TEKNOFEST

HAVACILIK, UZAY VE TEKNOLOJİ FESTİVALİ

TARIMSAL İNSANSIZ KARA ARAÇLARI YARIŞMASI

KRİTİK TASARIM RAPORU ŞABLONU

TAKIM ADI

FAR-MO

PROJE ADI

FAR-MO

BAŞVURU ID

467270

İÇİNDEKİLER

İÇİNDEKİLER i
1. Takım Organizasyonu 1
1.1 Proje Ekipleri ve Yetenekleri 1
1.2 İş Akış Çizelgesi 1
2. Ön Tasarım Raporu Değerlendirmesi 2
2.1 Değişiklikler ve Sebepleri 2
2.2 Bütçe Planlaması ve Karşılaştırması 2
3. Araç Özellikleri 3
3.1 Genel Özellikler 3
3.2 Aracın Mekanik Özellikleri 3
3.2.1 Boyutlar ve Gövde 3
3.2.2 Hareket 4
3.2.3 Manevra 4
3.2.4 Tekerlekler 4
3.2.5 Şase 4
3.2.6 Namlu 5
3.3 Elektronik Özellikler 5
3.3.1 Sistemin Elektronik Tasarımı 5
3.3.2 Enerji ve Güç Sistemi 6
3.4 Yazılımsal Özellikler 7
3.5 Araç İçi ve Dışı Haberleşme 7
4.Sensörler 8
4.1 Ultrasonik Sensör 8
4.2 Kızılötesi Sensör 9
4.2 Kamera 9
4.4 Eğim Sensörü 9
4.5 GPS Modülü 10
4.6 İlaç Seviye Sensörü 10
4.7 Akım Sensörü 11
4.8 Sıcaklık Sensörü 11
5. Araç Kontrol Ünitesi 11

5.1 Servo Motor 12
5.2 Ultrasonik Mesafe Sensörü 12
5.3 GPS Modülü 13
5.4 DC Motor Sürme Kontrolü 13
5.5 LoRa ile Haberleşme Kontrolü 14
5.6 Far-Mo Araç Arayüzü 15
6. Otonom Sürüş Algoritmaları 16
6.1 Yazılım süreçleri 16
6.2 Otonom sürüş 17
6.3 Yabani Ot Tespiti Yazılımı 18
6.3.1 YOLO Nedir? 18
6.3.2 CNN Nedir? 18
6.3.4 Data Augmentation Nedir? 20
7. Yabani Ot Zirai Mücadele Yöntemleri 21
7.1 Fıskiye Sistemi 21
7.2 İlaç Deposu 21
8. Özgün Bileşenler 22
8.1 Elektronik Özgünlükler 22
8.1.1 Lora Haberleşme Modülü 22
8.1.2 Akım Sensörü 22
8.2 Yazılımsal Özellikler 22
8.3 Mekanik Olarak Özgün Bileşenler 22
9. Güvenlik Önlemleri 24
10. Simülasyon ve Test 25
10.1 Redüktörlü DC Motor testleri 25
10.2 Simulasyon Ortamı Gazebo 25
10.3 ROS (Robot Operating System) 25
10.4 Simülasyon Algoritması 26
10.5 Mukavemet Testleri 27
10.5.1 Şase Mukavemet Testi 27
10.5.2 DC Motor Tutucular Mukavemet Testi 28
10.6 Prototip Araç ile Yapılan Testler 28

10.6.1 Otonom Sürüş İçin Yapılan Testler 28
10.6.2 Yabani Ot Tanımak İçin Yapılan Testler 28
11. Kaynakça 29

1. Takım Organizasyonu

1.1 Proje Ekipleri ve Yetenekleri

Yazılım Ekibi: İKA'nın otonom, yarı otonom şekilde saha taraması yapmasını sağlayan ve yabani ot tespiti için çalışan ekibimizdir. Aynı zamanda radar sistemi kurulumu, algoritma oluşturulması, sensör yönetimi, motor kontrolü ve simülasyon ortamı kurulumu gibi bölümlerden sorumlu ekibimizdir.

Elektronik Ekibi: İKA'nın güç ve haberleşme yönetiminden sorumludur. Gerekli elektronik aksanların seçimi, kurulumu ve testlerini gerçekleştiren, mekanik ve yazılımsal sistemlerin bir arada çalışmasını sağlayan entegrasyon ekibimizdir.

Mekanik Ekibi: İKA'nın mekanik tasarımı, mekanik yapı malzemelerinin seçimi, araç mukavemeti ve yürüyüş sisteminden sorumlu ekibimizdir. Aynı zamanda fıskiye sisteminin kurulumu ve tasarımından sorumludur.

