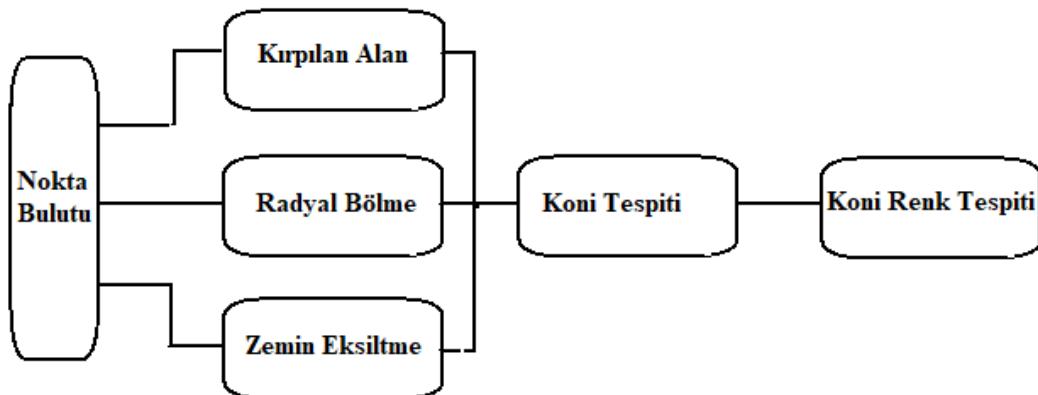
Yazılım yapısını açıklamak için öncelikle aracın piste çıktığında yaşayacağı problem ve ihtiyaçları doğru belirlemek gereklidir. Aracın pisti tamamlamak için öncelikle pisti algılayabiliyor olması gerekmektedir. Pistte üzerinde hareket edebileceği yolu; bu yolun sınırlarını, başını ve sonunu biliyor olması gerekmektedir. Eğer bunu tamamlayabilirse karşılaşacağı ikinci problem, aracın pistin neresinde olduğunu doğru belirleyebilmesidir. Eğer bunun belirlenmesi de başarı ile gerçekleşecek olursa da aracın ne yapması gerektiğini biliyor olması ve onu gerçekleştirebilmesi beklenmektedir.

Bu doğrultuda yazılım yapısı üç başlık altında incelemektedir. Bunlardan birincisi algılama; ikincisi hareket tahmini ve haritalandırma, üçüncüsü de kontrol kısmı olarak belirlenmiştir. Aracın bunları gerçekleştirebilmesi adına öncelikle bir tur boyunca SLAM (Anlık konumlandırma ve haritalandırma) gerçekleştirmeli ve daha sonra yarışmak için on tur atmalıdır. Bu ilk turu yavaş tur olarak adlandırılmaktadır. Yavaş tur boyunca SLAM ile haritalandırmasını tamamlayan araç daha sonraki turlarda sadece haritaya göre kendi konumunu belirleyerek hareket eder ve buna da konumlandırma aşaması denir. SLAM aşaması ile konumlandırma aşamasının birbirinden ayrılmasına ihtiyaç duyulmasının sebebi aracın LiDAR ve kamera gibi sensörlerinin yüksek hızlarda aldıkları verileri birleştirip anlık bir haritalandırma yapmasındaki zorluktan kaynaklıdır. Harita yani pist bir kere oluşturulduktan sonra değiştirilmeyeceğinden ve konumu değişen tek şey aracın kendisi olacağından yavaş tur sonraki aşamalarında sadece aracın konumlandırmasını yapmak yeterlidir.

### 4.3.2 Algılama

Aracın görüşünü sağlayabilmek adına iki ayrı sensör kullanılmıştır. Bu sensörler LiDAR ve stereo kameradır. Bu iki sensör birbirinden ayrı olarak araca veri aktarırlar. Bu sayede iki sensörden birinde arıza olması durumunda dahi araç yarışmaya devam edebilir. Bu iki sensörün ana görevi pistin sınırlarını oluşturan konileri görmek ve onların konumu hakkında bilgi sağlamaktır. Böylece araç pistte ilk turunu gerçekleştirirken haritalandırma yapabilmektedir. LiDAR konilerin yerini belirlemekte kullanılırken konilerin rengini de tespit edebilir. Fakat konilerin rengini tespit etmekte asıl öne çıkan sensör kameradır. Kamera da konilerin yerini tespit etmekte ve çoklu nesne tespit etme için kullanılır.

#### 4.3.2.1 LiDAR Tabanlı Algılama

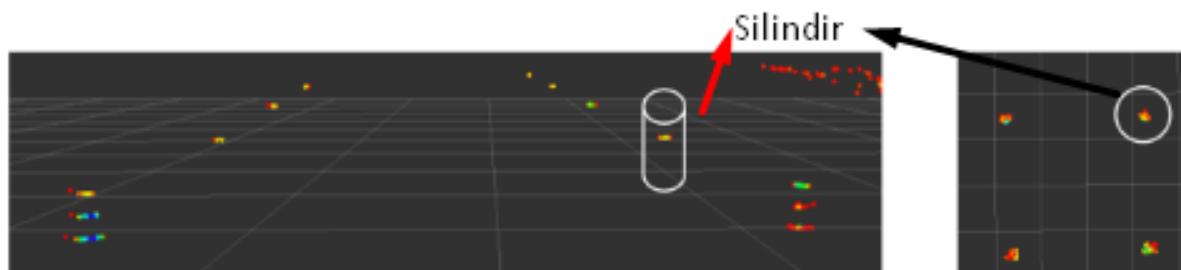


Şekil 4-8 LiDAR tabanlı algılama

LiDAR sensöründen nokta bulutu olarak tabir edilen veri akışı gerçekleşir. Bu nokta bulutu LiDAR sensöründen çıkan lazer ışınlarının temas ettiğleri ilk yüzeyi temsil eder. Bu nokta bulutundan aracın tespit etmesini beklediğimiz konilere ait olan noktaların ayıklanması gerekmektedir. Bunu üç adımda gerçekleştirmek mümkündür ve bu adımlar ön-isleme kısmını oluşturur. (Şekil 4.3.3.1.1) Birinci adım olarak 360 derece tarama yapan LiDAR sensörünün aracın arka kısmının taramış verilerini ayıklamak gereklidir. Bu şekilde 180 derecelik bir açıda yani aracın önünü görebilecek şekilde nokta bulutu elde edilir. Daha sonra ise bu 180 derecelik görüş açısı kesirlere bölünür. Bu sayede farklı kesirler üzerindeki nokta bulutları arasında karşılaştırma yapılabilir. En son olarak da farklı kesirlerdeki nokta bulutları arasında sürekli doğru parçaları gezdirilir ve tespit edilen cisimlerin en alt noktası belirlenir. Belirlenen bu en alt noktaların zeminin de eğimine göre altında kalan kısmı zemini temsil eder. Burada kesirlere bölme yönteminin kullanılmasının asıl sebebi yolun görüntüsünün perspektifidir.

Konileri sadeleştirmek asıl amacımız olduğundan zemin olarak kabul edilen bu kısım nokta bulutundan ayırtırılır ve nokta bulutunda sadece konilerin noktaları kalmış olur. Böylece ön-isleme kısmı tamamlanır fakat bu zemin kaldırma işlemi sırasında konilere ait bir kısım noktalar da kaldırılmış olur.

Bu aşamada ise koni tespit etme adı verilen kısma geçilmiş olunur. Burada konileri doğru belirleyebilmek için nokta bulutundan kaldırılan koni noktalarını da toplamak gerekmektedir. Öklid mesafesine dayalı kümeleme algoritması (teorik altyapı kısmında açıklanmıştır) kullanılarak nokta bulutundan yer noktalarının kaldırılmış ve kaldırılmamış durumları için ayrı ayrı kümelendirme gerçekleştirilir ve bu filtreden geçen koniler bir silindir şeklinde belirlenir daha sonra renk belirlenmesi amacıyla renk tahmin modülüne iletılır. Aşağıda konilerin araç bakışından ve yukarıdan görünümü (Şekil 4.3.2.1.2) ile silindir olarak çevrelenmesi görülmektedir.



**Şekil 4-9 Konilerin araç bakışından ve yukarıdan görünümü**

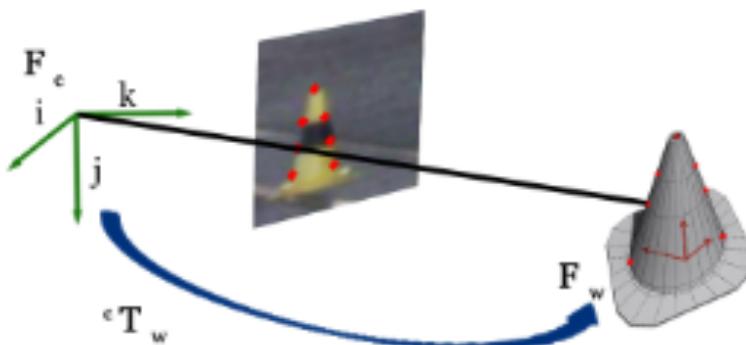
Renk tahmin modülünde ise CNN (Evrişimli sinir ağları) kullanılarak silindir içerisindeki nokta bulutu  $32 \times 32$  ızgara yapısına bölünür. Bu yöntem ile sarı ve mavi konilerin nokta bulutunda renk yoğunlukları (sarı renk yoğunluğu fazla mavi renk yoğunluğu az) ile oluşturdukları farklılıklar tespit edilir. Bu sayede sarı ve mavi koniler birbirinden ayırt edilebilir.

#### 4.3.2.2 Kamera Tabanlı Algılama

Kamera da LiDAR ile benzer işlevi görmektedir. İki sensörün bir arada kullanılmasının sebebi bir arıza durumunda aracın yarışmaya devam edebilmesi ve sensör füzyonu ile ikili doğrulama sistemi oluşturulmasıdır. Kamera tarafından çoklu nesneler algılanabilir ve bu çoklu nesnelerin yanı konilerin konumu ve rengi belirlenir.

