TEKNOFEST İSTANBUL
HAVACILIK, UZAY VE TEKNOLOJİ FESTİVALİ

ROBOTAKSİ – BİNEK OTONOM ARAÇ YARIŞMASI
KRİTİK TASARIM RAPORU

ÜNİVERSİTE: Boğaziçi Üniversitesi

TAKIM ADI: BURST

TAKIM LİDERİ: Kerem Zaman

AKADEMİK DANIŞMAN: Dr. Sinan Öncü

İçindekiler

1. Takım Organizasyonu **3**

2. Ön Tasarım Raporu Değerlendirmesi **3**

3. Araç Özellikleri **5**

4. Sensörler **7**

5. Araç Kontrol Ünitesi **13**

6. Otonom Sürüş Algoritmaları **20**

7. Özgün Bileşenler **21**

8. Güvenlik Önlemleri **35**

9. Test **37**

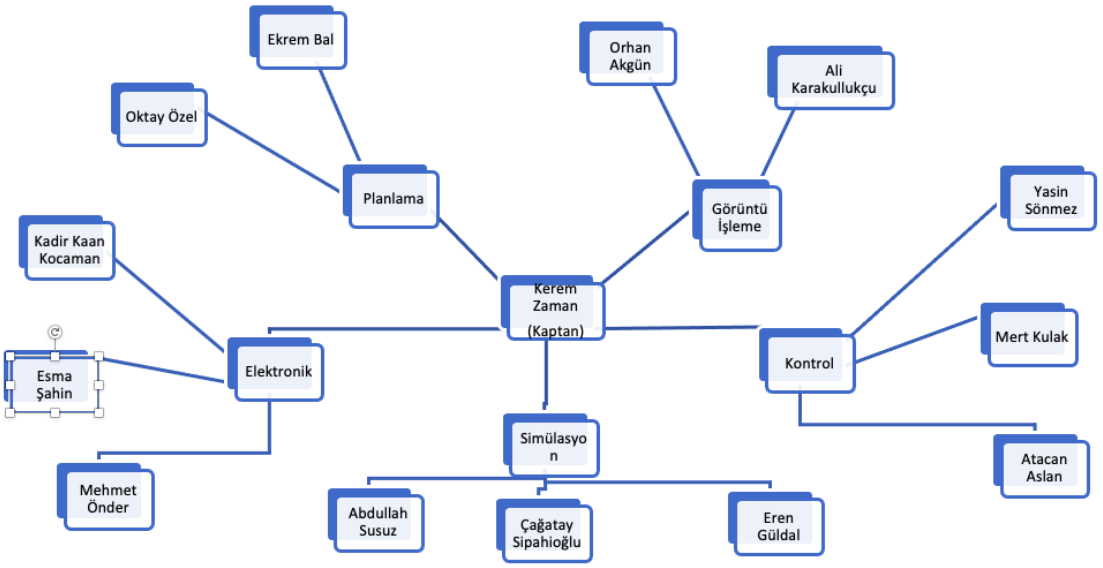
10. Referanslar **38**

1. Takım Organizasyonu

Takımımız ana 3 alt takımdan oluşmaktadır. Mekanik grubunun hedefi geçen sene Efficiency Challenge yarışına katılmış aracımızı kontrol edilebilir hale getirmek ve sistemleri daha güvenli hale getirmektir. Bu amaçla yapılanlar raporun devamında detaylı anlatılmıştır. Makina mühendisliği bölümü öğrencilerinden oluşmaktadır.

Elektronik takımımızın görevleri araç kontrol ünitesini tasarlamak ve aracı "drive-by-wire" hale getirmektir. Elektrik ve elektronik mühendisliği öğrencilerinden oluşmaktadır.
Otonom takımımız bilgisayar mühendisliği ve elektrik elektronik mühendisliği öğrencilerinden oluşmaktadır. Takımımızın otonom grubu lideri Kontrol bölümünden sorumlu Ahmet Fatih Köse'dir. Otonom bölümü kendi içerisinde Simülasyon, Görüntü İşleme, Elektronik ve Kontrol bölümlerine ayrılmaktadır. Ekibin şemada yer verilmeyen kısmı otonom kısma yardımcı olan mekanik ve sponsorluk işleri ile ilgilenmektedir.

Takımdaki üyelerin görev dağılımları şemada detaylı bir şekilde verilmiştir.



Zamanlama planımız Gantt Şeması olarak ifade edilmiştir. Temel olarak amacımız, testlerimizi adım adım yapmaktır.

2. Ön Tasarım Raporu Değerlendirmesi

Ön tasarım raporumuzda ve simülasyonda hedef bulma ve planlama konuusndaki eksikliklerimize yönelik aldığımız yorumları değerlendirdik ve bu eksiklikleri kritik tasarım aşamasında giderme yönünde adımlar attık. Bu başlıktaki incelemelerimizi üç alt başlık altında topladık; değişen kısımlar, eksik olduğumuz kısımlar ve güçlü olduğumuz kısımlar.

Ön tasarım raporuna göre değişen en önemli şey, hedef nokta belirleme algoritmamızdır. Simülasyon aşamasında, sürekli güncellenen hedef nokta planlayıcının gerekli minimum frekansta çalışmamasına yol açtığından hedef nokta bir kere belirlendikten sonra noktaya ulaşana dek güncellenmeyeceği bir çözüm uygulamak zorunda kalmıştık. Sabit hedef nokta çözümü, noktanın yanlış bulunması durumunda sistemi geri dönülemez bir duruma sokuyordu. Bu sebeple kritik tasarım aşamasında adaptif olarak belirlenen hedef noktanın uygunluğunu sürekli kontrol edip sadece gerektiği durumlarda güncelleyen bir algoritma kurduk.

Takımımıza bu süreçte biri simülasyonda ikisi planlamada görevli olmak üzere 3 yeni arkadaşımız katıldı.

Son olarak da görüntü işleme için veri setimiz değişti. Simülasyondan alınan verileri kümemizden çıkardık ve tamamen gerçek veriler ile devam etmekteyiz.

İkinci değerlendirme noktamız ise değerli jüri üyelerinin bize verdiği dönütlerden yola çıkarak eksiklerimizi testip etmektir. Kontrol için kullandığımız denklemlerin de analizlerini daha iyi yaptık ve hedef nokta belirleme algoritmamızı daha detaylı bir biçimde ifade ettik.

Son olarak da güçlü olduğumuz kısımlar değerlendirildi. Bunların en başında simülasyon ile gerçek aracımızın benzerliği ve aracın dinamiklerinin sisteme aktarılmasındaki başarılarımızı göz önüne alarak fiziksel testlerde olası zaman kayıplarını önlemek için farklı senaryoların

denendiği daha kompakt ve işlevleri teker teker ölçen simülasyon test ortamları hazırladık ve yeni algoritmalarımızı önce burada test ettik.

3. Araç Özellikleri

Ön Tasarım Raporundaki araç özellikleri ve ölçümler yeni aracımıza göre güncellenmiştir.

3.1. Fiziksel ve Mekanik Özellikler

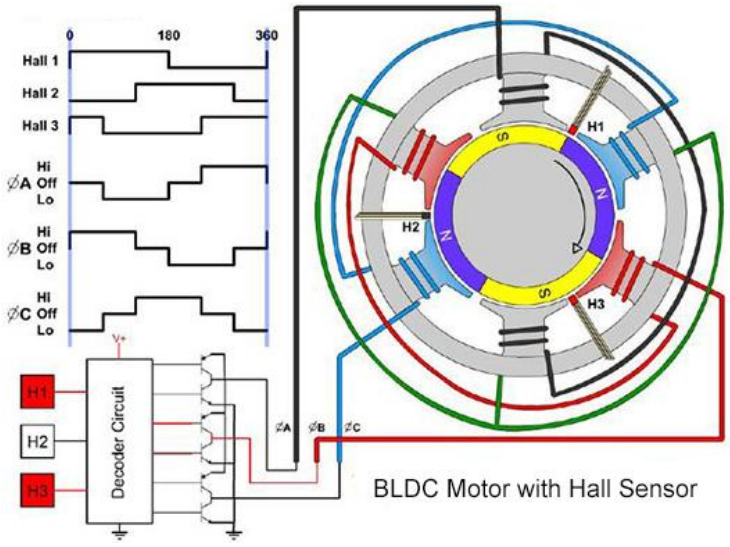
Özellik	**Birim**	**Değer**
Uzunluk	mm	2300
Genişlik	mm	1250
Yükseklik	mm	1100
Teker Sayısı	#	4
Şase	Materyal	Çelik
Kabuk	Materyal	Fiberglas
Ön Teker Çapı	mm	600
Ön Teker Genişliği	mm	100
Arka Teker Çapı	mm	600
Arka Teker Genişliği	mm	100
Motor Tipi	-	BLDC
Motor Gücü	kW	3
Motor Verimliliği	%	88
Batarya	-	Li-ion 18650
Batarya Nominal Voltaj	V	59.2
Batarya Maksimum Voltajı	V	67.2
Batarya Toplam Kapasitesi	Wh	2800

3.2. Kontrol Sistemleri

3.2.1 Motor Sürücüsü

Aracın tüm motorları BLDC (Fırçasız DC Motor) motor olarak seçilmiştir. BLDC motorlar hakkında genel bilgi vermek gerekirse; motorda stator, sargıları taşıyan çok tabakalı yığılmış çelik plakalardan oluşur. Statördeki bu sargılar yıldız modeli ve delta modeli olarak düzenlenebilir. Yıldız model düşük devirde yüksek tork verebilirken, delta model düşük devirde düşük tork verebilir. BLDC motorların rotoru ise kalıcı mıknatıslardır. Uygulama ve kullanım alanına bağlı olarak mıknatısların sayısı değişkenlik gösterebilir. Mıknatıs sayısı arttıkça motordan alınan tork miktarı artarken elde edilen maksimum hız azalmaktadır.

Fırçasız DC motorların çalışma prensipleri, fırçalı motorlarla benzerlik gösterir. Aralarındaki temel farklılık motorların pozisyon geri besleme mekanizmalarıdır. Fırçalı DC motorda rotorun pozisyon geri bildirimi motorun fırçalarından ve mekanik komütatörlerden elde edilirken, fırçasız DC motorlarda rotor pozisyonu sensörler aracılığıyla elde edilir. Bu sensörlerden en yaygın olanı Hall efekt sensörleridir.



Kaynak: bldcmotor.org

Aracın ana itici motorunda da Hall efekt sensörlerinden yararlanılmakta, bu sensörlerden elde edilen çıktı çizgisel hız bilgisine çevrilerek kullanımı sağlanmaktadır. Elde edilen bu hız bilgisi aracın iki farklı yerinde kullanım yeri bulmaktadır. Öncelikle bu hız bilgisi motorun kontrol sistemine geri beslenmekte, böylece motorun hızı istenilen referans değerinde sabit tutulabilmektedir. İkinci kullanım yeri olarak da aracın göreceli konumunun hesaplanmasında kullanılmakta, aracın hızı ve geçen zaman değerlerinin nümerik yöntemlerle entegrasyonu, aracın odometri bilgisinin hesaplanmasını sağlamaktadır. Odometri ve hesaplanması konuları detaylı olarak sensörler bölümünde işlenmiştir.

Aracın motorunun kontrol sistemi olarak, endüstriyel standart olan PID kontrolcü kullanılmış, gerekli parametreler kullanılarak ideal kazanç değerleri hesaplanmıştır. Kontrolcünün girdisi olarak önceki paragrafta bahsedilen Hall efekt sensörlerinden elde edilen motorun çizgisel hız bilgisi kullanılmış, kontrolcünün çıktısı olarak da motor sürücüye iletilen sinyal alınmaktadır. Araçta pedal-by-wire kullanıldığı için, sürücünün pedala basması ya da motor kontrolcüden çıkan sinyal PWM (Sinyal Genişlik Modülasyonu) ile motor sürücüye aktarılmaktadır. Buna göre motor sürücünün girdi voltajı görev döngüsü (duty cycle) oranına göre belirlenmektedir.

3.2.2. Fren Sistemi

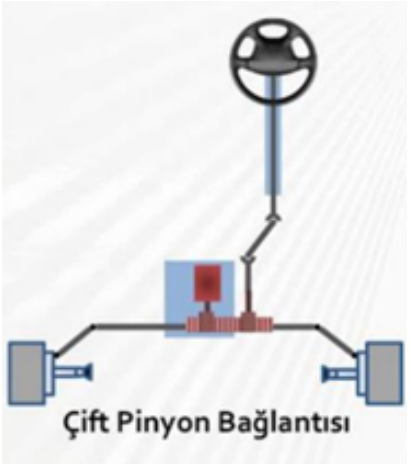
Araçta bulunan ana itici motor sürücüde halihazırda bulunmakta olan, fren esnasında en yüksek enerji verimliliğinin elde edildiği rejeneratif elektrik fren, fren uygulamasında öncelikli seçenek olarak tasarıma dahil edildi. Ancak bu tip fren sistemlerinin temel sorunu olan yeterince güçlü ve ani fren yapamama probleminden dolayı tasarıma hidrolik fren sistemi eklenmesi uygun görüldü.

Hidrolik fren sisteminin en büyük avantajları, güçlü ve ani fren yapabilme kapasitesi, ve olası bir elektronik problemde aracın fren yapabilme yeteneğinden ödün vermeden durma eylemini gerçekleştirebilmesidir. Aynı zamanda araca implementasyonunun da kolay olması dolayısıyla araçta, step motor ile kontrol edilen hidrolik fren kullanılmasına karar verildi.

Fren sisteminde elektronik fren uygulaması (brake-by-wire) temel alınmıştır. Aracın içinde mevcut bulunan sürücü fren pedalına bastığında ya da otonom sürüş kontrolcüsünden fren yapma komutu geldiğinde, bu aksiyonlar fren kontrolcü devresine açma kapama sinyali olarak ulaşmaktadır. Aracın kullanılacağı senaryolar düşünülerek en optimum yöntemin, fren komutunun aracı tamamen durdurmak için kullanılması gerektiği sonucuna ulaşılmıştır. Bu yöntem gereğince, fren kontrolcüsü freni, aracı durduracak kuvvette uygulamakta, araç için hareket sinyali gelmediği sürece de fren uygulamasına devam etmektedir.

3.2.3. Direksiyon Sistemi

Araç direksiyon mekanizması için kullanılan Kremayer ve 2 adet pinyon dişlileri sisteminin merkezini oluşturur. Bu dişlilerden biri doğrudan araç direksiyonuna, biri de direksiyon step motoruna bağlıdır. Bu tip bir sistemde kullanılabilecek değişkenler: Step motor torku, pinyon dişlileri yarıçapı, Kremayer modül sayısı olarak sıralanabilir.



Kaynak: TUBITAK Efficiency Challenge Sunumları

Direksiyonu çevirmek için yapılan kuvvet ölçümleri sonucu, gereken kuvvetler araç statik durumda iken 120 - 180 Newton, araç hareket halinde iken 60 – 90 Newton olarak hesaplanmıştır. Bu değişim zeminin statik ve dinamik sürtünme katsayılarının farkından kaynaklanmaktadır. Kullanılan Kremayer modül sayısı, 20'den 50'ye kadar pinyon dişlisinin kullanımına izin vermektedir. Pinyon dişlerinin yarıçapı 2 - 5 cm arası değişmektedir. 4 cm'lik pinyon dişlileri için tork hesaplaması:

$$0.04\,m \times 100\,N = 4\,N.m$$

Bu hesaplamalar sonucunda, 22mm yarıçapında pinyon dişlisi ve 8.5 N.m tork değerine sahip step motorunun kullanılmasında karar kılındı.