1.2 İş Akış Çizelgesi

TEKNOFEST TİKA YARIŞMASI	Tarih Aralığı	Şubat	Mart	Nisan	Mayıs	Haziran	Temmuz	Ağustos	Eylül
1. Başvuru Süreci	24 Şubat-07 Mart								
1.1 Takım Üyelerinin Belirlenmesi	24 Şubat-03 Mart								
1.2 Başvuru Belgelerinin Tamamlanması	24 Şubat - 07 Mart								
2. Ön Tasarım Raporu Süreci	3 Mart - 16 Mart								
2.1 Alt Ekiplere Ayrılma	7 Mart - 8 Mart								
2.2 Literatür Taraması ve Kavramsal Tarama	8 Mart - 15 Mart								
2.3 Malzemelerin Araştırılması ve Seçilmesi	10 Mart - 16 Mart								
3. Kritik Tasarım Raporu (KTR) Süreci	18 Mart - 27 Temmuz								
3.1 Sponsor Bulma Çalışmaları	19 Mart - 27 Temmuz								
3.2 Tasarım Geliştirme Süreci	28 Mart - 8 Haziran								
3.3 Malzemelerin Kesinleştirilmesi	20 Nisan - 15 Mayıs								
3.4 KTR Sürecinin Tamamlanması	18 mart - 6 Haziran								
4. Prototip İmalatı ve Sistem Entegrasyonu	01 Mayıs - 25 Haziran								
4.1 Malzemelerin Temin Edilmesi	01 Mayıs - 5 Temmuz								
4.2 Prototipin Mekanik Entegrasyonu	16 Mayıs - 25 Haziran								
4.3 Prototipin Elektronik ve Yazılım Entegrasyonu	16 Mayıs - 25 Haziran								
5. Test Aşamaları ve Eksiklerin Giderilmesi	08 Haziran- 31 Ağustos								
5.1 Hareket Kabiliyeti ve Videoların Sisteme Yüklenmesi	26 Haziran- 4 Temmuz								
5.2 Aracın Son Halinin İmalatının Tamamlanması	26 Haziran- 26 Temmuz								
Tarımsal İKA Proje Sürecinin Tamamlanması									

Şekil-1: İş Akış Çizelgesi

2. Ön Tasarım Raporu Değerlendirmesi

İKA üzerinde gerçekleştirdiğimiz mühendislik hesapları ve testler sonucunda, ön tasarım raporunda belirtmiş olduğumuz birtakım özellikleri değiştirdik. Yaptığımız değişiklikler ile birlikte aracı, artık daha işlevsel ve daha düşük maliyetle imal edilebilecek duruma getirdik ve şekil-2'de belirtildi.

2.1 Değişiklikler ve Sebepleri

Değişen	Değişme Sebebi	Sonuç
Araç Boyutları: 600x600x500	Aracın engel geçme kabiliyetini arttırmak ve toprak arazisinde daha sağlam yol tutuşu sağlamaktır.	Araç boyutları: 600x500x370
STM-32 Fiyatı: 799 TL	Yapılan hesaplamalara göre pin sayısı ve kullanış kolaylığı gibi sebeplerden dolayı değişikliğe gidilmiştir.	1 adet Arduino Mega Fiyatı : 444 TL 1 adet Arduino Uno Fiyatı : 243 TL kullanılmasına karar verilmiştir.
LiDaR Fiyatı: 2.313,91 TL	Kullanış rahatlığı ve daha sağlıklı bir radar sistemi kurmak için kaldırılmıştır.	Mesafe sensör sayısı arttırarak yeni bir radar sistemi kuruldu.
HC-SR04 2 Adet Fiyatı : 70,90 TL	Yeni radar sisteminde daha hassas ve sağlıklı sonuçlar elde etmek için yetkinlik bakımından işimize daha uygun mesafe sensörleri kullanılacaktır.	6 adet US-100 Ultrosonik Mesafe sensörü Fiyatı : 732 TL 2 adet Mz-80 Kızılötesi Mesafe Sensörü Fiyatı : 115,30 TL
Araç Maksimum Hızı: 15cm/s	İKA'nın yarışmayı zamanında tamamlaması için aracın maksimum hızını arttırma gereksinimi duyduk.	Araç Maksimum Hızı: 25 cm/s
Değişen Parça Maliyeti : 3.182,81 TL		Belirlenen Yeni Parçaların Maliyeti : 1.534,3 TL