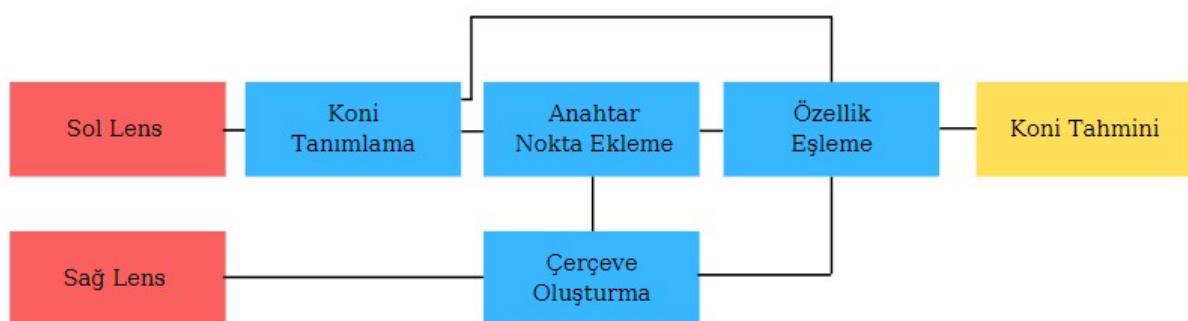
Konilerin görüntü üzerinde tespit edilebilmesi için YOLO kullanılmıştır. YOLOv2'ye tanımlanan koni fotoğrafları ile eğitilmiş ve gerçek zamanlı tespit yapabilmesi sağlanmıştır. Fakat konilerin tespiti tek başına yeterli değildir. Bu konilerin konumunun tespiti de büyük önem arz etmektedir. Kameradan alınan görüntüler iki boyutludur ve konilerin konumlandırılmasının yapılabilmesi için üç boyutlu bir veriye ihtiyaç vardır. Bu sebepten

dolayı koni görüntülerinin bazı noktaları anahtar noktalar olarak belirlenmiştir. Bu işleme anahtar nokta regresyonu adı verilmektedir. Bu yöntem, iki boyutlu cisimlerin üç boyutlu tahminlerini yapmakta kullanılır.



**Şekil 4-10 7 noktalı koni**

Kullanılan kamera çift lensli bir stereo kameradır. Bu kameranın sol lensinden gelen veriler ile sağ lensinden gelen veriler birbirinden farklı işlemlere tabii tutulurlar. Böylece istenilen derinlik algısı sağlanılmış olur. Sol lensten gelen veriler tespit edilen konilerin verileridir. Bu veriler YOLO tarafından çerçeve içerisine alınır ve anahtar noktalar üzerine yerleştirilir. Daha sonra bu anahtar noktalar yerleştirilmiş görüntü, perspektif-n-noktası yöntemi ile sol lensteki görüntüden sağ lensteki görüntüye iz düşümü alınır ve sağ lensteki görüntüde tahmini bir kutulama yapılır. Aşağıda sağ ve sol lenslerden gelen görüntülerin işleniş akışı verilmiştir.

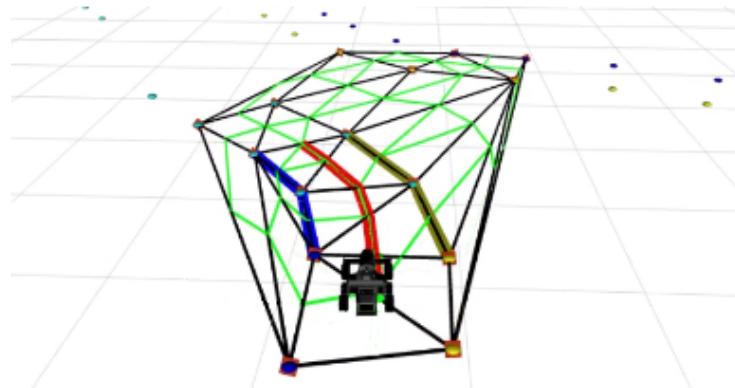


**Şekil 4-11 Stereo kamera ile koni tanımlama ve pozisyonlandırma**

#### 4.3.3 Hareket Tahmini ve Haritalandırma

Hız ve hareket tahmini yapabilmek otonom araç için kritik bir görevdir. Gerçeğe yakın tahmin edilen hız ve hareketler sayesinde aracın pist içerisindeki lokalizasyonu daha doğru tespit edilebilmektedir. Bunun dışında aracın pisti optimum sürede tanımlaması için doğru zamanda

hızlanması ve yavaşlaması gerekmektedir. Eğer hareket ve hız tahmini doğru gerçekleştirilemez ise doğru ivmelenmeler de kaçırılmış olunur. Haritalandırma ise iki kısımda incelenebilir. Birincisi yavaş tur sırasında gerçekleştirilen SLAM(Anlık konumlandırma ve haritalandırma) kısmıdır. Bu kısımda araç pisti global model haritalandırmayı bitirir ve daha sonraki turlarda anlık haritalandırma yapmaz. Sadece yerel model olarak kendi konumunu belirler. Bu projede yavaş tur sırasında aracın anlık olarak haritalandırma yapabilmesi için RRT(Rapidly-exploring Random Tree (Şekil 4.3.4.1)) algoritması kullanılmıştır. Bu algoritmanın çalışma prensibi tüm alanı hızlı bir şekilde tarayarak noktalar arasında rastgele yollar oluşturup o yolların içinden en mantıklı olanları seçmesi üzerindedir.



**Şekil 4-12 RRT**

#### 4.3.4 Kontrol

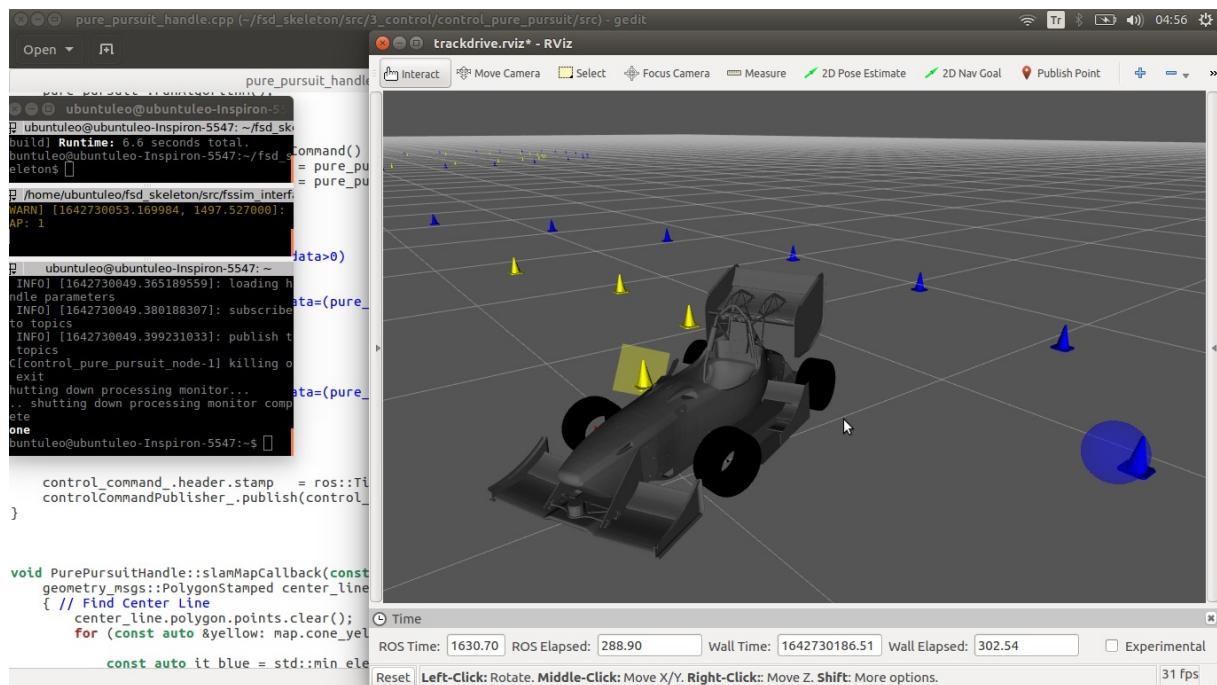
Aracın kontrolünü sağlama kısmında daha önce elde ettiğimiz verileri ve çıkarımları kullanmakla beraber bazı yeni yaklaşımlar da getirmemiz gerekmektedir. Aracımız yavaş tur sırasında RRT algoritması ve SLAM yöntemi ile haritalandırma yapmayı başardıktan sonra elimizde pistin haritası ve aracın konumu bulunmuş olur. Artık yarışma turlarına geçmek gerektiğinden aracımızın haritalandırma modundan konumlandırma moduna geçmesi gerekmektedir. Böylece kendini harita üzerinde konumlandırmış olur. Aracın konumunu değiştirecek olan şey ise hızlanıp yavaşlaması ve ön tekerlekleri ile yaptığı manevralar sayesinde olur. Bu durumda aracın konumunun hangi şartlarda nasıl değişeceğini daha iyi belirleyebilmek adına bir araç modeli oluşturulması gerekmektedir. Çünkü araç pist içerisinde hemen hemen hiçbir zaman lineer bir hareket sergilemeyeceği için bu hareketleri tahmin etmek ve yönlendirmek zorlaşmaktadır. Bunun için aracın modeli dinamik bisiklet modeli adı verilen bir yöntem ile hazırlanmıştır. Aracın yatay ve dikey hızı; ağırlığı, tekerleklerle

uygulanan yanal kuvvetler, direksiyon açısı ve ağırlık merkezinin konumunun da dahil edilerek aracın hareketinin hesaplandığı bu yöntem, dinamik bisiklet yöntemidir.

Şekillendirme formülasyonu ise aracın haritalandırılmış pistte mümkün olan en kısa ve hızlı yol belirlemeye yarayan yöntemdir. Daha sonra bu dinamik modelden elde edilen veriler, şekillendirme formülünden gelen veriler ve aracın pist üzerindeki konumuna göre aracın pist üzerinde belirlediği yolda hareket etmesini sağlayacak MPC (Model Predictive Control) yani model öngörülü kontrol formülü oluşturulur. Bunlar ışığında aracın kontrolü ESC ve ECU tarafından sağlanmaktadır.