Direksiyon sisteminde kullanılan step motorunun sürülmesi için PID kontrolcü kullanıldı. Buna göre kontrolcüye girdi olarak, otonom sürüş kontrolcüsünden gelen direksiyon açı değeri (referans girdisi) ve lineer potansiyometre kullanılarak elde edilen o anki direksiyon açısı değeri (geri besleme değeri) verildi. Kontrolcünün çıktısı olarak da elektronik direksiyon (steer-by-wire) işlevselliği kullanılarak, PWM (Sinyal Genişlik Modülasyonu) değerinde hesaplanan sinyal değeri alındı. Otonom sürüş kontrolcüsünden veya araçta mevcut bulunan sürücünün direksiyonu döndürmesi sonucu alınan bu PWM değerindeki sinyal, step motor sürücüsüne yönlendirilerek step motorun ön tekerleri istenilen değere yönlendirmesi sağlandı.

3.2.4. Geri Bildirim

Hız geri bildirimi, BLDC motorlarda bulunan Hall efekt sensörlerinden alınmaktadır. Bu sensörler bize 1.8 cm'ye ve motor tekerinin 5 derece dönüşüne kadar hassasiyetle bize bilgi vermektedir.

Direksiyon yönelim geri bildirimi, Kremayer'e bağlanan lineer potansiyometre aracılığıyla ölçülmektedir. Bu değer direksiyon motor kontrolcüsüne geri beslenmekte, böylece step motor ve dişli sistemlerden kaynaklanan hatalardan sistem korunmaktadır.

Fren motorunun geri bildirimi fren motoruna yerleştirilen potansiyometre aracılığıyla elde edilmekte, böylece tehlike anında düzgün çalışması kritik öneme sahip olan fren sistemi hatalardan arındırılmaktadır.

3.2.5 Uzaktan Haberleşme Sistemi

Yarışma kurallarınca, aracın görevine başlaması ve acil durum durdurma işlevi kablosuz haberleşme sistemi üzerinden yapılmalıdır. Bunun için güncel olarak kullanılan kablosuz haberleşme protokolleri karşılaştırılmış, bunların avantajları ve dezavantajları değerlendirilmiştir.

Technology	Frequency	Data Rate	Range	Power Usage	Cost
2G/3G	Cellular Bands	10 Mbps	Several Miles	High	High
Bluetooth/BLE	2.4Ghz	1, 2, 3 Mbps	~300 feet	Low	Low
802.15.4	subGhz, 2.4GHz	40, 250 kbps	> 100 square miles	Low	Low
LoRa	subGhz	< 50 kbps	1-3 miles	Low	Medium
LTE Cat 0/1	Cellular Bands	1-10 Mbps	Several Miles	Medium	High
NB-IoT	Cellular Bands	0.1-1 Mbps	Several Miles	Medium	High
SigFox	subGhz	< 1 kbps	Several Miles	Low	Medium
Weightless	subGhz	0.1-24 Mbps	Several Miles	Low	Low
Wi-Fi	subGhz, 2.4Ghz, 5Ghz	0.1-54 Mbps	< 300 feet	Medium	Low
WirelessHART	2.4Ghz	250 kbps	~300 feet	Medium	Medium
ZigBee	2.4Ghz	250 kbps	~300 feet	Low	Medium
Z-Wave	subGhz	40 kbps	~100 feet	Low	Medium

Kaynak: www.postscapes.com

Araç kablosuz haberleşmesi dizaynının temel amaçları:

- Pist içerisinde araca ulaşabilecek menzilde (100 metre civarı)
- Düşük hata oranlı
- Anlık iletişimi sağlayabilecek kararlı ve güvenilir
- Düşük bilgi yükü
- Enerji tüketimi araç enerji sistemi tarafından tolere edilebilir

- Düşük maliyetli

Bu amaçlar değerlendirildiğinde, aracımıza en uyumlu olacak protokolün; uzak menzilli, düşük bit hızlı, ücretsiz lisanslı LoRaWAN olduğuna karar verildi. Modülümüz 433 Mhz frekans bandında çalışmaktadır.

3.3. Elektronik Hareket Sistemleri

3.3.1. Elektronik Gaz Pedalı (Steer-by-wire)

Araç gaz pedalı ve motor sürücüsü PWM modülasyonuyla araç motor sürücüsüne bağlıdır. Motor sürücü, bu sinyali gerekli torka dönüştürmektedir. PWM sinyali aracın gaz pedalına bağlı bir potansiyometre ile alınabildiği gibi 0-5 Volt arası bir sinyal veren bir mikrokontrolcü ile de verilebilmektedir.

3.3.2. Elektronik Direksiyon (Steer-by-wire)

Aracımızda kullandığımız direksiyon sistemi Kremayer ve pinyon dişliler sistemiyle kontrol edilmektedir. Direksiyonun yönelimi ve direksiyon step motor sürücüsünün yönetilmesi için Sinyal Genişlik Modülasyonu (PWM) tekniği kullanılmaktadır. Buna göre motor sürücüyü yöneten voltaj değeri, verilen “görev döngüsü (duty cycle)” yüzdesine göre belirlenmektedir.

3.3.3. Elektronik Fren (Brake-by-wire)

Araçta kullanılan fren sistemi, fren kontrolcüsüne gelen komutun, aç-kapa yöntemi ile değerlendirilmesi ve fren motoru tarafından frenin uygulanması ile gerçekleşmektedir. Otonom sürüş kontrolcüsünün ya da araç sürücüsünün fren komutu vermesi sonucu motor kontrolcüsü gereken PWM sinyalini step motor sürücüsüne aktarmakta, böylece istenilen hız ve konum PWM sinyalinin görev döngüsü oranı belirlenerek kontrol edilebilmektedir.

4. Sensörler

4.1. Odometri

Odometri, mobil robotlarda robotun konumu ve yönünün bir referans noktasına göre hesaplanması işlemidir. Bunun için başlangıç noktası koordinat sisteminin orijini olarak kabul edilir, başlangıç yönü ise koordinat sisteminin x ekseni olarak kabul edilir. Robotun hareketi sonrası bulunacağı konum ve yön bilgileri bu başlangıç varsayımları üzerinden tanımlanır.

Yapılan araştırmalar sonucu bu konum ve yön bilgilerinin elde edilmesi için motorlara bağlı Hall Effect Sensörleri ile direksiyonun dönüş açısını algılayabilecek bir lineer potansiyometrenin kullanılmasına karar kılınmıştır. Bu sensörler kullanılarak robotun güncel

konum ve yön bilgileri ile bu bilgilerin güncel değişimleri elde edilebilir. Bu bilgileri kullanarak da robotun gelecek konum ve yön bilgileri tahmin edilebilir ve gelecek konum ve yön bilgileri istenen şekilde olması için kontrol edilebilir.

Odometre hesaplamaları için Ackermann Direksiyon Geometrisi baz alınmıştır. Bu geometri kapsamında araç herhangi bir dönüş yaparken, öndeki iki tekere ve arkadaki tekerlere dik 3 adet doğrunun kesiştiği bir nokta etrafında döner. Bu prensip kapsamında aracın güncel hızı, direksiyonun dönüş açısı ve şasi ölçüleri kullanılarak aracın ne yönde gideceğini hesaplamak mümkündür. Bu hesap çizilen rota yönünde olacak şekilde kontrol edilir.

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} s\cos\theta \\ s\sin\theta \\ \frac{s}{l}\tan\varphi \end{bmatrix}$$

Yukarıda verilen denklemlerde; **x** durum vektörünü, **ẋ** hız vektörünü, x ve y aracın hareket noktasına göre koordinatlarını, θ aracın koordinat sistemindeki yönelimi; s aracın anlık çizgisel hızı, l aracın ön ve arka tekerleri arasındaki uzaklığını, φ ise direksiyonun dönüş açısını gösterir. Direksiyonun dönüş açısı lineer potansiyometre tarafından radyan cinsinden ölçülür.

Aracın direksiyonunun dönüş açısının hata payı ihmal edilebilir oranlarda olduğu için $tan(\varphi) = \varphi$ olarak kabul edilebilir.

Buna göre ikinci denklem aşağıdaki şekilde düzenlenebilir:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} s\cos\theta \\ s\sin\theta \\ \frac{s}{l}\varphi \end{bmatrix}$$

Aracın güncel konumu (x_k)ile bir sonraki konumu (x_{k+1}) arasındaki ilişki ise aşağıdaki şekilde tanımlanır. Denklemdeki dt ifadesi ise sensör ölçümleri arasında geçen birim zamanı ifade eder. Pratikte ise 0.01 saniye gibi düşünülebilir.

$$\mathbf{x}_{k+1} = \mathbf{x}_k + dt \times \dot{\mathbf{x}}$$

Hesaplamalarda kullanılan çizgisel hız değeri motordaki Hall Effect Sensörü ile elde edilebilir. Bu sensörler içerisindeki enkoderler aracılığıyla, tekerin her 15° derecelik dönüşü için sayaçlarını 1 arttırmaktadır. Her birim zamandaki (dt) enkoder sayacındaki artış miktarı sayesinde tekerin anlık açısal hızı hesaplanabilir. Bu değer arka tekerin yarıçapı (0.215 metre) ile çarpılarak çizgisel hıza dönüştürülür.

4.2. Atalet Ölçü Birimi (IMU):

IMU modülü (Atalet Ölçü Birimi) aracın anlık ivmesini, açısal hızını ve 3 boyutlu uzaydaki oryantasyonunu ölçmeye yarayan bu işlevselliği nedeniyle de örneğin havasal araçlarda ve birçok mobil robotta yaygın olarak kullanılan bir yönelim ve ivme hesaplama sensörüdür.

Aracın yön ve ivme bilgilerinin elde edilebilmesi için, 3 eksen ivme-ölçer, 3 eksen jiroskop ve 3 eksen manyetometre içeren 9 serbestlik dereceli, 16 bit Analog/Dijital çeviriciye sahip MPU-9250 IMU sensörü kullanılmıştır. Bu sensör üzerindeki jiroskop ve manyetometreyi kullanarak aracın referans yönüne göre olan dönmesini ölçebilir. Bu ölçümün Ackermann geometrisinde kolaylıkla kullanılması adına sensör arka teker ekseni üzerine yerleştirilmiştir.

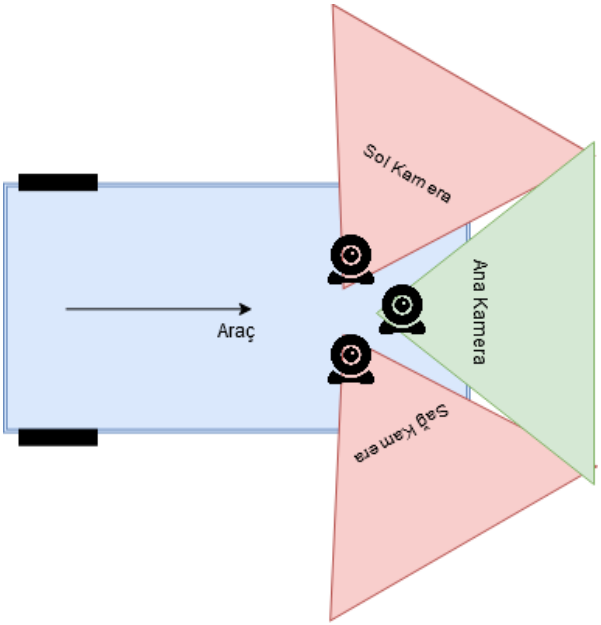
Ayrıca jiroskoptan 40 Hz frekansında elde edilen açısal hız (rad/s) değerlerinin integrali alınarak da açısal yön bilgisi elde edilmiştir.

IMU'daki manyetometre ise bu yön bilgilerini, aracın Dünya'nın manyetik alanıyla yaptığı açıyı hesaplamak üzere doğrulamak için kullanılmıştır. Bu iki sensör aracılığıyla güvenilir bir yön verisi elde edilmiştir.

Son olarak da bu tarz elektromekanik jiroskoplarda sıkça görülen kayma hatalarının düzeltilmesi için sensör başlatıldığında 3 saniye boyu hareket etmeden jiroskop verileri toplanmış, bu veriler doğrultusunda jiroskopun kayması hesaplanmış ve bu değer ileriki ölçümlerden çıkarılmıştır.

4.3. Kamera

Aracın trafik levhaları, şeritler gibi çevresel verileri alması için 3 adet kamera kullanılmaktadır. Her kamera 78 derecelik yatay ve 90 derecelik dikey görüş açısına sahiptir. Aracın tam önüne bakan ana kamera trafik levhalarının ve şerit işaretlerini görmekle görevlidir. 60 derece sağ ve 60 derece sola bakan destekleyici kameralar ise yine trafik levhalarının ve ana kamera ile görülemeyebilecek park işaretleri ile kavşakların görülmesi amacıyla kullanılmaktadır. Kamera verileri, öznitelik çıkarımı, yapay öğrenme ve derin öğrenme vasıtasıyla eğitildi ve işlendi.
Aracımızın kamera konfigürasyonu aşağıdaki figürde gösterildiği gibidir.



4.4. 2D LIDAR

Son olarak 2 boyutlu bir LIDAR sensörü aracın etrafındaki engel ve cisimlerin algılanması için kullanıldı. Parkurun sınırlarını belirleyen bariyerler bu sensör sayesinde algılandı ve işlendi. LIDAR sensörleri prensip olarak gönderdikleri ışık ışınlarının civardaki cisimlerden sekip dönme sürelerini hesaplama prensibiyle çalışmaktadır. Bizim aracımızda ise Rplidar S1 kodlu LIDAR modeli ise bu prensip ile 360 derece, 0.1-40m lik bir alandaki cisimlerin algılanmasında kullanıldı. Bu sensör 0.391 derece açısal çözünürlük ile her 100ms'de uzaklık ölçüm verisi elde etmemizi sağladı. Bu çıktılar aracın kontrolü, rota planlaması ve acil duruşu için kullanıldı.

5. Araç Kontrol Ünitesi

Araç kontrol ünitesinin temel işlevi, aracın otonom sürüşü sağlayan bilgisayardan aldığı kontrol komutlarını aracın aktüatörlerine iletmek ve aynı zamanda aracın sağlık kontrollerini yapmak, bu sağlık bilgilerini haberleşme sistemi ile bir merkeze kayıt etmektir.

Efficiency Challenge raporu için grafiğini çizmiş bulunduğumuz kontrol ünitesinin en ortasında bir adet Arduino Mega bulunuyor. Bu kontrolcü iki adet BLDC motor sürücüsü, batarya kontrol ünitesi, telemetri ünitesi ve ekranı kontrol ediyor. Aynı zamanda Arduino-ROS kütüphaneleri sayesinde kontrol bilgisayarından aldığı bilgileri kontrol ünitelerine aktarıyor.

5.1 Kontrol Mimarisi

Arduino Mega'da kesme tabanlı (interrupt driven) bir mimari kurduk. Bu mimari gereğince her bir kesmenin belli bir önem sırası var ve bu önem sırasına göre ana kontrolcü tepki veriyor.

Sistem dışında ayrıca bir mikro-kontrolcü tarafından denetlenen bir fren sistemi daha mevcut. Acil durumlarda LoraWan haberleşme altyapısı ile haberleşen bu sistem araca dışardan acil durum freni uygulanmasını mümkün kılıyor.

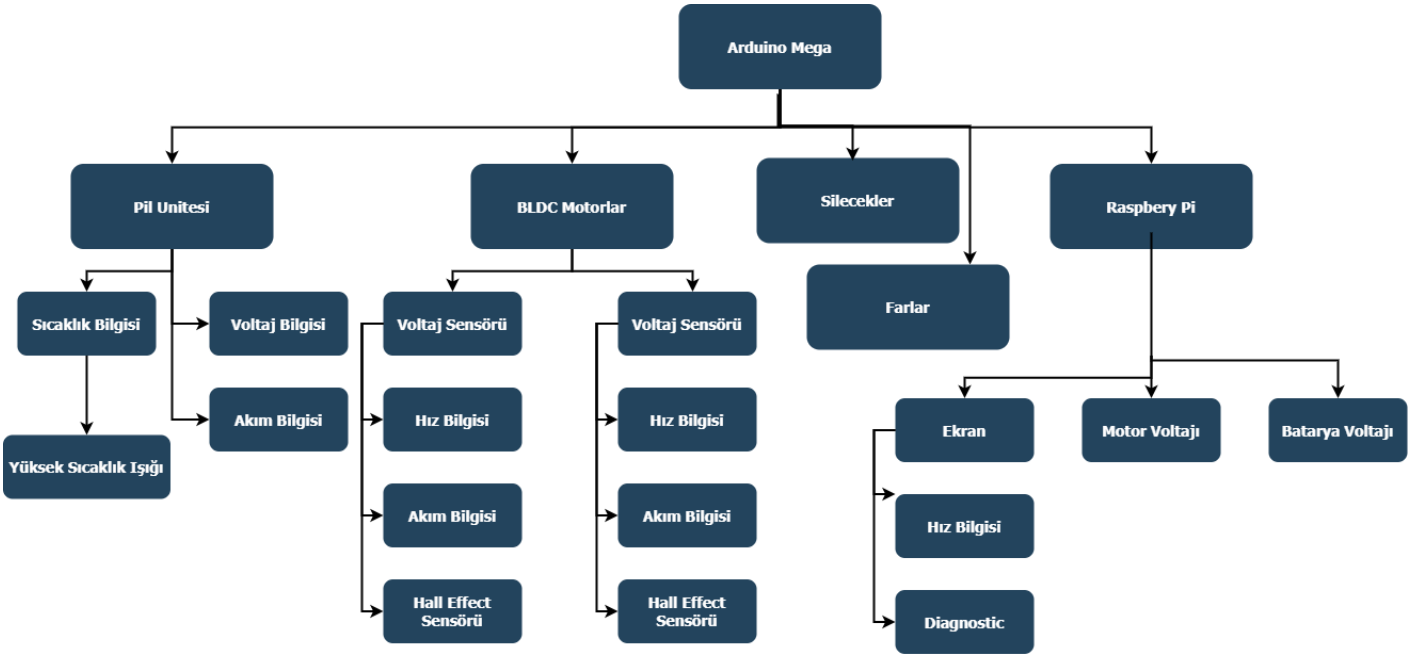
Kesmelerin önem sırası ise, sırasıyla

- fren
- batarya sıcaklık uyarısı
- direksiyon
- gaz kontrolü
- ekran

fonksiyonlarına cevap verecek şekilde kurgulandı. Herhangi bir kesme gelmez ise sistem aracın rutin işlerini yapıyor.

5.2 Haberleşme Mimarisi

Modüler bir yapıda olan sistemimizde 5 noktada haberleşme kullanılıyor.



A. Otonom Kontrol Bilgisayarı ile USB Seri Haberleşme

ROS-Arduino kütüphanesi kullanılarak yapılıyor. Fren, gaz ve direksiyon komutları için ön ve son kodlar (header, footer) tanımlanmış durumda ve

mesajlarına içeriğine göre bir ISR üretiliyor. USB kablosu ile arduino ana bilgisayara bağlanmış durumdadır.

B. Sistem Üniteleri arasında I2C Seri Haberleşme

Fren, direksiyon ve batarya ünitesinin her birinde ayrı kontrolcüler bulunuyor ve bu kontrolcülerle iletişim kurulması için I2C protokolü kullanılıyor.

C. Kontrol Merkezi ile Tek Yönlü Kablosuz Haberleşme.

Bu haberleşmenin amacı; hız, batarya sıcaklığı, batarya durumu vb bilgilerin bir kontrol merkezi tarafından takip edilmesine imkan veren, sadece araçtan dışarıya doğru bir haberleşme biçimidir. Bu iş için ESP2866 WiFi modülü kullanılmıştır. Bununla birlikte açık kaynak kodlu Blynk kütüphanesi ile merkez istasyonunda çalışan bir uygulama yazılmıştır.

D. Ekran Kontrolü için Raspberry Pi ile Seri Haberleşme

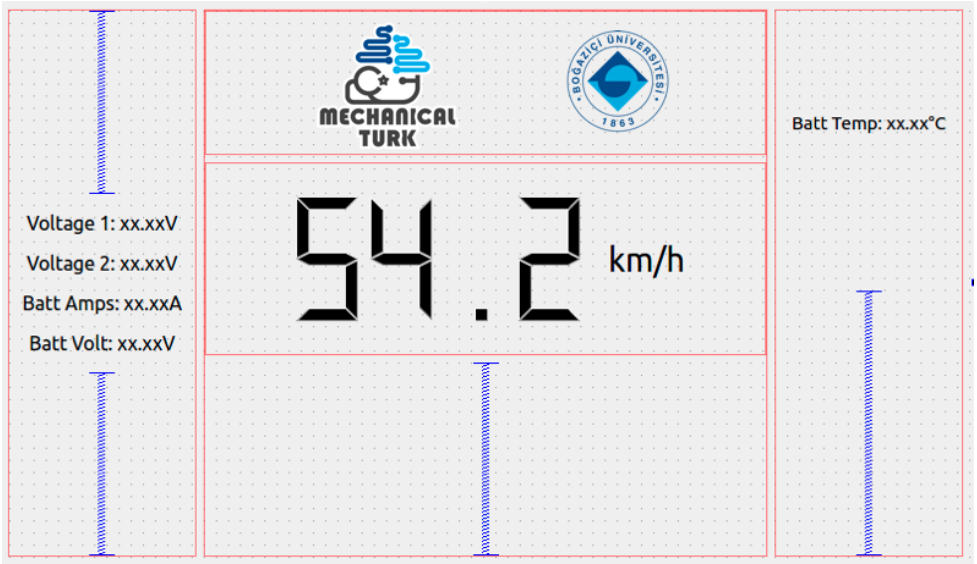
USART ile haberleşiliyor. Belli komutlara karşılık ekrana araç durum bilgisi aktarılıyor.

E. Acil Durum Fren Komutu

Araca dışarıdan verilebilen tek komuttur. Sadece frenlere bağlı olan mikro-kontrolcüyü harekete geçirerek aracın acil durumlarda durmasını sağlar.

5.3 Ana Ekran

Bu bölüm geçen sene Efficiency Challenge için tasarlamış olduğumuz ekran hakkındadır. Hala aynı ekranı kullanmaktayız. QT platformu ile Raspberry Pi üzerinde çalışan bir ekran tasarladık. Aracın kritik bilgilerini kullanıcıya göstermek için kullanılıyor.



6. Otonom Sürüş Algoritmaları

6.1. Görüntü İşleme

BURST olarak, aracımızın daha da otonom çalışması için görüntü işleme yöntemleri yoğun olarak kullanılmıştır.Bu amaç doğrultusunda görüntü işleme yöntemleri kullanılarak gelişme olması planlanan iki temel alan, trafik işaretlerinin tanınması ve yol/şerit takibidir. Bunu yapmak için ayrı iki sistem tasarlanıp araca ROS modülleri olarak dahil edilmiştir.

6.1.1. Trafik İşareti Tanıma

Trafik işaretlerinin tanınması görevinde kullanılan model derin öğrenme tabanlıdır. Trafik işaretlerini tespit edecek modelin geliştirilme sürecindeki kriterler, gerçek zamanlı tespit ve işaret tespit modelindeki başarımdır. Derin öğrenme tabanlı modelimiz için gerekli olan veriyi elde etmek için BURST simülasyon ortamından yararlanılmıştır. Kritik tasarım aşaması içinse kendi topladığımız gerçek dünya fotoğraflarıyla beraber katkıda bulunduğumuz İTÜ TTVS veri seti kullanılmıştır. Kendi topladığımız veriler için verinin işaretlenmesinde renk eşiği, morfolojik operasyonlar ve şekil çıkarımı yöntemleri,işaretleme işlemlerini kolaylaştırmak için kullanılmıştır. Verinin işaretlenmesi sürecinde açık kaynaklı bir imge işaretleme aracı kullanılmış olmasıyla beraber , üzerinde geliştirme yapılmış olup BURST simülasyon veri seti için kullanıma uygun hale getirilmiştir. Derin öğrenme modeli C programlama dilinde ve Darknet kütüphanesinden yararlanarak yazılırken, veri işaretleme aracı Python dilinde geliştirilmiştir.

6.1.1.1. Simülasyon Trafik İşareti Otomatik Veri İşaretleme

BURST aracımızın simülasyonda kullanması için geliştirilen Trafik İşaret Tanıma Modeli, derin öğrenme tabanlı bir model olması nedeniyle işaretlenmiş verilere yüksek miktarda ihtiyaç duymaktadır. Hem yüksek veri ihtiyacını karşılayabilmek hem de işaretleme işlemini kolaylaştırmak için iki aşamadan oluşan bir veri işaretleme yöntemi kullanılmıştır:

1. Simülasyon ortamındaki araç üzerinden alınmış görüntüler üzerinde renk eşiği, morfolojik operasyonlar ve şekil çıkarım yöntemlerinden yararlanarak otonom şekilde belirlenmesi ve el ile kategorilendirilmesi
2. Otomatik yapılan işaretlemelerde hataları bulup, düzeltebilmek için bir işaretleme aracı geliştirilip bu araç kullanılarak verinin elle işaretlenmesinin sağlanması

BURST simülasyon ortamı, yarışma ortamına üst düzey benzeyerek simüle etmek amacıyla tasarlansa da gerçek dünyada karşılaşılan imge üzerindeki gürültü miktarına oranla çok daha azdır. Bu gürültünün az olması araç üzerindeki kameradan elde edilen görüntünün temel görüntü işleme yöntemleriyle işlenmesi gereksinimini ortaya çıkarmaktadır. Simülasyon

içerisinde kullanılmış olan yapay öğrenme modelleri yüksek başarıma ulaşsa da, aracımızın gerçek dünya senaryolarında derin öğrenme modeli kullanacağı göz önünde bulundurularak, simülasyonda da derin öğrenme modeli kullanılmıştır. Bu amaç doğrultusunda, simülasyondaki araç üzerinde bulunan kamera ile imgeler kaydedilmiş, derin öğrenme modelinde kullanılmak üzere işaretlenmiştir.

Veri işaretleme kısmı hem yorucu hem de uzun bir süreç olduğundan dolayı bu adımı hızlandırmak için belli yöntemlere kullanılmıştır. Yukarıda bahsedilen adımlardan birincisi olan görüntü işleme ile otomatik işaretleme adımları aşağıda özetlenmiştir.

Renk eşiklerinin net olarak belirleyebilmek için kameradan gelen BGR (Blue, Green, Red, OpenCV kütüphanesi imgeleri BGR formatında) imgeler HSV (Hue, Saturation, Value) renk uzayına tahvil edilmiş, renk eşikleri bu renk uzayında belirlenmiştir. Trafik işaretlerini tespit edebilmek için aşağıdaki gibi 5 farklı renk eşik değeri belirlenmiştir:

1. Beyaz Renk Eşiği:

Beyaz renk eşiği olarak HSV değerleri (0, 0, 235) ve (0, 0, 255) aralığı belirlenmiştir.

2. Siyah Renk Eşiği:

Siyah renk eşiği olarak HSV değerleri (0, 0, 0) ve (0, 0, 5) aralığı belirlenmiştir.

3. Kırmızı Renk Eşiği:

Kırmızı renk eşiği filtresine ait iki değer aralığı belirlenmiştir. birinci kısım için HSV değerleri (0, 100, 100) ve (10, 255, 255) aralığı kullanılmıştır. İkinci kısım için ise HSV değerleri (160, 100, 100) ve (179, 255, 255) aralığı belirlenmiştir. Aşağıdaki renk uzayını temsil eden görselden kırmızı eşik değerinin neden iki kısımdan oluştuğu anlaşılabilmektedir. Öyle ki, HSV renk uzayında kırmızı rengi temsil eden Hue değerleri OpenCV kütüphanesi içerisinde 0 ile 10 derece arasında ve 160 ile 180 derece olmak üzere iki aralık arasında

bulunmaktadir.

4. Sarı Renk Eşiği:

Sarı renk eşiği olarak HSV değerleri (20, 100, 100) ve (40, 255, 255) aralığı belirlenmiştir.

5. Yeşil Renk Eşiği:

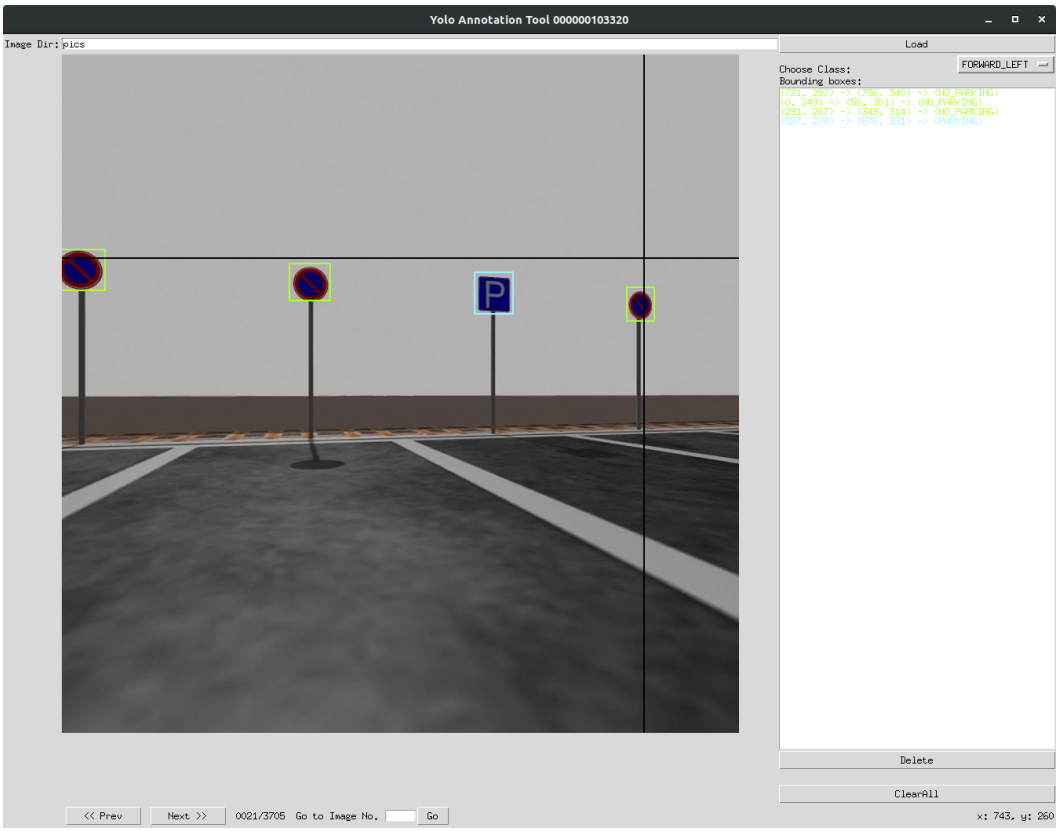
Yeşil renk eşiği olarak HSV değerleri (50, 100, 70) ve (70, 255, 255) aralığı belirlenmiştir.