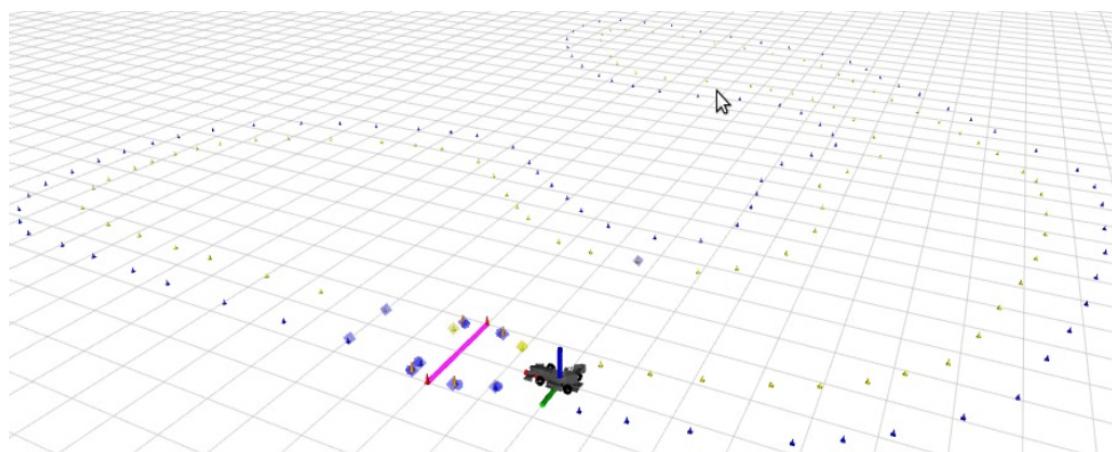
## 5. TASARIM DOĞRULAMA

Bu bölümde daha önce bahsedilen tüm geliştirme ortamlarından elde edilen çıktılar yer almaktadır. Matematiksel modellemeler tamamlandıktan sonra, geliştirme ortamlarından test ederek oluşturulan tasarımlı doğrulamak amaçlanmıştır.



**Şekil 5-1 Aracın Rviz Simülasyonundan Görüntüsü**

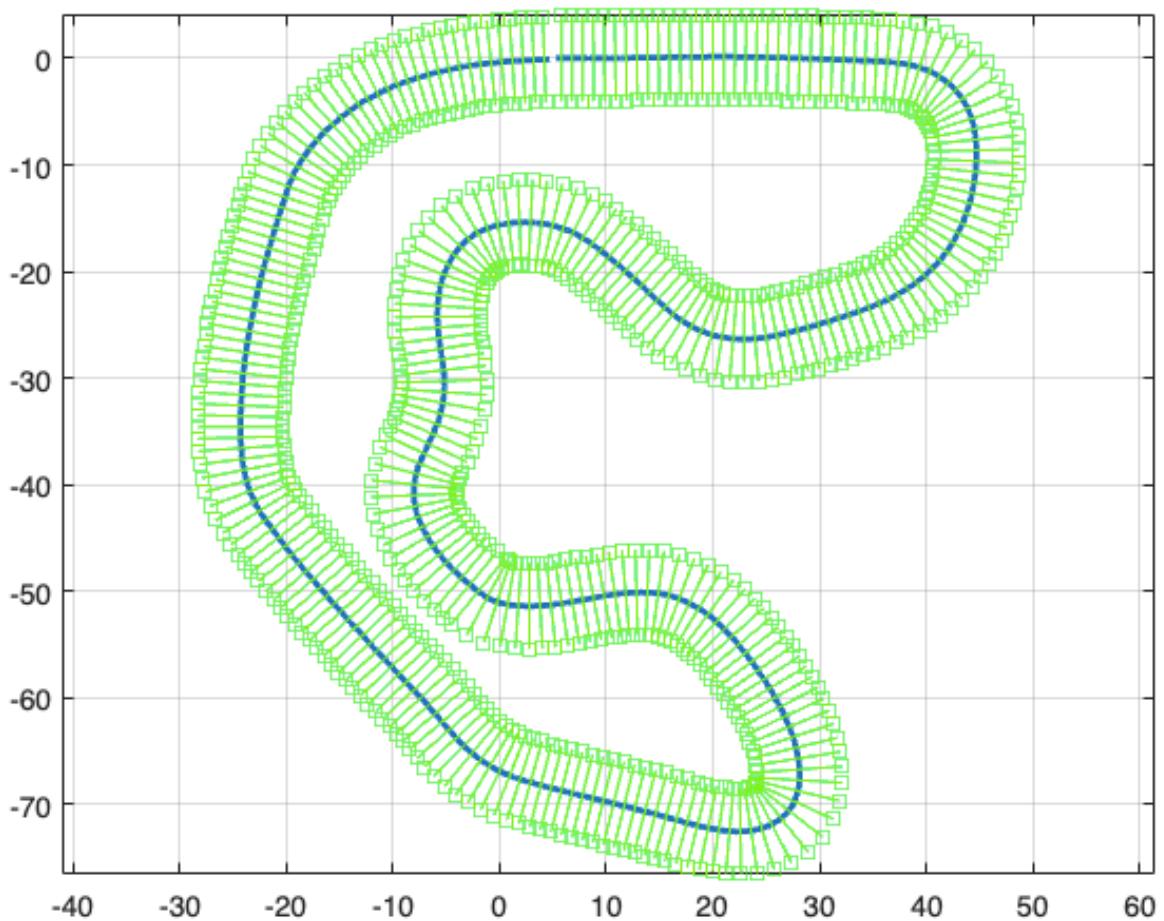
### 5.1 Pist Özellikleri



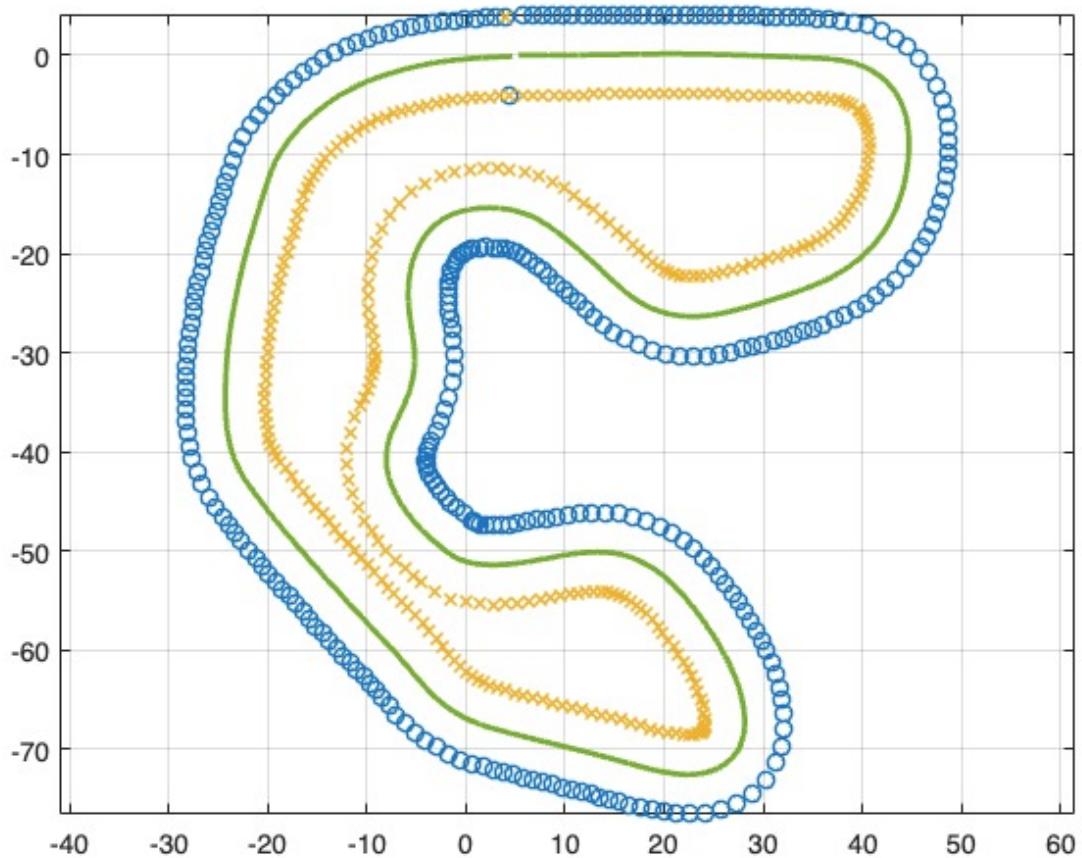
**Şekil 5-2 Aracın Pistinin Uzaktan Görünümü**

Tasarlanan otonom aracın, SAE tarafından düzenlenen FSD yarışmasına rekabet ettiğinden daha önce bahsetmiştik. Bu durumda tasarlanan pistin özelliklerinin de bu yarışma özelliklerine göre belirlenmesi gerekmektedir. Başlangıç/bitiş çizgisinin bulunduğu kısımda

biri sağda biri solda olmak üzere iki turuncu koni, ve pist boyunca pistin sol sınırında mavi, sağ sınırında sarı koni bulunacak şekilde simülasyona eklenmiştir. Pistte düz yolda mavi ve sarı konilerarası belirlenen mesafe 3.5 metredir. Pist, zorlayıcı olması açısından bazı farklı viraj tipleri düşünülverek oluşturulmuştur. Aracın azami hızına çıkabilmesi, fren yapabilmesi ve daha sonra çok keskin bir viraj alıp alamadığını görmek için başlangıç çizgisindeki en uzun düz yoldan hemen sonra sağa doğru keskin bir viraj konulmuştur. Aracın, farklı yönlere ardı ardına slalom yaparak dönüşünde pistten çıkış çıkmayacağı test edilmek için sol-sağ-sol-sağ şeklinde ardışık virajlar eklenmiştir. Aracın hızlı durumdayken yavaşlayıp U dönüşüne benzer bir virajı alıp alamayacağı ve koordinat düzlemindeki orijin noktasına ters nasıl hareket edeceğini gözlemlemek için sert bir U dönüşü de eklenmiştir.

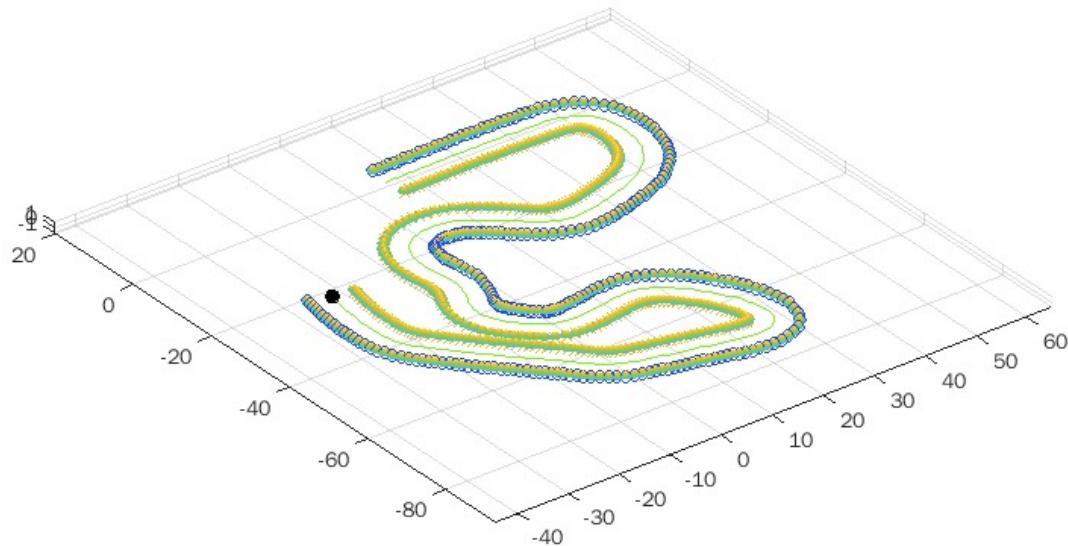


Sekil 5-3 Pistin MATLAB Üzerinden Oluşturulması



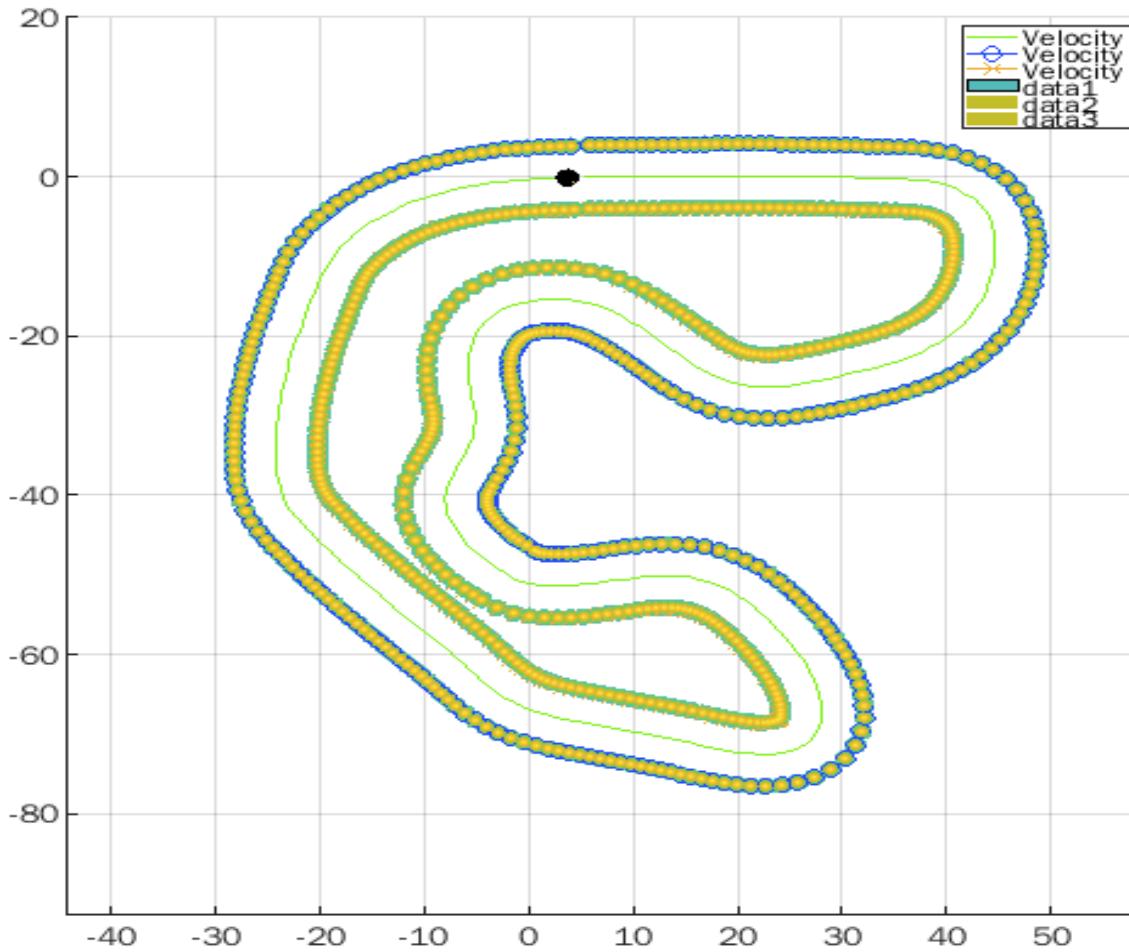
Şekil 5-4 Pistin MATLAB Üzerinden Oluşturulması

## 5.2 Haritalandırma ve Lokalizasyon Simülasyonu



**Şekil 5-5 Haritalandırmannın MATLAB Üzerinden Gösterimi**

Aracımız RViZ programında hareketini sürdürürken arka planda RViZ programı ile entegre olarak Gazebo programı da çalışmaktadır. Araç aslında haritalandırma ve lokalizasyon yani SLAM aşamasını başarılı bir şekilde gerçekleştirmektedir. Bunun görselleştirilmesi ekran kartını fazlaıyla yorduğundan, bilgisayar performansını artırmak adına haritalandırma ve lokalizasyon görselleştirmesi MatLAB programı üzerinden yapılmıştır. Bunun bizim projemizde hiçbir eksik özelliği yoktur çünkü pistimiz düz bir zemindedir ve aracın yüksekliği ile konilerin yüksekliği zaman içerisinde değişmemektedir. Bu yüzden simülasyonumuz iki boyutlu olarak da haritalandırma görselleştirmesini gerçekleştirebilmektedir. Yukarıdaki resimde görünen siyah nokta aracımızı temsil etmektedir ve araç RRT algoritması ile ilk turunu tamamlarken bu görüntü çekilmiştir. Bu görüntüde araç, pistin tamamladığı kısımlarındaki mavi ve sarı konilerin yerini tespit etmiş ve bu koniler arasında hareket edeceği yolu da yeşil çizgi ile işaretlemiştir. Bu kısmı aynı zamanda haritalandırma olarak da adlandırılır. İkinci turdan itibaren Saf Takip metodu ile aracımız bu çizгиyi takip edecektir. Aşağıdaki görselde ise aracın Saf Takip yaparken yol çizgisini takip ettiği lokalizasyon görüntüsü verilmiştir.



**Şekil 5-6 Aracın MATLAB Üzerinden Haritasını Oluşturma**

### 5.3 İlk Turun Tamamlanması

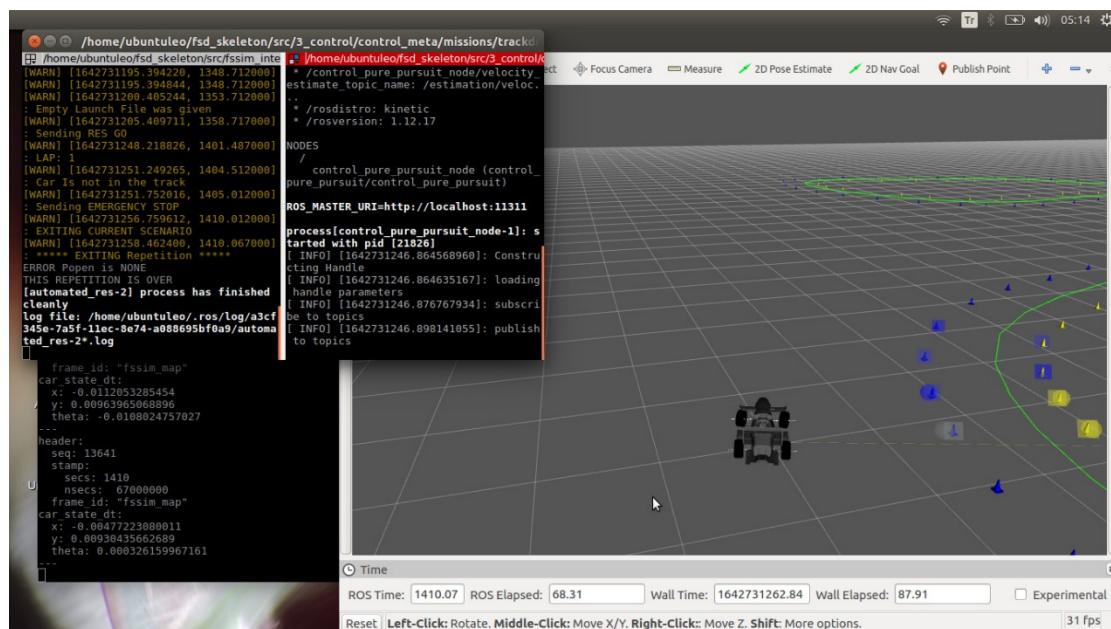
Daha önce bahsettiğimiz gibi araç, ilk turunu RRT algoritması ve Saf Takip kontrolcüsü sayesinde tamamlamaktadır. Detaylı olarak anlatacak olursak; araç piste ilk konulduğu anda çevresini tanıtmamaktadır. Bu durumda aracın çevreyi algılama yöntemi olan sensörlerin devreye girmesi gerekmektedir. Bu sensörler LiDAR ve kameradır. LiDAR ve kameradan gelen verilerin ROS paketlerinde de bulunan Genişletilmiş Kalman Filtresi yardımı ile füzyonu gerçekleştirilir. Böylece araç, çevresinde neler olup bittiğini görebilir. Kamera derinlik algısı ve konilerin renk tespiti için büyük önem arz etmektedir. Daha sonra aracımız RRT algoritması ile çevresine rastgele yollar atamaya başlar. Bu atadığı yollardan en gidilebilir olanını seçerek yeni yol çizgisini oraya doğru belirler. Saf Takip kontrolcüsü ise bu anlık olarak belirlenen yola bakış mesafesi doğrultusunda anlık olarak noktalar atar. Bu noktaları da takip eder. Bu turda aracımız oldukça yavaştır çünkü RRT algoritmasının

haritalandırmada kullanılabilmesi için birçok farklı yol olasılığı oluşturması ve bunlar arasında seçim yapması gereklidir. Aracımız ilk turunu tamamladıktan sonra yarış turuna geçer.