Yukarıda belirtilen renk eşikleri uygulandığında imge üzerinden bir maske elde edilmiş olur. Elde edilen maskeye sırasıyla yayma (dilation) ve aşındırma (erosion) yöntemleri uygulanarak nesne sınırlarının belirlenmesi amaçlanır. Belirtilen bu morfolojik operasyonların uygulanmasının beraberinde, imge üzerindeki nesnelerin tespiti amacıyla şekil çıkarımı yöntemleri uygulanmış olup belirlenen bir alan büyüklüğünün üzerindeki bölgeler nesne olarak sınıflandırılmıştır.

Bu yöntemle belirlenen trafik işaretleri el ile sınıflandırılmış ,hızlı bir şekilde ön işleme gerçekleştirilmiştir. Simülasyon ortamında elde edilen bir imge ve bu imge üzerinden yukarıdaki yöntemler ile ulaşılan bir trafik işareti aşağıdaki şekillerde örneklendirilmiştir.

6.1.1.2. Simülasyon Veri Setinin Elle İşaretlenmesi

Otomatik ön işaretlemenin sonrasında veri setinin kalitesinin artırılması hedeflendiğinden manuel olarak işaretleme yapılması aşamasına geçilmiştir. Bu işaretlemede, temelinde Python programlama dili ve TKinter kütüphanesi kullanılan açık kaynaklı bir işaretleme aracı geliştirilmiştir.İşaretleme için kullanılan bu aracımız açık kaynaklı olarak https://github.com/tunahansalih/Yolo-Annotation-Tool adresinden erişilebilir. Aşağıdaki resimde geliştirilen işaretleme aracından elde edilen bir ekran görüntüsü verilmiştir.



İşaretleme aracımız, sırasıyla girdi olarak verilen imgeleri işaretlememiz için göstermekte ve işaretlenen her bir imgeden sonra imge üzerindeki elle girilen trafik işareti ve ışıklarını içeren sınırlayıcı kutu bilgilerini bir yazı dosyasına kaydetmektedir. Simülasyon ortamında işaretlenmek üzere 18 sınıf belirlenmiştir. Bu 18 sınıfın 15'i trafik işareti, 3'ü ise trafik ışıklarından oluşmaktadır:

1. Giriş olmayan yol
2. Taşıt trafiğine kapalı yol
3. İleri ve sola mecburi yön
4. İleri ve sağa mecburi yön
5. İleriden sola mecburi yön
6. Hız sınırlaması sonu (20 km/saat)
7. Azami hız sınırlaması (30 km/saat)
8. Azami hız sınırlaması (20 km/saat)
9. İleriden sağa mecburi yön
10. Sağa dönülmez
11. Sola dönülmez
12. Dur
13. Park etmek yasaktır

14. Park Yeri
15. Durak
16. Yeşil Işık
17. Sarı Işık
18. Kırmızı Işık

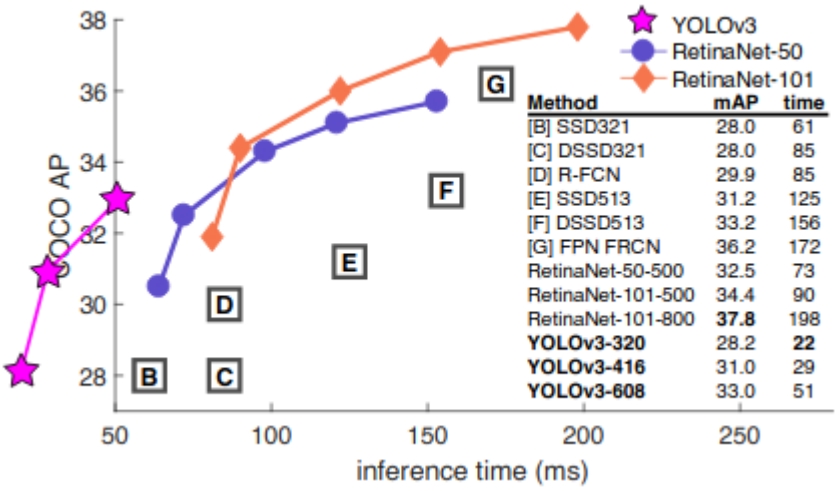
İşaretleme aracından yararlanılarak 1000 adet imge kolaylıkla işaretlenmiştir. İşaretlemiş olduğumuz her bir imge için bu nesneye denk gelen sınırlayıcı kutu bilgileri, ayrı ayrı her bir imge için oluşturulan bir dosyaya aşağıda görülen biçimde, her satıra bir nesne bilgisi gelecek şekilde yazılmıştır.

<Sınıf İndeksi> <X Koordinatı> <Y Koordinatı> <Genişlik> <Yükseklik>
<Sınıf İndeksi> <X Koordinatı> <Y Koordinatı> <Genişlik> <Yükseklik>
<Sınıf İndeksi> <X Koordinatı> <Y Koordinatı> <Genişlik> <Yükseklik>
…

Bu dosya biçimi YOLO nesne tanıma modelinin kullandığı format olması sebebiyle herhangi bir ön değişiklik işlemine maruz bırakılmadan direkt olarak model eğitimine dahil edilebilmektedir. Kullanılan X koordinatı, Y koordinatı, genişlik ve yükseklik verileri 0 ile 1 arasında değişmekte olup çevreleyen kutunun imgeye göre konumunu ifade etmektedir. X koordinatı ve Y koordinatı sınırlayıcı kutunun orta noktasının bulunduğu konumunu ifade ederken yükseklik ve genişlik değerleri ise sınırlayıcı kutu boyutlarının imgenin boyutlarına göre olan oranını göstermektedir.

6.1.1.3. Simülasyon Trafik İşareti Tanıma Modeli

Burst simülasyon ortamından elde edilen veri seti, YOLOv3[6] tabanlı bir derin öğrenme modeli kullanılmıştır. YOLO modeli tek aşamalı bir nesne tespit modeli olup yüksek hızlarda yüksek başarım gösterdiği için tercih edilmiştir. YOLOv3 modelinin diğer derin öğrenme tabanlı modeller ile kıyas tablosunu aşağıda görebilirsiniz (Kaynak: [6]).



Derin öğrenme tabanlı nesne tanıma sistemleri genel olarak iki kısımda incelenmektedir:

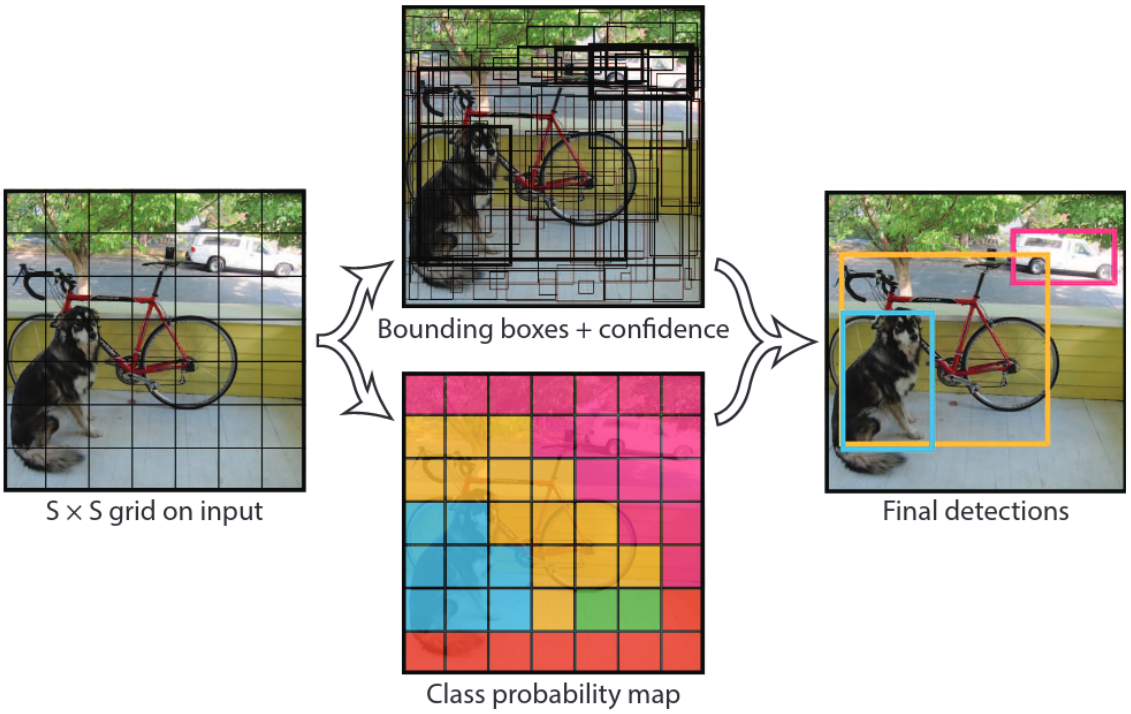
1. Tek aşamalı nesne tanıma modelleri
2. İki aşamalı nesne tanıma modelleri

Tek aşamalı nesne tanıma modelleri, daha hızlı çıkarım yapabilirken, iki aşamalı nesne tanıma sistemleri daha yüksek başarımlı çıkarım yapmaktadırlar. İki aşamalı modellerin hız kaybı, bu modellerin genel olarak nesne tespiti ve sınıflandırma için iki farklı mimari içermesinden kaynaklanmaktadır. Bu modellerin en yoğun kullanılanları R-CNN[7], Fast R-CNN[8] ve Faster R-CNN[9] modelleridir. Bu modeller temel olarak bir nesne tespit mimarisi ve bir nesne sınıflandırma modelinden oluşmaktadır. Tek aşamalı modeller ise nesne tespit ve nesne sınıflandırma için tek bir mimari kullanmaktadır. Tek bir mimarinin kullanılması çıkarım süresini düşürürken başarımı düşürmektedir. Aracımızın gerçek zamanlı nesne tespiti yapabilmesi için çıkarım süresi hayati önemde olduğu için tek aşamalı modellerden YOLOv3 tercih edilmiştir.

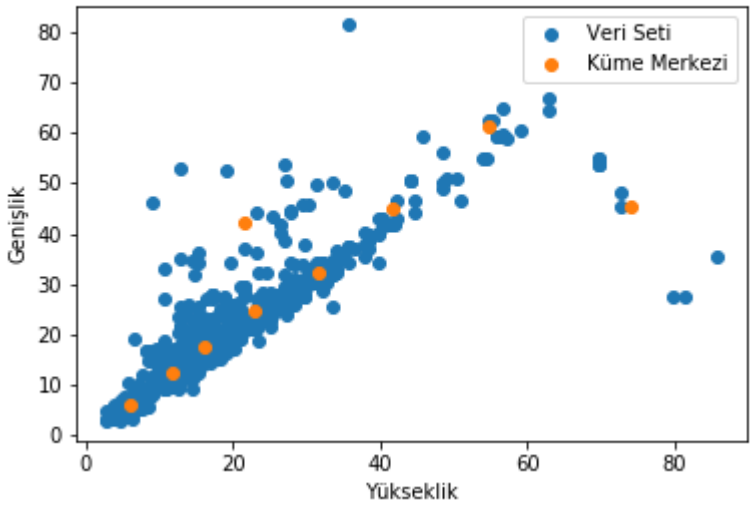
YOLOv3 mimarisinde, Burst simülasyon ortamı üzerinde trafik işareti tespiti yapabilmek için temel olarak iki değişiklik yapıldı:

1. Nesne sınıflandırmayı doğru yapabilmek için derin öğrenme modelinin son katmanındaki nöron sayısı değiştirilmiştir.
2. Nesne tespitini daha doğru yapabilmek için Burst simülasyon veri seti üzerinden elde edilen nesne boyutları kullanılarak çapa boyutları değiştirilmiştir.

YOLOv3 modeli, imgeyi ızgaralara bölerek her bir ızgara için sınıf olasılıkları ve sınırlayıcı kutu boyutları çıkarımı yapmaktadır. Sınırlayıcı kutu boyutları çıkarımı yapmak için YOLO modeli ön tanımlı sınırlayıcı kutular kullanmaktadır. Bu sınırlayıcı kutu boyutları, veri setinden öğrenilmiştir. YOLO mimarisine ait örnek bir çıkarım aşağıda gösterilmiştir(Kaynak [10]).



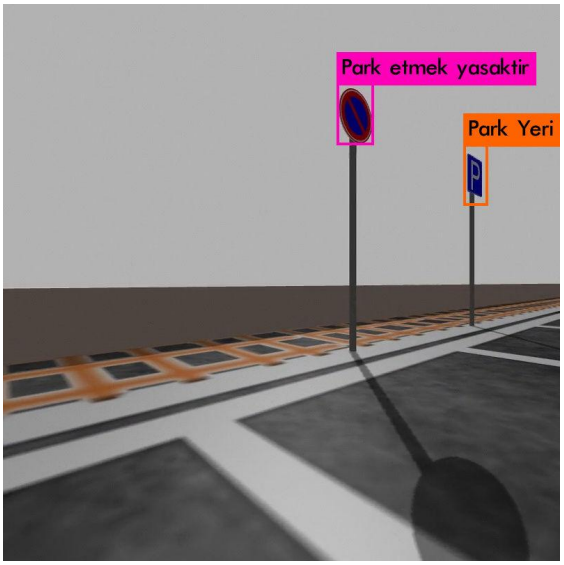
Trafik işareti tespit modelinde kullanılacak ön tanımlı sınırlayıcı kutu boyutları, veri setindeki sınırlayıcı kutular üzerinde K-Means gözetimsiz kümeleme algoritması kullanılarak elde edilmiştir. Aşağıdaki grafikte, veri setinden elde edilen yükseklik ve genişlikler ile K-Means algoritması ile elde edilen sınırlayıcı kutular gösterilmiştir.



Simülasyon ortamında YOLOv3 modeli kullanılarak yapılan çıkarım örnekleri aşağıda gösterilmiştir.