## 5.4 Yarış Turlarının Tamamlanması

Yarış turuna başlarken artık aracımızın elinde pistin haritası, konilerin konumu ve pistte izleyeceği yol bellidir. Saf takip kontrolcüsü sayesinde araç, yolda daha önce belirlediği yol çizgilerini o çizgilere kendi doğrultusunda noktalar atayıp daha önce formüle ettiğimiz şekilde kendi tekerlek açısını belirler. Bu tekerlek açısı verisinin belirler ve araca anlık olarak bu komutu gönderir. Tabii ki de milisaniyelik olsa da buralarda hatalı açı verileri oluşmaktadır. Bunun önüne geçebilmek için Saf Takip kontrolcüsü ile hata kontrolü ve bu hatanın bir sonraki komutta iyileştirilmesi sağlanmaktadır.

## 5.5 Alınan Hatalar ve Önlemler



**Şekil 5-7 Simülasyonun Adımları**

Her projede olduğu gibi projemizi tamamlarken kendi proje isterlerimizi karşılayamayan hatalar da aldık. Bizim projemiz için bu hatalar aracın koni devirmesi, pistten çıkışması ve çok yavaş gitmesi gibi problemlerdi. Mesela yukarıda aracın pistten çıkışmış bir görüntüsü bulunmaktadır. Aracın kontrolü her ne kadar bir simülasyonda gerçekleşse de program, gerçek bir araca entegre edildiği zaman bir pistten çıkışmanın bedeli çok ağır olabilir. Hem can güvenliği hem de maddi kayıpları önlemek adına, aracın pistten çıkışını fark ettiği anda durması için acil durum önlemi alınmıştır. Yukarıdaki görselde de görebileceğimiz gibi araç

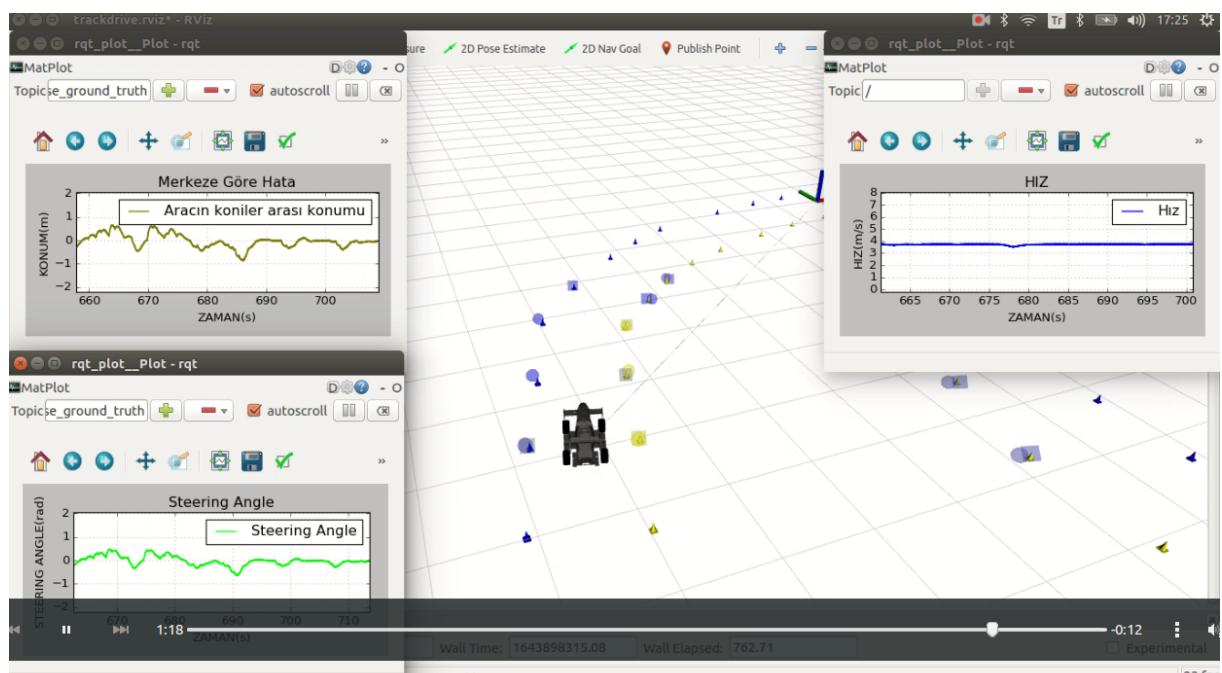
hızla pistten çıkış olmasına rağmen kendini frenlemiş ve 1.6 metre sonra durmuştur. Buna acil durum frenlemesi diyoruz. Ayrıca aracın hızını geliştirmeyi denerken birçok konı devirdik. Aracın bu konileri neden devirdiğini inceleyerek proje çıktıımızı geliştirme fırsatı da elde etmiş olduk.

## 5.6 İlk Turda (4 m/s) Hızdaki Test Sonuçları

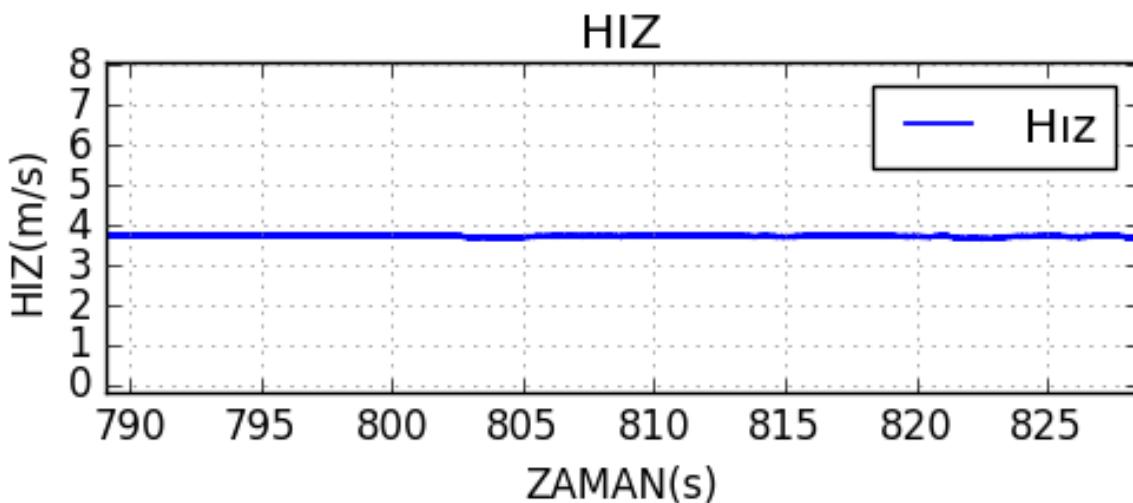
Aracın 4 m/s'deki Hız – Zaman grafiği, Steering Angle (Direksiyon Açısı) grafiği, Merkez Çizgisine (Centerline) Göre Hata grafiği verilmiştir.

Tüm bu parametreler, aracın pisti tamamlarken elde edilen sonuçlarla test edilmiştir. Elde edilen sonuçlar, matematiksel hesaplamalarla karşılaştırılmıştır.

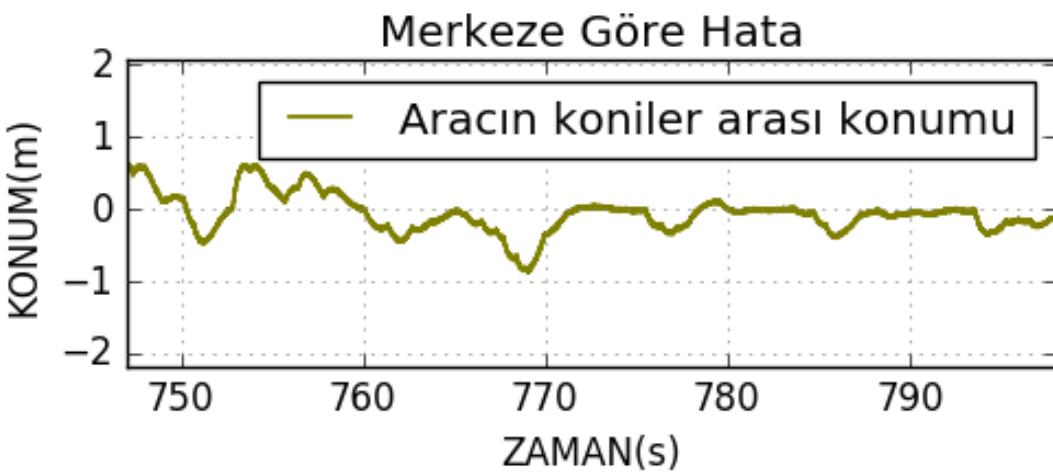
Araç ilk turunu tamamlarken gideceği yolu henüz bilmemiş için hızı yavaş durumdadır ve henüz "yarış moduna" girmemiştir. Pure pursuit(saf takip) yöntemini kullanabilmek için aracın way pointlerine(yol noktalarına) ihtiyacı vardır. Bu yol noktaları oluşturabilmek için aracın sensörlerinden(lidar ve kamera) gelen veriler ile konilerin yeri saptanır. Bu koni yerleri MatLAB aracılığı ile kullanıcıya da görselleştirilmiştir. Bu aşamada merkez çizgisi ve yol noktaları belirlenir. Araç, hareketine RRT algoritması ile konilerin rengine göre doğru yol seçerek yavaşça ilerlerken 3 metre yarıçapında, önüne doğru yol noktaları belirlemeye başlar. Bu yol noktaları birinci turun sonunda yoğunlaştırılır ve daha sonra bir merkez çizgisi elde edilir. Elde edilen merkez çizgisi daha sonraki turlarda kullanılmak üzere kaydedilmiş olur.