Sola donulmez
Ileri ve saga
Aza

Yesil Isik
Saga donulm
Giris olmayan yol


Park etmek yasaktir
Park Yeri

6.1.1.4. Gerçek Dünya Trafik İşareti Veri İşaretleme

Simülasyon veri seri işaretleme aşamasında geliştirdiğimiz bir çok otomatik işaretleme tekniği, gerçek dünya koşullarında istenilen başarıya ulaşmamakta ve otomatik işaretleme sonrasında gerçekleştiridiğimiz manuel kalite kontrol aşamasını daha da zorlaştırmaktadır. Bu nedenle, simülasyon veri seti için geliştirdiğimiz otomatik işaretleme aşamasını atlayarak gerçek dünya veri setinin tamamen manuel bir şekilde kendi geliştirdiğimiz açık kaynak anotasyon aracı kullanılarak işaretlenmesine karar verilmiştir.

Gerçek dünya veri setimiz iki ana veri setinden oluşmakta:

1. Takımımız bünyesinde edindiğimiz trafik ışık ve işaretleriyle çeşitli ışıklandırma ve çevre koşullarında oluşturduğumuz veri seti.

2. “UYANIK” aracıyla DRIVESAFE projesi kapsamında İstanbul’un çeşitli bölgelerinde araç içerisine yerleştirilmiş bir kamera vasıtasıyla elde edilmiş videolardan oluşan veri seti[12]

Kendi veri setimize ek olarak TEKNOFEST takımlarının kolektif çabasıyla oluşturulmuş İTÜ TTVS veri setini de kullandık.

6.1.1.5. Gerçek Dünya Trafik İşareti Tanıma Modeli

Simülasyon ortamı için geliştirdiğimiz trafik işareti ve ışığı tanıma modelimiz aracımızdaki ana bilgisayar üzerinde hız ve hafıza isterlerimizi yeterince karşıladığı için bu model üzerinde bir değişikliğe gidilmeyecektir.

6.1.1.6. Trafik İşaret Konumunun Tespiti

İşaretlerin bulunduğu konumu yaklaşık olarak hesaplanırken işaretlerin yüksekliğinin sabit olduğu ve parkur eğiminin ise ihmal edilebilir olduğu varsayılmıştır. Bu varsayımlar temel alınarak geometrik eksenlerde araç doğrultusunun x, sol doğrultunun y kabul edildiği 3 boyutlu koordinat sistemindeki işaretlerin yerinin tespiti aşağıdaki gibi hesaplanmıştır:

$$s_\theta = FOV_x(P_x/2 - s_{px})/P_x\, rad$$

$$s_x = \frac{P_y/2}{(P_y/2 - s_{py})}\, x\, \frac{h_{işaret} - h_{camera}}{tan(FOV_y/2)}$$

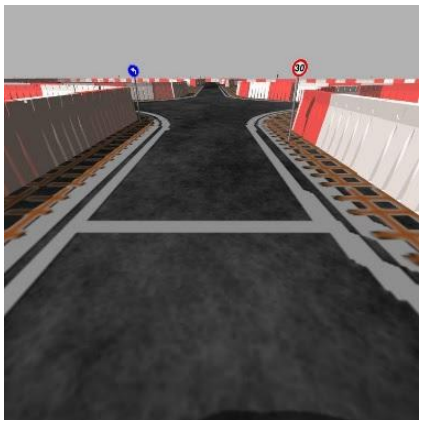
$$s_y = s_x tan(s_\theta)$$

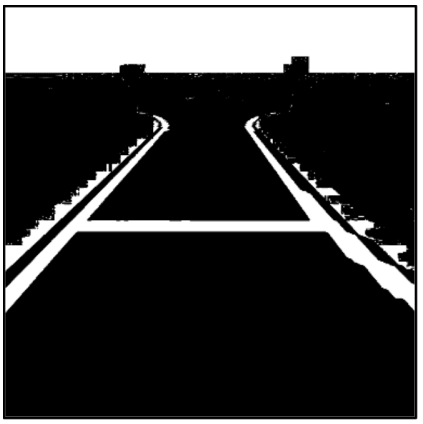
Buradaki s_θişaretin görece açısını, $s_{x,y}$ ise işaretin aracın sistemindeki görece konumu ifade eder. Kameraya has özellikler olan $FOV_{x,y} = 1.39\, rad$, $P_{x,y} = 800\, pixel$ olarak belirlenmiştir. Buradaki$(P_y/2 - s_{py})$ değeri imge üzerinde tespit edilen işaretin imge merkezinden trafik işaretine kadar olan pixel cinsinden yüksekliğidir. Bu değer işaretin araçtan uzaklığı arttıkça yükseklik farkı sabit kalacağından,işaretin araçtan uzaklığıyla ters orantılı olarak değişmektedir. $\frac{(h_{işaret} - h_{camera})}{tan(FOV_y/2)}$terimi ise işaretin kamera görüş alanından çıkacak kadar yakın olduğu noktanın x doğrultusundaki uzaklığıdır.

6.1.2. Şerit Tanıma Algoritması

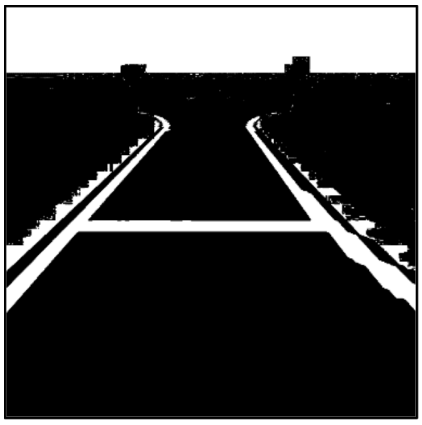
BURST Aracının simülasyon içerisindeki yol şeritlerini tanıyabilmesi için OpenCV, Numpy ve scikit-learn kütüphaneleri kullanılmış, çeşitli imge işleme yöntemlerinden ve denetimsiz öğrenme algoritmalarından yararlanılmıştır.

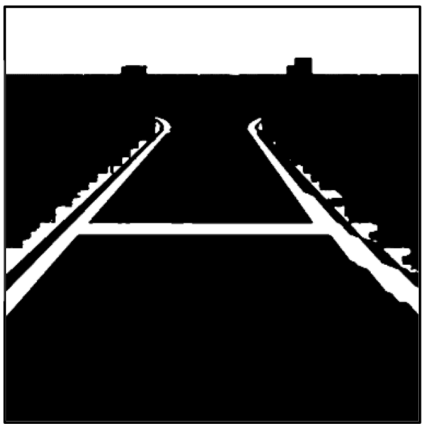
İlk olarak simülasyon kamerasından alınan imge RGB tonlarında olduğu için ortamdan alınan veri, lineer olarak daha kolay ayrıldığı HSV renk uzayına çevrilmiştir.HSV renk uzayında renk eşiği uygulamaları daha kolay yapılabilmektedir.Yeni imge üzerine uygulanan beyaz renk eşiği ile şerit olabilecek olan alanlar belirlenmektedir.Bu işlemlerin uygulandığı imgenin normaldeki görüntüsüyle, elde edilen imge maskesi aşağıda gösterilmiştir.



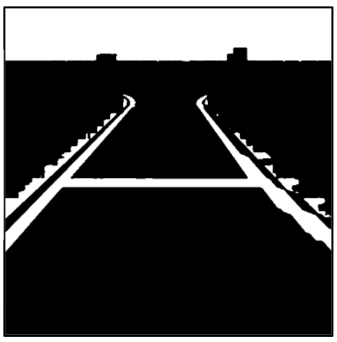


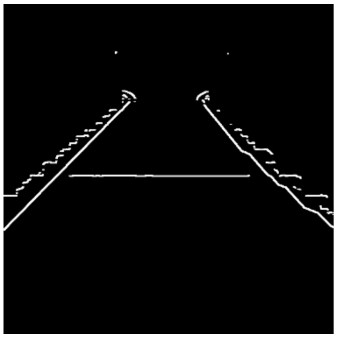
Elde edilmiş olan maskede birçok piksel boyutunda gürültü görülmekte olup, bu gürültü medyan gürültü filtresi uygulanıp azaltılmıştır.

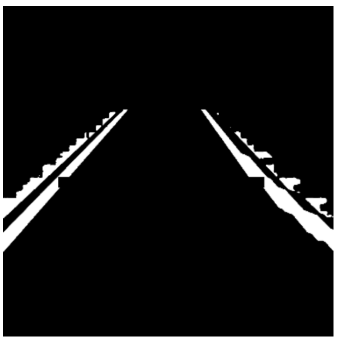




Sonraki aşama olarak imgedeki bitiş çizgisininin algılanabilmesi amacıyla x ekseni üzerinde Sobel filtre uygulanmıştır.Sobel filtresinde aracın gidiş yönüne dik bir çizgi tespit edilirse bu çizgi bitiş çizgisi olarak algılanır.Bitiş çizgisi olarak algılanan çizgi de imge üzerinden çıkarılarak yeni bir imge elde edilir.







Elde edilen siyah-beyaz imgeden perspektif dönüşüm matrisinin çıkarılmasıyla kuşbakışı görüntü elde edilir. Perspektif dönüşüm matrisini hesaplarken kaynak imgeden seçilen 2 paralel doğru üzerindeki 4 noktanın koordinatları ve o noktalara karşılık çıktıda olması istenilen koordinatlar kullanılarak gerekli homografi matrisi hesaplanmış, imge dönüştürülmüştür. Hesaplanan homografi matrisi, otonom aracın kamerasından alınan görüntünün kuşbakışı görüntüsünün çıkarmak amacıyla kullanılmıştır. Homografi matrisi şu şekilde hesaplanır:

$$Kuşbakş\,mge\,Koordinatlar \;=\; Homografi\,Matrisi\,x\,Kamera\,Koordinatlar$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Bu denklemde görülen homografi matrisindeki h_{33} değeri her zaman 1 olarak alınır. Bu matris eşitliğinden çözülebilmesi için aşağıdaki denklemler elde edilir:

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + 1}$$

$$y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + 1}$$

Yukarıdaki denklemleri tekrar düzenlenince şu denklemler elde edilir:

$$h_{11}x + h_{12}y + h_{13} - h_{31}xx' - h_{32}yx' = x'$$

$$h_{21}x + h_{22}y + h_{23} - h_{31}xy' - h_{32}yy' = y'$$

Homografi matrisinde elde edilmesi gereken 8 bilinmeyen değer, denklemi çözebilmenin mümkün olması için en az 8 denkleme ihtiyaç doğurur. Eğer kamera girdisi ve buna karşılık gelen kuşbakışı görüntünün koordinatları bilinirse homografi matrisinin değerlerini bulabileceğimiz 2 adet denklem elde edileceğinden, homografi matrisindeki 8 bilinmeyenin bulabilmesi için kamera görüntüsü ve kuşbakışı görüntüde karşılıklı 4 adet koordinat noktasının bilinmesi yeterli olur.

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x'_1 & -y_1x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y'_1 & -y_1y'_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2x'_2 & -y_2x'_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2y'_2 & -y_2y'_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3x'_3 & -y_3x'_3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -x_3y'_3 & -y_3y'_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x_4x'_4 & -y_4x'_4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -x_4y'_4 & -y_4y'_4 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ x'_3 \\ y'_3 \\ x'_4 \\ y'_4 \end{bmatrix}$$

Burada elde ettiğimiz matris denkleminin çözülmesi ile homografi matrisindeki bilinmeyen değerler elde edilmiştir.
Bu matrisin kullanılmasıyla elde edilen kuşbakışı görüntüye DBSCAN gözetimsiz kümeleme algoritması kullanıldığında sağ ve sol yol şeritleri belirlenebilir.

6.1.2.1 DBSCAN Algoritması

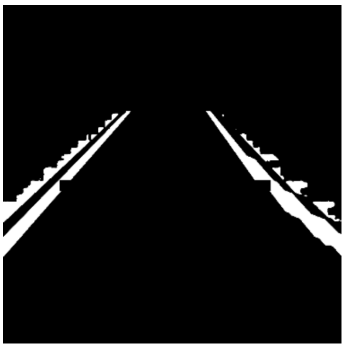
Tasarladığımız bu algoritma simülasyon ortamında yüksek başarım ile çalışsa da gerçek dünya ortamında gürültünün de artmasıyla karşılaşacağımız zorluğun artacağını düşünüyoruz. Bu nedenle derin öğrenme tabanlı LaneNet[13] benzeri şerit tanıma modelleri ile de denemelerimiz devam etmektedir.
DBSCAN algoritması [1] yoğunluk-temelli bir kümeleme algoritmasıdır. maksimum mesafe ε ve minimum nokta sayısı k olmak üzere bu algoritma aşağıdaki şekilde açıklanabilir [2]:

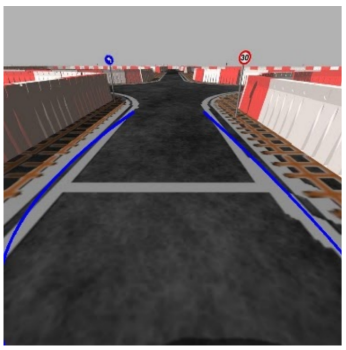
- Eğer bir p noktası kendisine ε mesafe uzaklıkta k noktaya sahipse p, bir çekirdek nokta olarak adlandırılır.
- Eğer bir q noktası p noktasından en fazla ε mesafe uzaklıkta ise q noktası p noktasından *doğrudan erişilebilirdir*.
- Eğer $p_1 = p$ ve $p_n = q$ ve her bir p_{i+1} noktası p_i noktasından doğrudan erişilebilir olmak şartıyla $p_1, ..., p_n$ diye tüm p noktalarının (q hariç) çekirdek nokta olduğu bir yol varsa, q noktası p noktasından *erişilebilirdir.*
- Eğer bir nokta hiçbir noktadan erişilebilir değilse o nokta gürültü noktasıdır.
- Böylelikle bir çekirdek noktası olabilen bir p noktası kendisinden erişilebilir olan tüm noktalarla bir küme oluşturur.

Dönüştürülen imgemizde 2 grup halinde yayıldığını gördüğümüz beyaz pikseller $\varepsilon = 100$, $k = 2$ seçilerek DBSCAN kümeleme algoritması ile kümelere bölüştürülmüştür. Daha sonrasında her bir kümeye ait olan pikseller için ayrı ayrı olarak yapılan polinomsal regresyon ile bu şeritleri en iyi şekilde temsil eden polinomik ifadeler çıkarılmıştır. En sonunda da bu polinomların ortalaması alınarak yolun ortası, yani takip edilmek istenen patikanin polinomu çıkarılmıştır.

DBSCAN algoritmasına girdi olarak bütün beyaz pikseller verildiğinde kümelere bölme işlemini otonom sürüşü aksatacak kadar uzun sürdüğü görülmüş ve bu durumun önüne geçebilmek için Canny Kenar Algılama algoritmasıyla [5] şeritlerin sadece kenarlarını temsil eden pikseller belirlenmiş ve bu seçilen pikseller DBSCAN algoritmasına verilmiştir. Bu değişikliğin yapılmasıyla DBSCAN algoritmasının 10 kat kadar hızlandırıldığı görülmüştür.