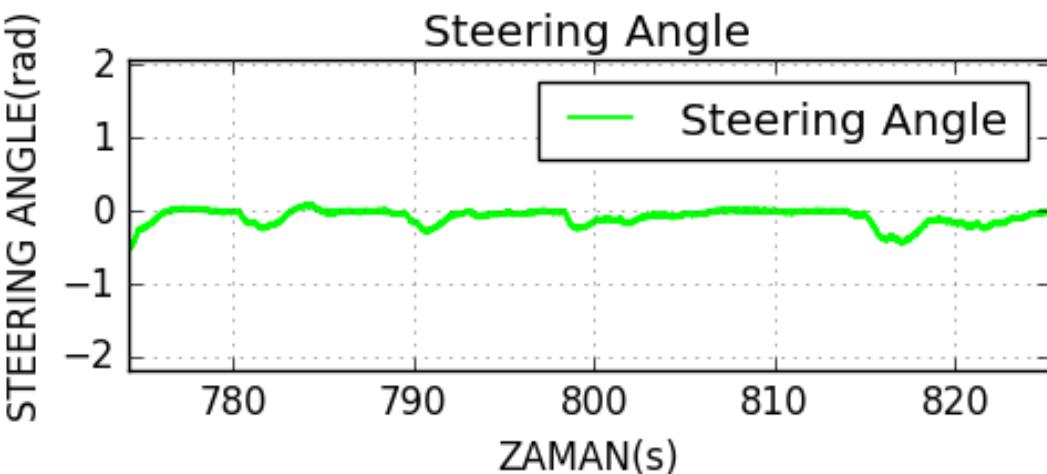
**Şekil 5-8 Aracın 4 m/s'deki Simülasyon Görüntüsü**



Şekil 5-9 Aracın 4 m/s'deki Hız-Zaman Grafiği



Şekil 5-10 Aracın 4 m/s'deki Merkez Çizgisine Göre Hata Grafiği



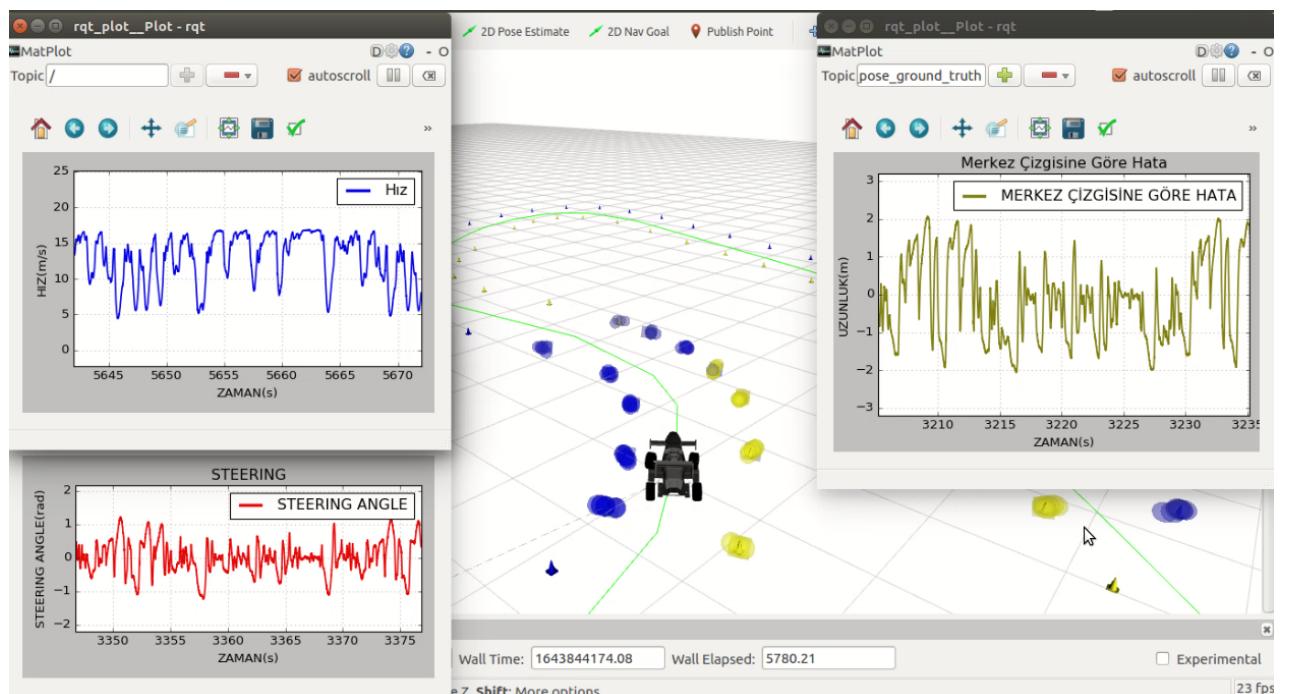
Şekil 5-11 Aracın 4 m/s'deki Direksiyon Açı Grafiği

## 5.7 17 m/s Hızdaki Test Sonuçları

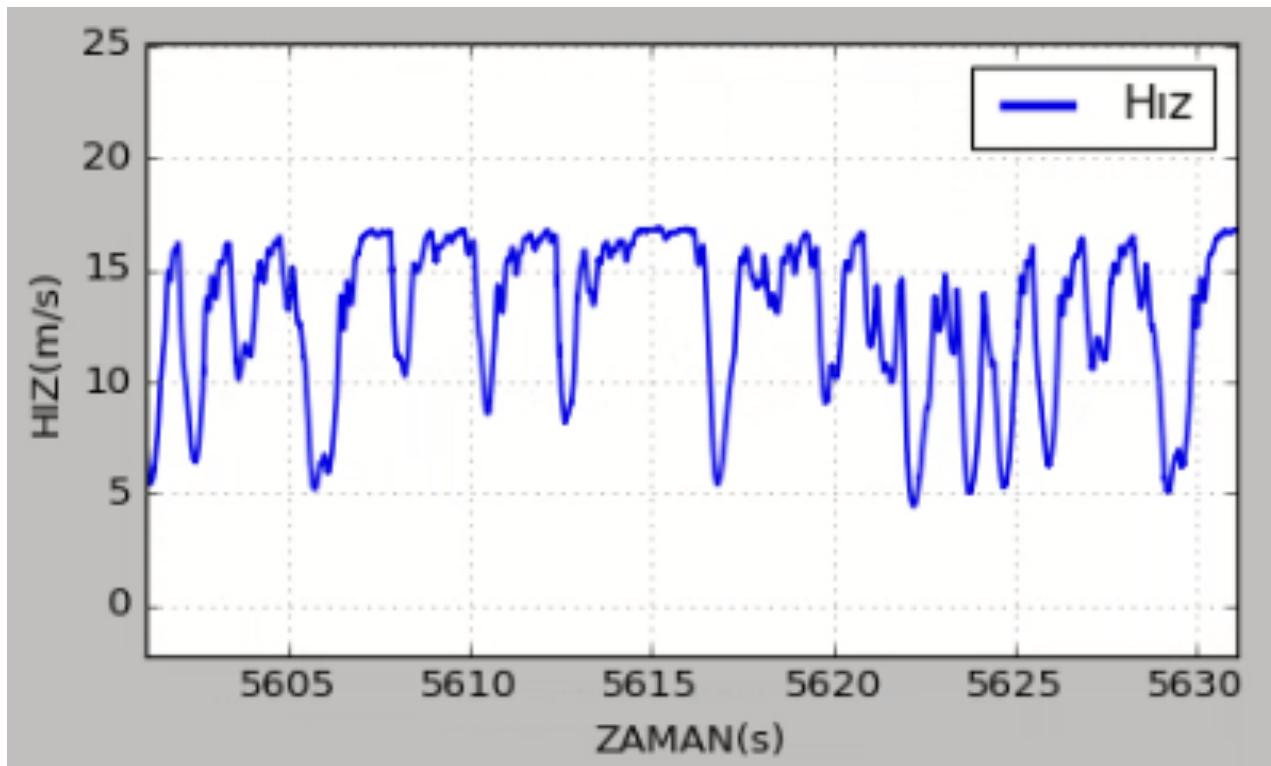
Aracın 17 m/s'deki Hız – Zaman grafiği, Steering Angle (Direksiyon Açısı) grafiği, Merkez Çizgisine (Centerline) Göre Hata grafiği verilmiştir.

Tüm bu parametreler, aracın pisti tamamlarken elde ettiği sonuçlarla test edilmiştir. Elde edilen sonuçlar, matematiksel hesaplamalarla karşılaştırılmıştır.

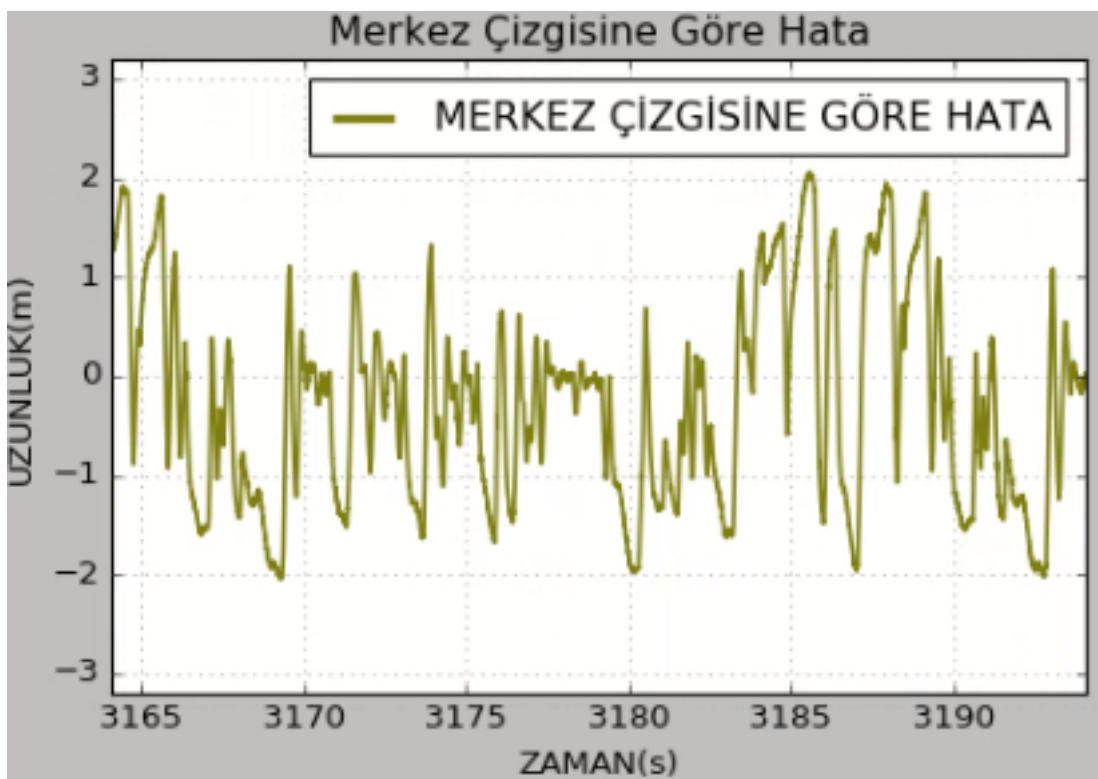
Koniler arasında aracın konum değişimini gösteren grafik ile steering angle grafiğinin uyumlu olduğu gözlemlenmektedir. Bu da araç merkez çizgisinden uzaklaşıkça metre cinsinden oluşan hatayı saf takip formülünün çıktısı olan steering angle(dümen açısı) ile düzelttiği görülür. Formülde, aracın kapalı döngüde iki hata tipi bulunmaktadır. Bunlar metre cinsinden ve açı cinsinden hatalardır. İki hata da kapalı döngüde araç hareketini sürdürürken tekrar tekrar konumunu düzeltmesini sağlar. Aracın hızı yükselirken pistte kalması amacıyla bazı parametreler değiştirilmiştir. Bunlar, merkez çizgisinde yol noktası yoğunluğunu artırmak, merkez çizgisinin konilerin orta noktasında değil de bir sonraki virajın dönüş yönüne yakın olacak şekilde ayarlanması ve aracın virajlarda steering angle(dümen açısı) arttıkça hız düşürmesi gerçeklestirmesi şeklindedir.