DBSCAN uygulandığında belirlenen sağ ve sol şerit pikselleri, üzerine bir polinomun oturtulmasıyla bu şeritler bir polinom ile temsil edilir.Kuşbakışı dönüşüm ve DBSCAN algoritmasıyla elde edilen sağ ve sol şerit aşağıda resimlerde görülebilmektedir.





6.2.Kontrol

Bu bölümde simülasyon isterlerini sağlayan MPC (Model Predictive Control) algoritması tabanlı bir kontrolcü ve patika belirleme süreci tanıtılmıştır.

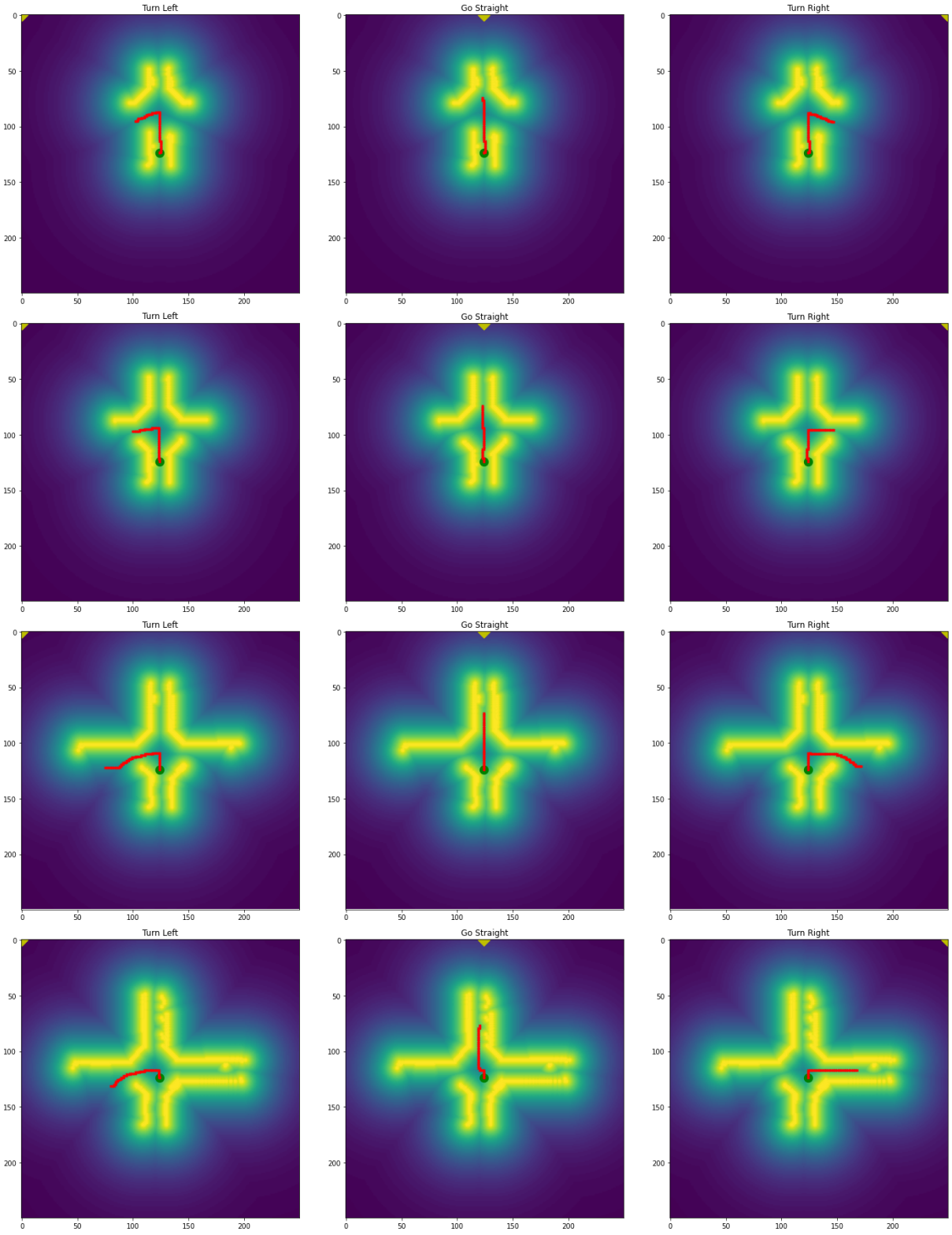
6.2.1. Hedef Nokta Belirleme

Hedef nokta belirlemek için ROS "Navigation" kümesinden "Costmap_2D" paketi kullanarak çevrenin haritası oluşturulur. Bunun için önce LIDAR verisi kullanılarak 2 boyutlu "doluluk haritası" çıkarılır ve bu harita "bölgesel harita" olarak adlandırılır . Bu harita üzerinde aracın gerçek ölçüleri de göz önüne alınarak bir "enflasyon katmanı" eklenir. Araç, bölgesel olarak hesaplanan doluluk haritası yanı sıra hafızasında daha önce geçtiği yollardaki bölgesel haritaları kullanarak "genel harita" da oluşturmaktadır.

Hedef noktayı belirlemek için 2 durum düşünülmüştür: Düz gitme ve dönme durumları. Hedef nokta, "genel harita" üzerinden belirlenen duruma özel olarak hesaplanan "sezgisel maliyet" ve "doluluk haritası" verilerini kullanarak maliyeti en az noktalar üzerinden giden patika bulunup bu patikanın belirli bir mesafe sonrasında kesilmesiyle bulunur. Bu yöntem A* algoritmasına benzemektedir.

Sezgisel maliyet, eğer araç sol tarafa dönme durumundaysa haritanın en sol ve ileri tarafı hedef nokta olarak belirlenip "Manhattan mesafesi" hesaplanarak bulunur. Eğer araç düz gitme durumundaysa aracın ilerisindeki, haritanın sonundaki nokta hedef nokta olarak belirlenip aynı yöntemle sezgisel maliyet bulunur. A* algoritması bu hedef nokta için bu sezgisel harita ve doluluk haritası kullanılarak çalıştırılmaya başlanır. Eğer başlanılan noktadan belirli bir mesafe uzaklaşılırsa algoritma durur ve en son ulaşılan nokta hedef nokta olarak belirlenir.

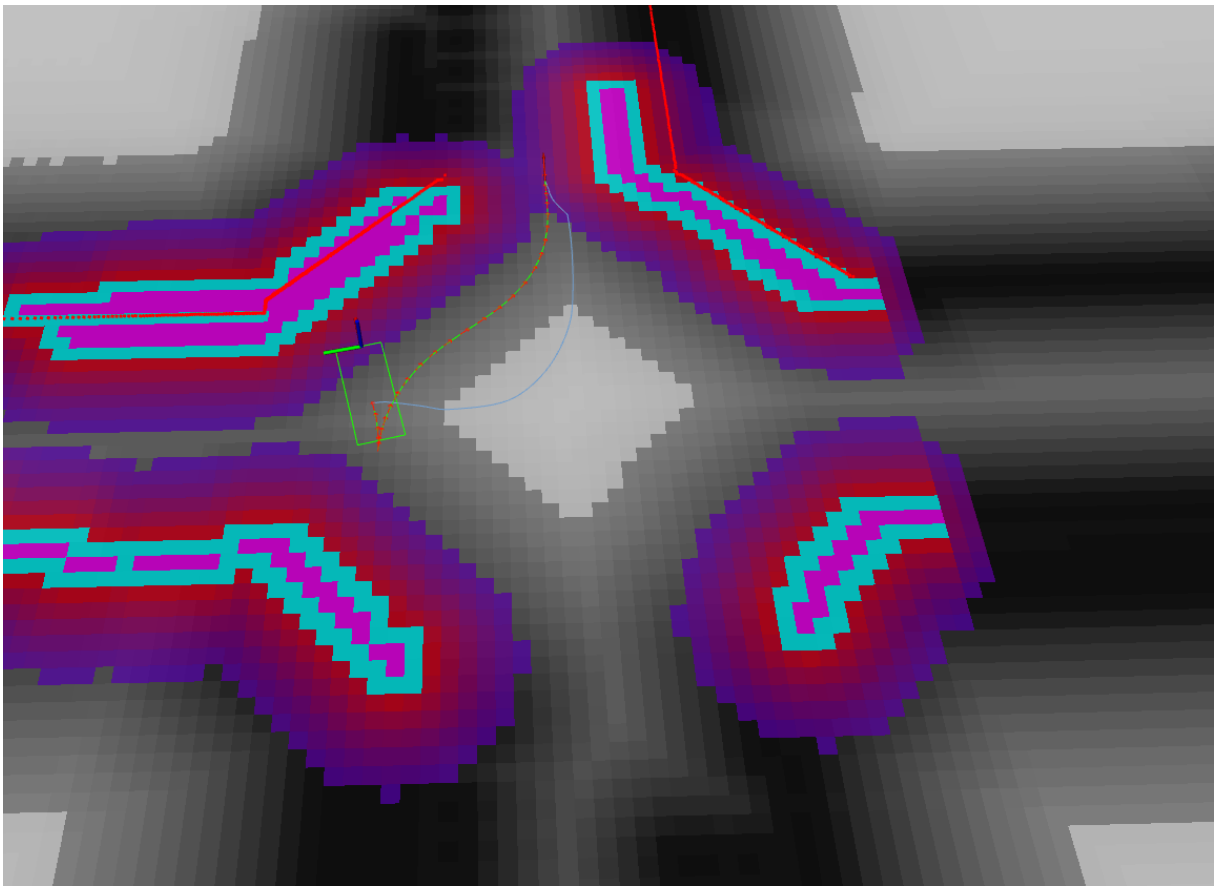
Aşağıdaki resimde 4 farklı senaryo için ve sola dönme, düz gitme ve sağa dönme durumlarında hesaplanan hedef noktaları görülmektedir.



6.2.2. Rota Takibi

Patika oluşturmak için ROS “Navigation” kümesinden “MPC Local Planner” (Rosmann et al.) paketi kullanılarak hedef noktaya olan patika çıkartılır. Herhangi bir hedef nokta verildiğinde araç genel harita üzerinden en kısa patikayı hesaplar ve ardından bölgesel harita

kullanılarak hedef noktaya giden en kısa patikayı takip edecek şekilde engellere çarpmadan giden optimal patika "MPC Local Planner" paketi sayesinde bulunur. Aşağıdaki şekil LIDAR verisi kullanılarak oluşturulan bölgesel harita ve enflasyon katmanı ile bunlara uygun olarak hesaplanan patikayı göstermektedir. Kırmızı noktalar LIDAR verisini, Mor alanlar bölgesel haritayı ve gri alanlar ise genel haritayı temsil etmektedir. Mavi patika genel patikayı yeşil patika ise bölgesel patikayı göstermektedir.



6.2.3. Rota Takibi Algoritması Detayları

Rota takibi sırasında tespit edilen patika noktalarını verilen araç dinamiğine en uygun şekilde takip etmek için hız, dönüş açısı, ivme gibi değerleri hesaplayan MPC tabanlı bir algoritma kullanılmıştır.

Model olarak tek izli doğrusal olmayan araç modeli (Bisiklet Modeli olarak da bilinmektedir.) tabanlı bir model seçilmiştir. Bu model $u(t)$ile ifade edilmekte olup t anındaki durum değişkenlerine göre t+1 anındaki değerlerin nasıl olacağını modellemektedir. Tespit edilen patika noktalarına en yakın şekilde geçen bir polinom $f\ (x)$ ile ifade edilmektedir. (x_t, y_t) aracın t anındaki konumu, ψ_t aracın t anındaki dönüş açısı, $\epsilon_t$dönüş açısındaki sapma, v_t aracın hızı, a_t aracın ivmesi, $\theta_t$aracın t anındaki doğrultusu, cte_tt anındaki *cross track* hatası,

L_f ise aracın önünden ağırlık merkezine olan uzaklığı olmak üzere $u(t)$ ve $u(t+1)$aşağıda gösterilmiştir.

$$u(t) = \left[x_t\ y_t\ \psi_t\ v_t\ a_t\ cte_t\ \epsilon_t\right]^T,$$

$$u(t+1) = \left[x_{t+1}\ y_{t+1}\ \psi_{t+1}\ v_{t+1}\ a_{t+1}\ cte_{t+1}\ \epsilon_{t+1}\right]^T,$$

$$x_{t+1} = x_t + v_t + cos(\psi_t).\ \Delta t\ ,$$

$$y_{t+1} = y_t + v_t + sin(\psi_t).\ \Delta t\ ,$$

$$\psi_{t+1} = \psi_t + \frac{v_t \theta_t}{L_f}.\ \Delta t\ ,$$

$$v_{t+1} = v_t + a_t.\ \Delta t\ ,$$

$$cte_{t+1} = f\ (x_t) - y_t + v_t sin(\epsilon_t).\ \Delta t\ ,$$

$$\epsilon_{t+1} = \epsilon_t - f'(x_t) + \frac{v_t \theta_t}{L_f}.\ \Delta t$$

Polinom katsayılarını model tahmini ve referans patikaya en uygun olacak şekilde ayarlamak için N adet referans noktası için *cross deviation* hatası (L_{CDE}), doğrultu hatası ($L_{heading}$), hız hatası (L_{speed}); cte , ϵ ve v ifadelerinin model çıktısı ve referans değerleri arasındaki fark alınarak hesaplanır ve önceden belirlenen ağırlık faktörleri ile çarpılarak minimize edilir. Hesaplama aşağıda gösterilmiştir.

$$L_{CDE} = W_{cte}\ (cte_{model} - cte_{ref})^2$$

$$L_{heading} = W_{\epsilon}\ (\epsilon_{model} - \epsilon_{ref})^2$$

$$L_{speed} = W_v\ (v_{\ model} - v_{ref})^2$$

$$min \sum_{\tau=t}^{t+N} L_{CDE} + L_{heading} + L_{speed}$$

6.2.4. Yolcu Alma/Bırakma

Yolcu alma noktasına giden yolun sabit olacağı belirtildiğinden işaret tespiti ve uygun biçimde rota takibi bu yolcu alma ve bırakma görevleri için yeterli olmaktadır. Araç harita üzerinde gidilmesi mümkün olan bir sonraki noktayı hedef nokta olarak seçer ve τ_{la} mesafe kalıncaya kadar Bölüm 6.2.1.'de anlatılan yöntemi kullanarak bu noktaya ulaşmaya çalışır. Sonrasında ulaşılması mümkün olan bir sonraki noktayı seçerek bu noktaya doğru hareketine

devam eder. Bu durum araç durak işareti ile karşılaşana kadar devam eder. Durak işaretinin tespit edilmesi durumunda işaretin hizasında 30 saniye bekledikten sonra göreve devam edilir.

6.2.5. Park Alanına Ulaşma

Aracın ikinci durak işaretinde yolcuyu bırakmasının ardından araç park alanına ulaşma davranışına geçiş yapar. Bu davranışın temel mekanizması, trafik işaretlerini ve şeritleri doğru tanıyıp takip ederek park alanına ulaşan güzergahı takip etmek ve park alanına giriş yapmaktır. Bu davranışın doğru çalışabilmesi için yolcu bırakma alanından itibaren doğru yönergelerle takip edilen güzergahın park alanına ulaşması gerekmektedir.