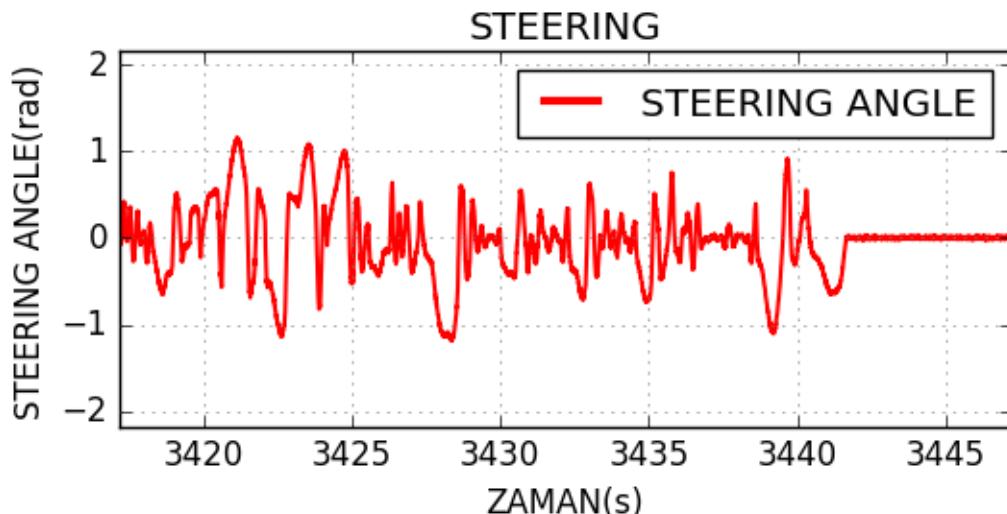
Şekil 5-12 Aracın 17 m/s'deki Simülasyon Görüntüsü



Şekil 5-13 Aracın 17 m/s'deki Hız-Zaman Grafiği



Şekil 5-14 Aracın 17m/s'deki Merkez Çizgisine Göre Hata Grafiği

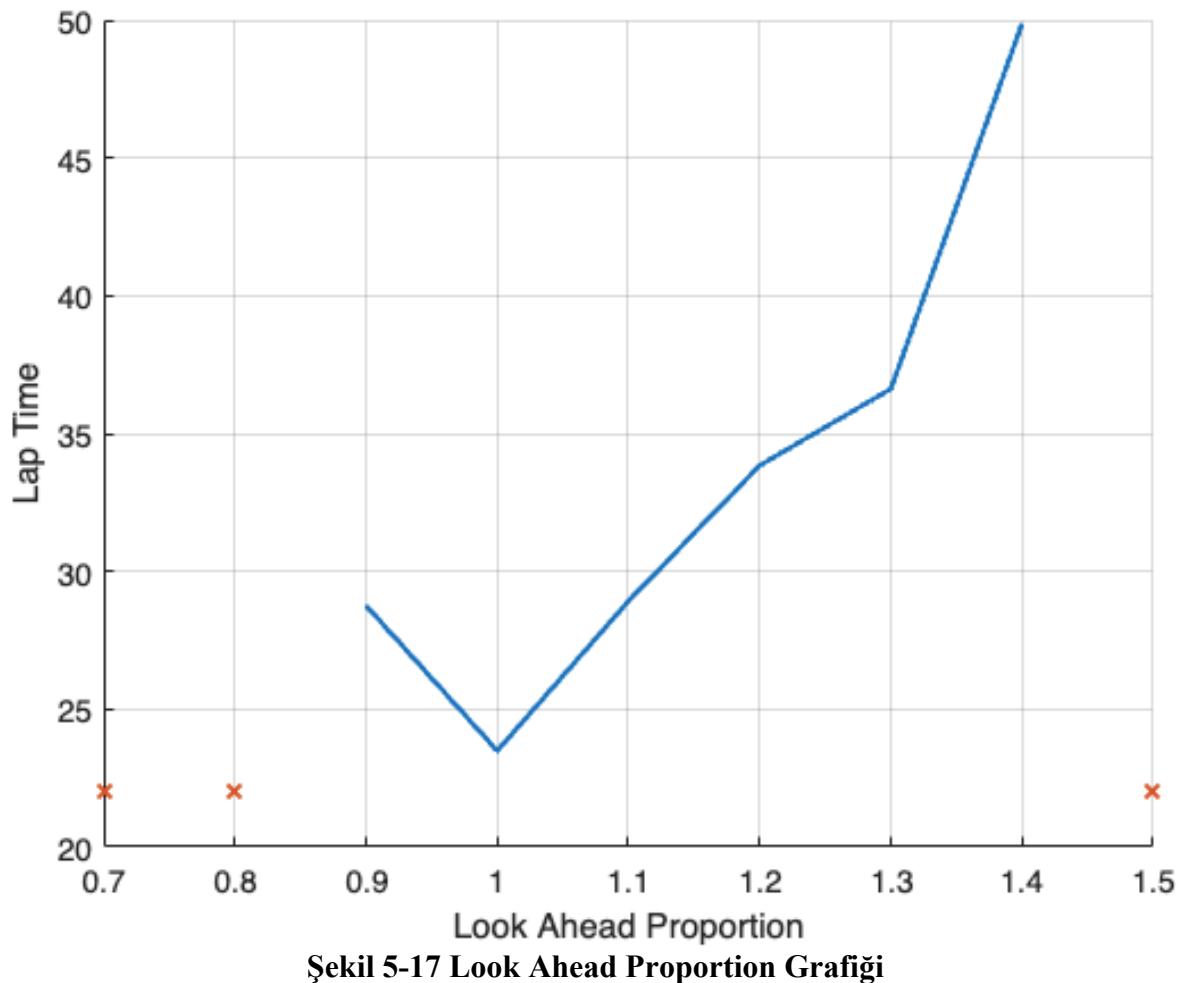


Şekil 5-15 Aracın 17m/s'deki Direksiyon Açıları Grafiği

```
[1643827995.581163, 1392.082000]: LAP: 2
[1643828019.091483, 1415.572000]: LAP Time: 23.490000
[1643828019.092015, 1415.572000]: LAP: 3
[1643828042.604670, 1439.067000]: LAP Time: 23.495000
[1643828042.605129, 1439.067000]: LAP: 4
[1643828066.118649, 1462.562000]: LAP Time: 23.495000
[1643828066.119020, 1462.562000]: LAP: 5
[1643828089.641848, 1486.057000]: LAP Time: 23.495000
[1643828089.642293, 1486.057000]: LAP: 6
[1643828113.124577, 1509.522000]: LAP Time: 23.465000
[1643828113.125113, 1509.522000]: LAP: 7
[1643828136.625622, 1533.002000]: LAP Time: 23.480000
[1643828136.626035, 1533.002000]: LAP: 8
[1643828160.182747, 1556.537000]: LAP Time: 23.535000
[1643828160.183223, 1556.537000]: LAP: 9
[1643828183.664596, 1580.002000]: LAP Time: 23.465000
[1643828183.665008, 1580.002000]: LAP: 10
[1643828207.133536, 1603.442000]: LAP Time: 23.440000
[1643828207.134153, 1603.442000]: LAP: 11
[1643828230.626030, 1626.912000]: LAP Time: 23.470000
[1643828230.627182, 1626.912000]: LAP: 12
[1643828254.163001, 1650.412000]: LAP Time: 23.500000
[1643828254.163573, 1650.412000]: LAP: 13
[1643828277.653325, 1673.877000]: LAP Time: 23.465000
[1643828277.654730, 1673.877000]: LAP: 14
[1643828301.146218, 1697.347000]: LAP Time: 23.470000
[1643828301.146781, 1697.347000]: LAP: 15
[1643828324.661383, 1720.842000]: LAP Time: 23.495000
[1643828324.661857, 1720.842000]: LAP: 16
[1643828348.154294, 1744.312000]: LAP Time: 23.470000
[1643828348.154859, 1744.312000]: LAP: 17
```

/home/reyhani/fsd\_skeleton/src/3\_c

Şekil 5-16 Aracın Hızlı Tur Tamamlama Süreleri



## 6. ÇALIŞMA TAKVİMİ

**Tablo 2 Çalışma Takvimi**

	1. Hafta	2. Hafta	3. Hafta	4. Hafta	5. Hafta	6. Hafta	7. Hafta	8. Hafta	9. Hafta	10. Hafta	11. Hafta	12. Hafta	13. Hafta	14. Hafta
Gazebo ortamında simülasyon yapılması														
Sponsor Görüşmelerinin Tamamlanması														
Parça Temininin Sağlanması														
Görüntü İşleme Yazılımının Testi														
Konumlandırma Yazılımının Testi														
SLAM Testi														
Veri Analizlerinin Tamamlanması														
Yarış Pistinin Hazırlanması														
Mekatronik Sistem Testi														
Raporlama														

Tüm Bitirme Çalışması süresi boyunca yapılması planlanan işlerin takvimi Tablo 2'de gösterildiği gibidir. Pistin Hazırlanması ve Parça Temini bütçenin fazla olması ve Covid-19 pandemisi nedeniyle ileri bir tarihe ertelenmiştir.