6.2.6. Park Etme

Araç park alanına girişinden itibaren $\tau_{search} = 20\,m$ uzaklıkta bulunan park alanlarına dik bir biçimde düşük sabit hızla hareket etmeye ve kurallara uygun park edebileceği bir park alanı aramaya başlar. Park edebileceği uygun bir alana dik yönde $\tau_{turn} = 7\,m$ kadar yaklaşıldığında aşağıdaki koşulun sağlanması durumunda hareketine başlar.

$$|c_{px} - c_x| < \frac{\tau_{turn}}{sin(\theta_0 - c_\theta)} - \frac{\tau_{turn}}{tan(\theta_0 - c_\theta)}$$

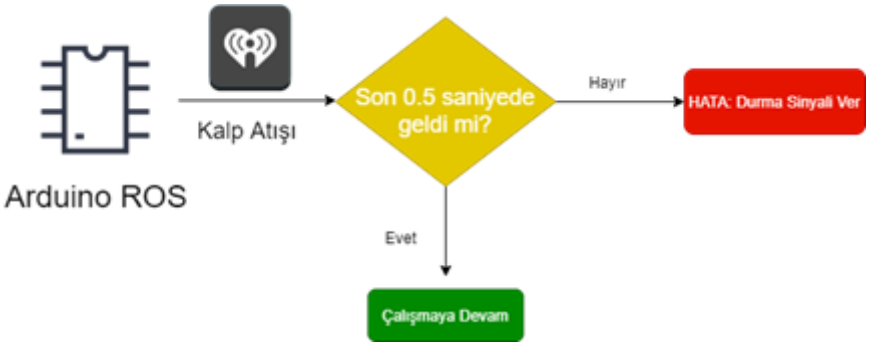
Bu ifadede $\theta_0$aracın başlangıç doğrultusu ile park alanına giriş doğrultusu arasındaki açı, c_{px} ise park edilecek noktanın x koordinatıdır. Simülasyon koşulları için $\theta_0$değerinin $\pi/2$rad olması gerektiği belirlenmiştir. Bu koşulların sağlanmasından sonra park edilecek alanın konumu planlayıcıya hedef konum olarak verilir, plan oluşturulur ve araç bu planı takip eder.

7. Özgün Bileşenler

Özgün bileşenler güncellenen şekliyle ön tasarım sunumundaki ile aynıdır. Buna göre, özgün bileşenler 8 başlık altında değerlendirilebilir.

7.1. Periyodik Yazılımsal Kalp Atışı

Araç sensörlerinden gelen verilerde kayıp ve aksama gibi istenmeyen durumların tespit edilebilmesi için, ana bilgisayar bölümünde periyodik olarak sensör kontrolleri yapılmaktadır. Buna göre bir sensörden gelen verinin zamanının 0.5 saniyeden daha önce olması durumunda kontrolcü güvenlik önlemleri gereğince ani duruş durumuna geçerek gazı kesmekte ve fren yapmaktadır. Böylece elektronik vb. sorunlar sebebiyle sensör iletişimi kesildiğinde olası kazaların önüne geçilebilmesi amaçlanmıştır.



7.2. Araç Görüntüleme Sistemi

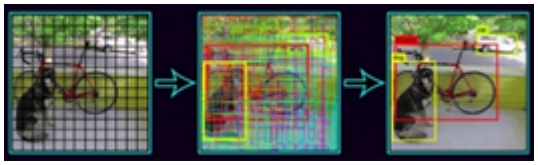
Aracın görüntüleme ihtiyacı iki farklı yoldan karşılanmaktadır. Bunlardan ilki, bir JSON objesi yardımı ile aracın kontrolcüsü üzerinden gerçekleştirilen görüntüleme; ikincisi de araç üzerinde fiziksel olarak bulunan monitör aracılığıyla gerçekleştirilen görüntüleme işlemidir..

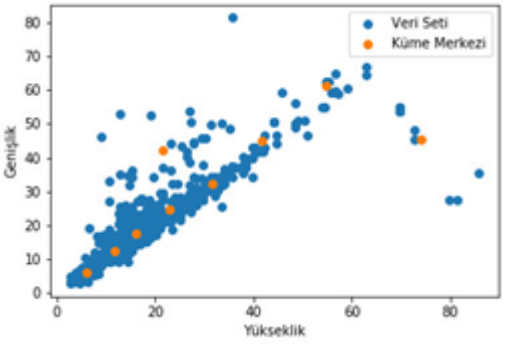
Aşağıdaki görselde araç monitörünün görünüşünün örnek değerlerle beraber bir örneği verilmiştir.

7.3. YOLO Kmeans Gruplama

Trafik işaret ve ışıklarının tanınmasında kullandığımız YOLO nesne tanıma modelinin başarımını artırılabilmesi amacıyla Kmeans Gruplama yönteminden yararlanıldı. Trafik işaret ve ışıklarının daha iyi tespit edilebilmesi için, YOLO modelinin nesne tespiti için kullandığı çapa boyutlarını ayarlamak için simülasyon veriseti üzerinde yapılan bir takım veri çalışmalarından yararlanıldı.

YOLO modelinin oluşturulması, kullanılması ve geliştirilmesi aşamaları, detaylarıyla birlikte otonom sürüş algoritmaları bölümünde anlatıldığından burada detaya girilmeyecektir.





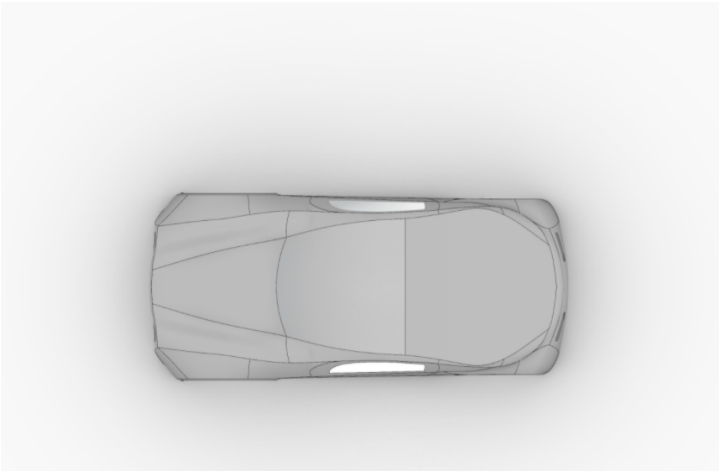
7.4. Veri Seti Oluşturulması

Yarışma koşulları ve görevleri göz önüne alındığında, derin öğrenme modellerinin eğitiminin özel olarak hazırlanmış veri setlerinden yararlanarak hazırlanmasının, yarışma görev başarımını artıracağı düşünüldü. Bu amaçla araçta kullanılacak derin öğrenme modellerinin eğitiminde kullanmak üzere simülasyon ve gerçek dünya veri setleri , takım üyeleri tarafından oluşturulmuştur. Aynı zamanda bu veri setlerinin işaretleyebilmek için, görüntü işleme ekibimiz tarafından yarı otomatik bir işaretleme aracı tasarlanmıştır.

7.5. Simülasyon Parametrelerinin Gerçek Araç Verilerinden Alınması

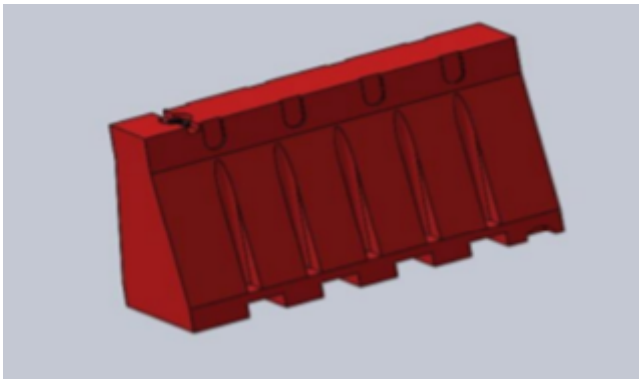
Araç simülasyon ortamımız, otonom sürüş algoritmalarımızın geliştirilmesi ve test edilmesi için kritik öneme sahiptir. Buna göre simülasyon ortamımızın ve aracımızın gerçek dünya koşullarına olabildiğince benzemesi, algoritmalarımızın geliştirilme sürecini oldukça kısaltmakta büyük rol oynamıştır.

Simülasyon ve gerçek dünya ortamının olabildiğince benzeşmesi için, araç modellemesi sırasında, aracın gerçek dünya verileri (durağan ve dinamik sürtünme katsayıları, direksiyon maksimum minimum açısı, direksiyon dönme hızı, araç ivmelenmesi ve yavaşlaması vb.) ölçülerek araç modeline dahil edilmiştir.



7.6. Yarışmada Kullanılan Bariyerlerin Simülasyona Aktarılması

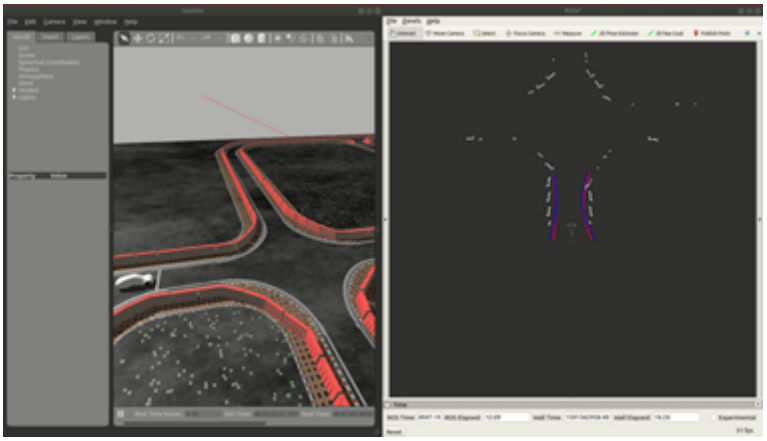
Bir önceki maddede bahsedildiği üzere, simülasyon ortamının gerçek dünya koşulları ile benzeşmesi, kritik öneme sahip bir konudur. Bunu sağlayabilmek amacıyla, 2019 Teknofest Robotaksi yarışmasında kullanılan bariyerler, ekibimizce bire bir modellenmiş ve simülasyon ortamına eklenmiştir.



7.7. Veri Kaybını Tolere Edecek Takip Algoritması

Sensörlerde yaşanabilecek anlık kopmaların etkisini azaltabilmek için, ana kontrol bilgisayarında çalışmakta olan bir takip algoritması geliştirilmiştir. Buna göre, sensör verilerinde (örneğin lidarda) oluşabilecek anlık anormal kopmalar (devam eden bariyerlerde kopma algılanması) tespit edilerek, takip algoritması ile bu verilerin yerinin doldurulması,

böylece hedef tespit ve yol planlama algoritmalarında istenmeyen sonuçların alınmasının önüne geçilmiştir.



8. Güvenlik Önlemleri

Araçta bulunan güvenlik önlemlerini mekanik, elektronik ve yazılımsal güvenlik önlemleri olarak üçe ayırmak mümkündür. Bu güvenlik önlemlerinin yürütme ve uygulanma önceliği sıralaması, şekilde ifade edilmiştir.

8.1 Mekanik Güvenlik Önlemleri

Araçta her 2 ön tekere de ayrı ayrı yerleştirilmiş hidrolik frenler, araca rejeneratif motor freni ile sağlanamayan güçlü ve ani fren yapabilme yetisini getirmektedir. Hem araç sürücüsünün hem de otonom kontrolcüsünün de erişiminde olan hidrolik fren sayesinde, güvenli ve kısa mesafeli fren yapılabilmektedir.

8.2 Elektronik Güvenlik Önlemleri

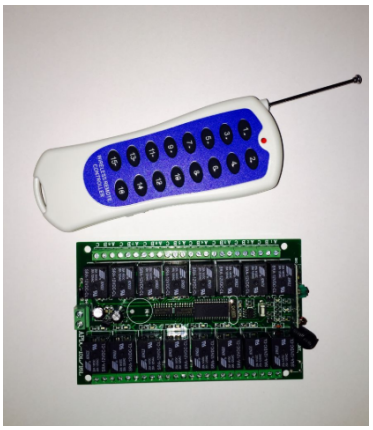
Elektronik güvenlik önlemleri olarak, araçta bulunan acil kapama sistemleri ve aracın her elektronik elemanına uygun büyüklükte bağlanan sigorta elemanı gösterilebilir.

Acil kapama sistemi, seri bağlanmış 3 elemandan oluşmaktadır. Buna göre, bu sistemlerden herhangi birinin acil kapama yapması sonucunda, bütün motor sistem kapanmakta, böylece araç durmakta ve batarya sisteminden kaynaklanabilecek olası sorunlar engellenebilmektedir.

Araç acil kapama sisteminin ilk elemanı acil durdurma butonudur. Varsayılan olarak acil durdurma devresini kapalı tutan bu sistem, butona basılması sonucunda bu devreyi açmakta, böylece motorlara giden güç kesilmektedir.

Acil kapama sisteminin 2. elemanı olan RF devresi, acil durdurma butonuna benzer şekilde varsayılan olarak kapalı vaziyette olup, kumanda ile açık konuma getirilmekte, böylece motorların gücü kesilebilmektedir.

Acil kapama sisteminin son elemanı ise, batarya yönetim sistemi dahilinde bulunan yüksek sıcaklık kesicisidir. Batarya sıcaklığının kritik seviyenin üstüne çıkması ile aktifleşen bu sistem, belirlenen eşiğin aşılması ile devreyi kesmekte, bataryadan güç çeken motorların kapanmasını sağlayarak batarya ve elektronik sistemi korumaktadır.



16 Kanallı RF kumanda ve Röle Seti

8.3 Yazılımsal Güvenlik Önlemleri

Yazılımsal güvenlik önlemleri, sensörlerden gelen verilerin işlenmesi sonucunda otonom kontrolcüsünün kararlarına göre devreye girmekte ve gerekli yönergeleri gerçekleştirmektedir.

Yazılımsal güvenlik önlemlerinin ilki, araç ana kontrol bilgisayarı dahilinde bulunan yazılımsal kalp atışıdır. Buna göre sensörlerden gelen veriler incelenmekte, son 0.5 saniye boyunca verinin gelmemesi ile kontrolcü motorlara dur komutu vermektedir. Böylece sensör iletişiminden kaynaklanan sorunlar sebebiyle oluşabilecek kazaların önüne geçilebilmektedir.

Yürütme önceliği prensibi, araç kontrolcüsünün gelen verileri işleme önceliğini belirtir. Buna göre, yürütme ve işleme önceliği: Fren – batarya – sıcaklık – direksiyon - gaz görüntüleme ekranı şeklindedir.

Araçta bulunan LIDAR sensöründen gelen uzaklık ölçümlerinde, belirli açılar arasında kritik eşiğin altında değerler okunması sonucunda kontrolcü, araca ani duruş sinyali göndermekte, buna göre gaz kesilmekte ve fren yapılmaktadır. Otonom araçlarda kritik bir öneme sahip olan bu davranış, yol güvenliği sağlanması açısından değerlendirilmiştir.