## 7. BÜTÇE

Proje için oluşturulan bütçe çalışmaları Malzeme Listesi ve Dış Destek Başvuruları'ndan oluşmaktadır.

### 7.1 Malzeme Listesi

Projenin gerçekleştirilebilmesi için gerekli tüm malzemeler donanımsal ürünlerdir. Donanım kısmında bahsi geçen parçaların Pazar araştırmaları yapılmış, paydaş beklenelerinden biri olan minimum maliyet hesabına göre seçimde bulunulmuştur. Tüm malzemeler Malzeme Listesi(Çizelge.6.1.1)'de görüldüğü gibidir.

**Tablo 3 Malzeme Listesi**

Malzemenin Adı	Modeli	Fiyatı(KDV Dahil)
İşlemci	NVIDIA Jetson TX2	2062₺
LiDAR	RPLidar S1	7258₺
Kamera	Zed Stereo Camera	9628₺
ESC	Flipsky FSESC 6.6 VESC 6	2357₺
DC Motor	Velineon 3500 Brushless Motor	894₺
Servo Motor	Digital High-Torque Waterproof Servo #2075	460₺
IMU	SparkFun 9DoF Razor IMU M0	446₺
Güç Kaynağı	MARC Power Pro 12V	1660₺
Anten	W-402 2.4-2.5GHz Anten	62₺
<b>Toplam</b>		<b>24827₺</b>

## 8. SONUÇLAR VE ÖNERİLER

Günümüzde en yaygın ölüm oranları trafik kazaları tarafından meydana gelmektedir. Birçok durumda bu çevresel faktörler olabiliyorken çoğunlukla sürücü hata kaynaklı kazalar yaşanmaktadır. Otonom araç endüstrisine olan ihtiyaç her geçen daha da artması ve bu alandaki çalışmaların başarıyla sonuçlanmasıyla birlikte otonom araçlara olan talep hız kazanmıştır. Dünya'da ve Türkiye'de de Ar-Ge çalışmalarıyla birlikte otonom araç sektörüne hizmet edecek robotik sistemler geliştirilmeye başlanmıştır. Otonom araç endüstrisinin eğlence sektörüne entegre olmasıyla birlikte gelecekte robotik sporlara fazlasıyla hakim olunacağı öngörlülmektedir. Bu da otonom araç endüstrisinde yapılacak olan çalışmaları kaynak sağlayacak, bu alana olan ilgiyi artıracak ve otonom araç pazarının büyümesine yarar sağlayacaktır.

Bitirme Çalışması kapsamında gerçekleştirdiğimiz bu projede;

- Otonom bir yarış aracının üzerindeki donanımları ve sensörleri belirleme,
- Bu sensörlerden gelen verilerin işlenmesi,
- Aracın Kinematik Bisiklet Modelinin oluşturulması,
- İlk turda RRT algoritması ile haritalandırma,
- Haritalandırılan pist üzerinde anlık olarak yol noktaları belirleme ve ilk turda bu yol Noktalarına Saf Takip yaklaşımı ile aracın riayet etmesi,
- Sonraki turlarda aracın planlanan yol üzerinde Saf Takip ile hareketinin sağlanması,
- Aracın hızının 4 m/s'den 17 m/s'ye çıkartılması,
- Simülasyon ortamında da bu çıktıların gösterilmesi sağlanmıştır.

Biz insanlar, araç sürerken sadece gözlerimizle çevreyi algılarız. Bu da her ne kadar aynı olmasa da kameradan alınan verilere benzetilebilir. Fakat bizler, otonom araç tasarlarken LiDAR sensörünü de kullanmaktayız. Günümüzde bazı araç firmaları LiDAR sensörü kullanmadan da otonom araç geliştirmelerine devam etmektedirler. Bu durumda biz de ileriki çalışmalar olarak sadece kameradan elde edilen veriler ile otonom bir yarış aracı tasarlanması üzerine düşünebiliriz. İnsanlar araç kullanırken sadece görüntü algılayarak araç kullanabiliyor ve hatta yüksek hızlarda yarışabiliyorlarsa bunu otonom araçların da rahatça yapabileceğini düşünmekteyiz.

Son olarak, kendi projemizde de yaptığımız gibi yol noktalarının belirlenmesinde kuadratik maliyet fonksiyonu kullanırken bu fonksiyondan alınan yol çizgisinin virajlarda virajın

apeksine yakın olacak ve bir sonraki virajın yönüne göre yol çizgisini virajın dönüş yönünün tersinde yol sınıra daha yakın olacak şekilde ayarlanmasını tavsiye ediyoruz. Bunun ayarlanması konusunda yaşanılan büyük problemlerden biri pist özelliklerine göre yarışın, saat yönünde mi yoksa saat yönünün tersine mi koşulacağının belirlenmesi olabilir. Önerimiz uygulanmadan önce bunun da dikkate alınması gereğinden bahsetmek isteriz.

## KAYNAKÇA

- [1] Zhang, Ji, Xiangjie Lv, and Yu Lv. "Research on Vehicle Control Strategy and Hardware in Loop for Pure Electric FSAE Vehicle." Journal of Physics: Conference Series. Vol. 1732. No. 1. IOP Publishing, 2021.
- [2] Kabzan, Juraj, et al. "Amz driverless: The full autonomous racing system." Journal of Field Robotics 37.7 (2020): 1267-1294.
- [3] Agnihotri, Abhijeet, et al. "Teaching Autonomous Systems at 1/10th-scale: Design of the F1/10 Racecar, Simulators and Curriculum." Proceedings of the 51st ACM Technical Symposium on Computer Science Education. 2020.
- [4] Meah, Kala, Donald Hake II, and Stephen Wilkerson. "Design, build, and test drive a FSAE electric vehicle." The Journal of Engineering 2020.10 (2020): 863-869.
- [5] Lantos, Béla, and Lőrinc Márton. Nonlinear control of vehicles and robots. Springer Science & Business Media, 2010.
- [6] Park, Myungwook, Sangwoo Lee, and Wooyong Han. "Development of Steering Control System for Autonomous Vehicle Using Geometry-Based Path Tracking Algorithm." Etri Journal 37.3 (2015): 617-625.
- [7] P. Biber et al., "The Normal DistributionsTransform: A New Approach to Laser ScanMatching," Proc. IEEE/RSJ Int'l Conf. Intelli-gentRobotsandSystems, 2003, pp.2743–2748.
- [8] Felzenszwalb et al., "Object Detectionwith Discriminatively Trained Part-BasedModels," IEEE Trans. Pattern Analysis andMachine Intelligence, vol. 32, no. 9, 2010,pp. 1627–1645.
- [9] M. Arulampalam et al., "A Tutorial on Par-ticle Filters for Online Nonlinear/non-Gaussian Bayesian Tracking," IEEE Trans.Signal Processing, vol. 50, no. 2, 2002, pp.174–188.
- [10] Barış Samancı. "Accelerometer, Gyroscope, IMU nedir?".  
<http://www.barissamanci.net/Makale/26/accelerometer-gyroscope-imu-nedir/>,  
09/06/2011

[11] MATLAB. “Understanding Sensor Fusion and Tracking, Part 3: Fusing a GPS and IMU to Estimate Pose”.

[https://www.youtube.com/watch?v=hN8dL55rP5I&ab\\_channel=MATLAB](https://www.youtube.com/watch?v=hN8dL55rP5I&ab_channel=MATLAB),

23/10/2019

[12] MATLAB. “Understanding Kalman Filters, Part 5: Nonlinear State Estimations”.

[https://www.youtube.com/watch?v=Vefia3JMeHE&ab\\_channel=MATLAB](https://www.youtube.com/watch?v=Vefia3JMeHE&ab_channel=MATLAB),

17/05/2017

[13] AerospaceControlsLab. “Controlling Self Driving Cars”.

[https://www.youtube.com/watch?v=4Y7zG48uHRo&ab\\_channel=AerospaceControlsLab](https://www.youtube.com/watch?v=4Y7zG48uHRo&ab_channel=AerospaceControlsLab), 22/07/2015

[14] MATLAB. “Understanding SLAM Using Pose Graph Optimization || Autonomous Navigation, Part 3”

[https://www.youtube.com/watch?v=saVZtgPyyJQ&ab\\_channel=MATLAB](https://www.youtube.com/watch?v=saVZtgPyyJQ&ab_channel=MATLAB),

08/07/2020

[15] Güner, Şafak. Otonom Bir Otomobil İçin Hız Kontrolörü Tasarımı Ve Uygulaması.

Diss. Fen Bilimleri Enstitüsü, 2013.

[16] Culley, Jacob, et al. "System design for a driverless autonomous racing vehicle." 2020 12th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP). IEEE, 2020.

## ÖZGEÇMİŞ

**Ad Soyad:** Alperen ŞENTÜRK  
**Doğum Yeri ve Tarihi :** AYDIN, 1996  
**Adres :** Yıldız Teknik Üniversitesi Beşiktaş Kampüsü, İstanbul  
**E-Posta :** alperensenturk@gmail.com  
**Lisans :** Yıldız Teknik Üniversitesi, Makine Fakültesi,  
Mekatronik Mühendisliği  
**Mesleki Deneyim:** BSH Ev Aletleri San. Ve Tic. A.Ş. – Kalite Mühendisi

**Ad Soyad:** Mahmut REYHANİ  
**Doğum Yeri ve Tarihi :** HATAY, 1997  
**Adres :** Yıldız Teknik Üniversitesi Beşiktaş Kampüsü, İstanbul  
**E-Posta :** mahmutreyhani07@gmail.com  
**Lisans :** Yıldız Teknik Üniversitesi, Makine Fakültesi,  
Mekatronik Mühendisliği  
**Mesleki Deneyim:** Tosçelik Profil ve Sac Endüstrisi A.Ş. – Stajyer Mühendis

**Ad Soyad:** Zeynep Esra İŞLER  
**Doğum Yeri ve Tarihi :** İSTANBUL, 1996  
**Adres :** Yıldız Teknik Üniversitesi Beşiktaş Kampüsü, İstanbul  
**E-Posta :** zeynepesraisler@gmail.com  
**Lisans :** Yıldız Teknik Üniversitesi, Makine Fakültesi,  
Mekatronik Mühendisliği  
**Mesleki Deneyim:** Ford Otomotiv Sanayi A.Ş. – Ürün Geliştirme Mühendisi