Yazılımsal güvenlik önlemlerinin son aşaması ise, araç uzaktan haberleşme sisteminde ve araç içi elektronik haberleşme sistemlerinin iletişim protokolleri sebebiyle dışardan erişilemez şekilde tasarlanmasıdır. Örneğin araçta bulunan RF devresi ve tetikleyici kumandasının iletişimi, uzaktan müdahaleye kapalı şekilde tasarlanmıştır.

9. Test

Testlerimiz Yazılım, Donanım, Araç Kontrol Testi ve Belirlenen Noktaya Gitme olmak üzere 3 farklı alandan oluşmaktadır.

9.1 Donanım Testleri

Donanım testlerini jüri ile görüşmelerimizi baz alarak gerçekleştirdik. Sistemimizde bellek ve yük analizi yaptık.

9.1.1. Grafik Kartı Testi

YOLO algoritmamız 8GB belleğe sahip grafik kartımızın yalnızca 2.6GB’lık bölümünü kullandı. Grafik kartımız, yaklaşık bir saat kullanım boyunca 2.6GB bellek kullanımını

aşmadı. Kullanım boyunca da ekran kapalı olacağından grafik kartımızın performansının oldukça yeterli olduğunu diyebiliriz.

9.1.2. İşlemci Testi

6 çekirdekli işlemcimizin, görüntüleme yazımımız ve sistemlerimizin hepsi birlikte çalışırken yaklaşık %430 doluluk oranında çalışarak 4 çekirdeğin tamamını doldurduğunu gördük. Görüntüleme yazılımını kapattıktan sonra ise işlemcimiz %210 doluluk oranında çalıştığını gözlemledik. Bu sonuçlar bilgisayarımızın işlemcisinin gereken işlemler için yeterli olduğunu göstermektedir.

9.2 Yazılım Testleri

9.2.1 ROS Node analizleri

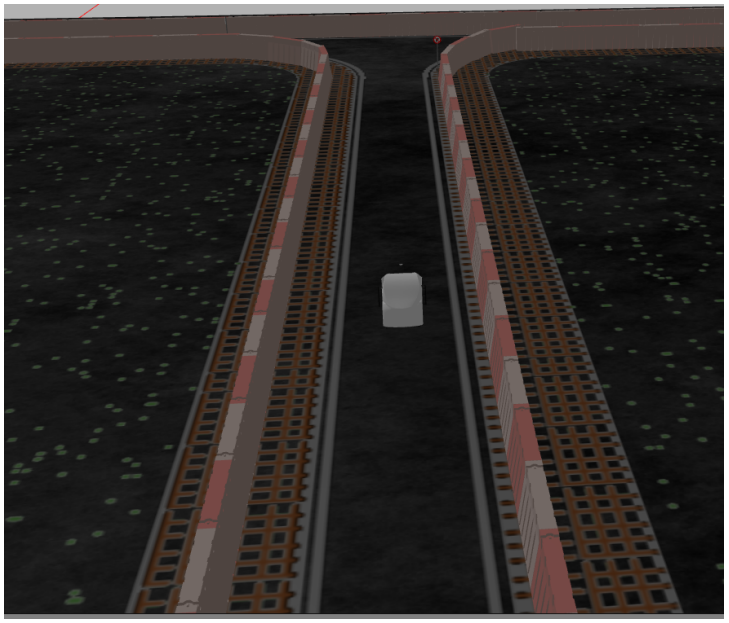
ROS Node denilen program parçalarının her biri için ‘Yazılım Profilleme’ denen analiz biçimini uygulayarak bu Node’ların 50ms olan maksimum tepki süresini aşıp aşmadıklarını inceledik. Sadece görüntüleme yazılımlarının tepki süresinin 50ms üzerine çıktığını gördük, ancak ekran yarışma esnasında kapalı olacağı için bu durum bir sorun teşkil etmeyecektir ve sistemimiz 20Hz’de kolayca çalışabilecektir.

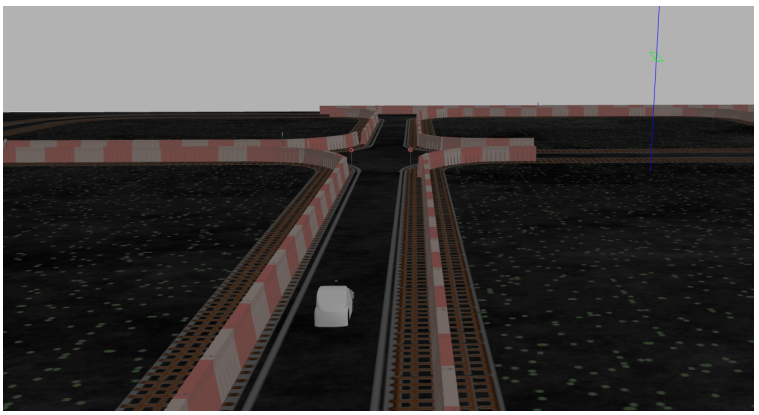
9.2.2 Simülasyon Parça Testleri

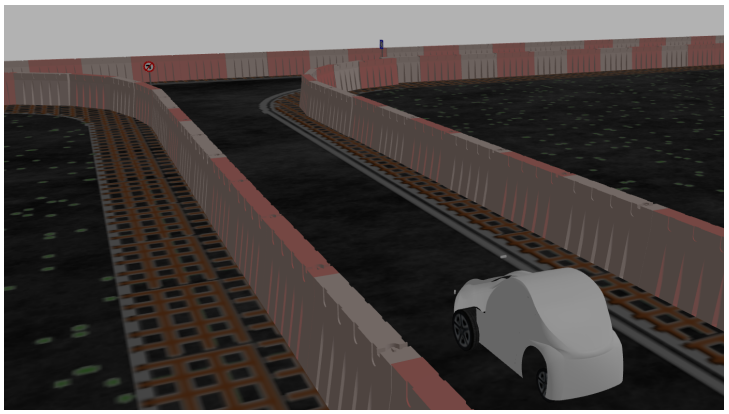
Aracımız gazebo simülasyon ortamında birçok farklı parça teste tabi tutuldu. Hem trafik işaretlerini hem de şeritleri tanımasını test eden bu testler genel olarak şu şekildedir:

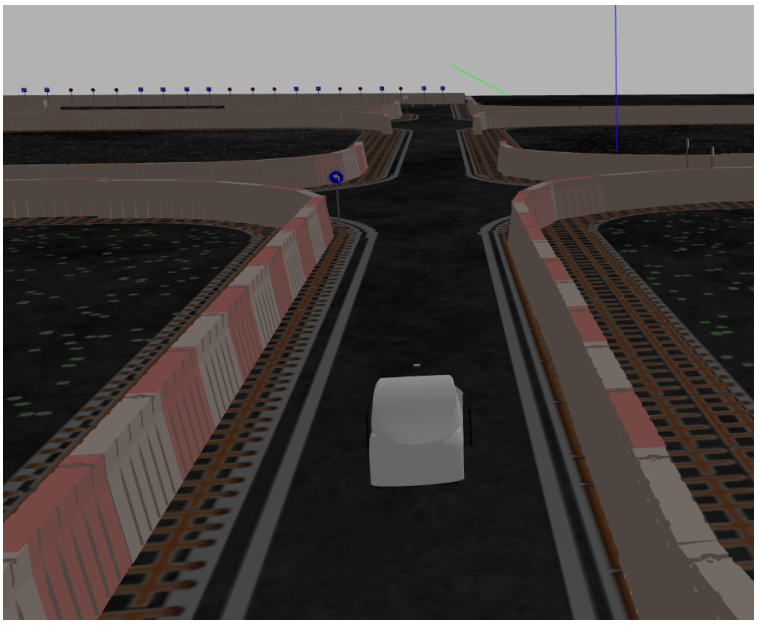
- 4’lü kavşakta sola dönüş tabelasıyla sola dönüş
- 4’lü kavşakta sağa dönüş tabelasıyla sağa dönüş
- 4’lü kavşakta sağa dönülemez ve sola dönülemez tabelalarıyla düz olarak ilerleme
- T şeklindeki kavşakta sola dönüş tabelasıyla sola dönüş
- T şeklindeki kavşakta sağa dönüş tabelasıyla sağa dönüş
- T şeklindeki kavşakta sağa dönülemez veya sola dönülemez tabelalarıyla düz olarak ilerleme
- Sağa doğru virajlı yollarda ilerleme
- Sola doğru virajlı yollarda ilerleme
- Düz yolda durakta durma
- 4’lü kavşakta sola dönüşün hemen ardından durakta durma
- 4’lü kavşakta sağa dönüşün hemen ardından durakta durma
- T şeklindeki kavşakta sola dönüşün hemen ardından durakta durma
- T şeklindeki kavşakta sağa dönüşün hemen ardından durakta durma
- 4’lü kavşakta sağa dönülmez ve araç giremez işaretleriyle sola dönüş
- 4’lü kavşakta sola dönülmez ve araç giremez işaretleriyle sağa dönüş
- T şeklindeki kavşakta sağa dönülmez işaretleriyle sola dönüş
- T şeklindeki kavşakta sola dönülmez işaretleriyle sağa dönüş
- Park Etme

Aşağıdaki resimlerde bazı örnekler verilmiştir.





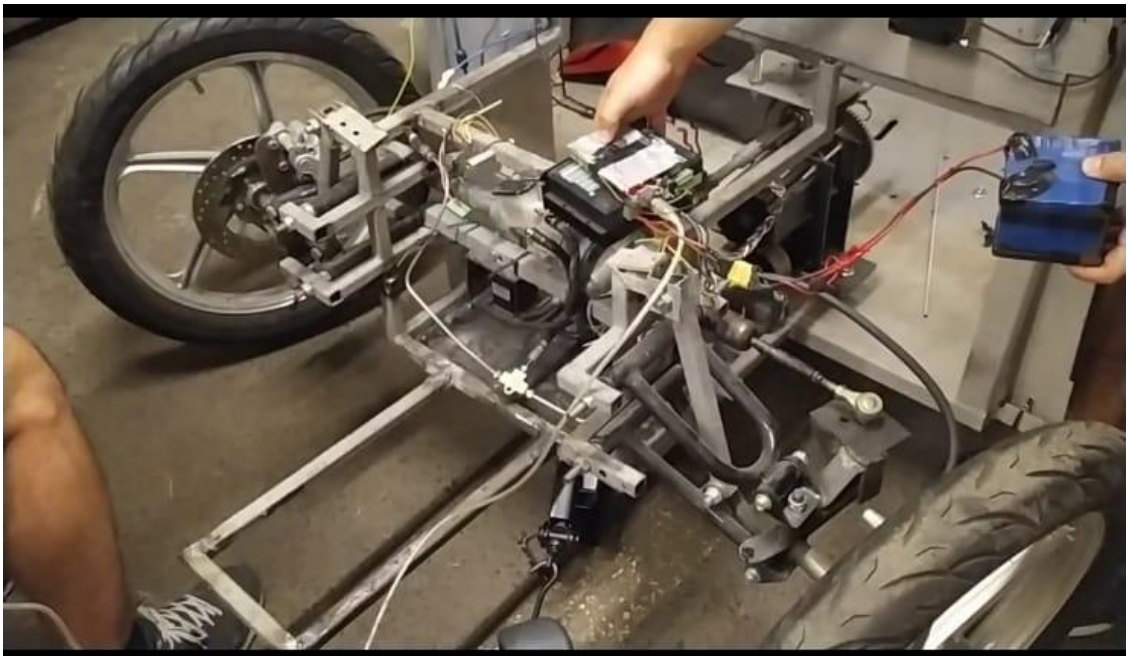




9.3 Araç Kontrol Hata Payı Analizi

9.3.1 Direksiyon ve Fren Hata Payı

Direksiyonumuzda yaptığımız analizde motorun dönmesi ve buna bağlı olarak hareketin ne zaman başladığını inceledik. Yaptığımız incelemelerde 800 adımdan oluşan motorumuzun ilk 3 adımının boşa döndüğünü gördük. Bu durumun olası etkilerini ve bu etkilerin bir sorun oluşturup oluşturmadığını uzun süren testler yaparak inceledik. Bu hatayı lineer potansiyometre kullanarak çözümledik. Aracı dilediğimiz an kolayca durdurabildiğimiz için frenlemedeki hata payının önemsiz olduğuna karar verdik.



9.4 Belirtilen Konuma Gitme Testi

9.4.1 Odometri Testi

Odometri testini uygulamak için aracımıza (0,0) koordinatlı konumdan (3,6) koordinatlı konuma gitmesi için komut verdik. Yaptığımız incelemeler ve hesaplamalar neticesinde odometri hata payını bin metrede 0.2 metre olarak hesapladık. Bu değerle birlikte odometri hata payımızın yarışma için sorun teşkil etmeyeceği sonucuna vardık.

10. Referanslar

[1] Ester, Martin; Kriegel, Hans-Peter; Sander, Jörg; Xu, Xiaowei (1996). Simoudis, Evangelos; Han, Jiawei; Fayyad, Usama M., eds. *A density-based algorithm for discovering clusters in large spatial databases with noise*. Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96). AAAI Press. pp. 226–231
[2] DBSCAN, Wikipedia, https://en.wikipedia.org/wiki/DBSCAN (Erişim tarihi: 12.04.2019)
[3] Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford (2001). "Section 24.3: Dijkstra's algorithm". *Introduction to Algorithms* (Second ed.). MIT Press and McGraw–Hill. pp. 595–601. ISBN 0-262-03293-7.
[4] DALAL, Navneet; TRIGGS, Bill. *Histograms of oriented gradients for human detection*. In: international Conference on computer vision & Pattern Recognition (CVPR'05). IEEE Computer Society, 2005. p. 886--893.
[5] CANNY, John. *A computational approach to edge detection*. In: Readings in computer vision. Morgan Kaufmann, 1987. p. 184-203.
[6] REDMON, Joseph; FARHADI, Ali. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.
[7] GIRSHICK, Ross, et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014. p. 580-587.
[8] GIRSHICK, Ross. Fast r-cnn. In: *Proceedings of the IEEE international conference on computer vision*. 2015. p. 1440-1448.
[9] REN, Shaoqing, et al. Faster r-cnn: Towards real-time object detection with region proposal networks. In: *Advances in neural information processing systems*. 2015. p. 91-99.
[10] REDMON, Joseph, et al. You only look once: Unified, real-time object detection. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016. p. 779-788.
[11] YILDIRIM, Metehan, DAL Ferhat Melih, MERAL Tuna Han Salih, TÜRKSOY Kadir, ZAMAN, Kerem, KÖSE, Ahmet Fatih, ÖZCAN, İbrahim, ÖNCÜ, Sinan. BURST: Software and Simulation Solutions of an Autonomous Vehicle. In: *2020 28th Signal Processing and Communications Applications Conference (SIU)*. 2020. *Accepted for oral presentation.*
[12] H. Abut, H. Erdogan, A. Ercil, et al. Data collection with "UYANIK": too much pain; but gains are coming. In: *Chapter 3 in Corpus and Signal Processing for Driver Behavior.* 2008.
[13] WANG, Ze; REN, Weiqiang; QIU, Qiang. Lanenet: Real-time lane detection networks for autonomous driving. *arXiv preprint arXiv:1807.01726*, 2018.
[14] Rosmann, Christoph, et al. "Online Motion Planning based on Nonlinear Model Predictive Control with Non-Euclidean Rotation Groups."