Şekil-2: Değişiklikler ve Sebepleri

2.2 Bütçe Planlaması ve Karşılaştırması

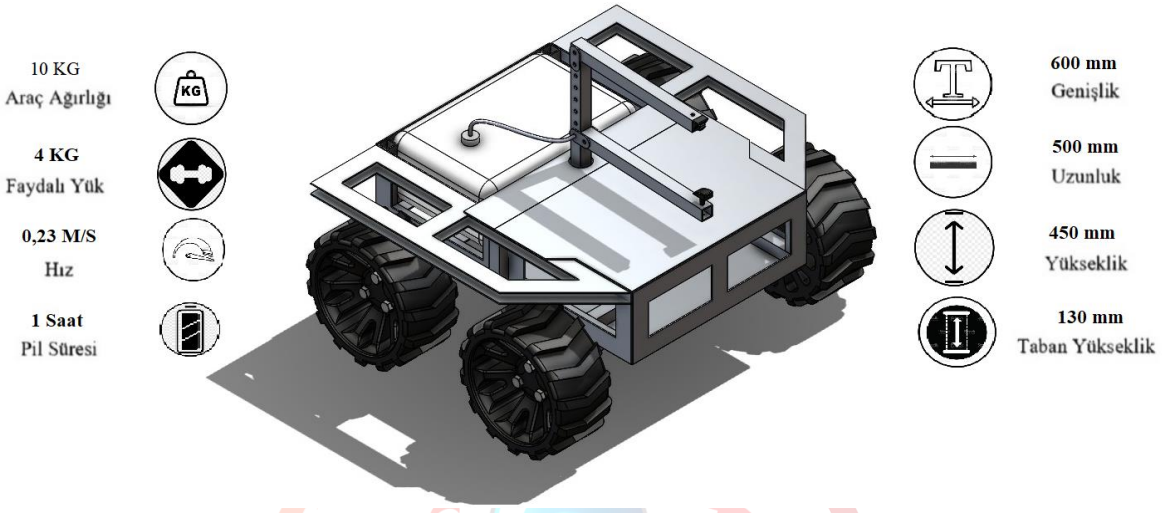
KTR AŞAMASI BÜTÇE PLANLAMASI							
Malzeme	Adet	Birim Fiyatı	Toplam Fiyatı	Malzeme	Adet	Birim Fiyatı	Toplam Fiyatı
Raspi Kamera	1	₺ 686,69	₺ 686,69	Voltaj Regülatörü	2	₺ 5,00	₺ 10,00
Raspbeary Pi	1	₺ 4.491,38	₺ 4.491,38	Encoder	4	₺ 72,00	₺ 288,00
Arduino Mega	1	₺ 444,00	₺ 444,00	Redüktörlü DC Motor	4	₺ 159,00	₺ 636,00
Arduino Nano	1	₺ 224,54	₺ 224,54	Motor Sürücüsü	4	₺ 119,59	₺ 478,36
Arduino Uno	1	₺ 243,00	₺ 243,00	MG995 Servo Motor	1	₺ 79,00	₺ 79,00
us-100 Mesafe Sensörü	6	₺ 122,00	₺ 732,00	Opel Meriva Su Fıskiye Motoru	1	₺ 83,63	₺ 83,63
mz-80 Kızıl Ötesi Sensör	2	₺ 57,65	₺ 115,30	İlaç Deposu	1	₺ 41,25	₺ 41,25
ADXL345 Eğim Sensörü	1	₺ 54,25	₺ 54,25	Araç silecek fiskiyesi	1	₺ 76,00	₺ 76,00
NEO-GM GPS	1	₺ 91,00	₺ 91,00	LORA Anten	2	₺ 112,00	₺ 224,00
AR-064 Su Seviye Sensörü	1	₺ 11,00	₺ 11,00	LORA Modül	2	₺ 298,00	₺ 596,00
ACS-712 Akım Sensörü	1	₺ 61,00	₺ 61,00	LORA USB Stıck	1	₺ 174,00	₺ 174,00
DHT-11 Sıcaklık Sensörü	1	₺ 29,06	₺ 29,06	Sigma Profil	295cm	₺ 110,33	₺ 330,99
12V 12Ah Kuru Akü	1	₺ 721,03	₺ 721,03	Alüminyum Levha	2 m kare	₺ 1.368,56	₺ 1.368,56

KTR Maliyeti	₺12.290,04	ÖTR Maliyeti	₺13.938,55	KAR	₺1.648,51

Şekil-3: Bütçe Planlaması ve Karşılaştırması

3. Araç Özellikleri

3.1 Genel Özellikler



Şekil-4: Genel Özellikler

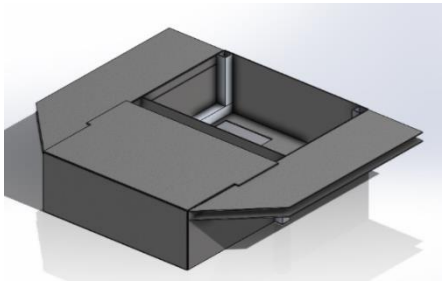
(2021, Teknofest, Tarımsal İnsansız Kara Aracı Yarışması, Space and Beyond Technology)

Far-Mo takımı olarak amacımız hayatın her yerine kolayca entegre olabilen bir İKA ortaya çıkarmaktır ; bu sebepten dolayı aracımızın tasarımını yere daha sağlam basabilen , hem toprak arazide hem de kaldırımda kolaylıkla yol alabilecek ve modüler kabiliyetlere sahip bir şekilde imal ediyoruz .Bu aracın en büyük özelliği ; aracın merkezinde bulunan depo ve fıskiyenin kolaylıkla araçtan çıkarılıyor ve aynı kolaylıkla takılabiliyor oluşudur ilerleyen zamanlarda aracımız için bir robot kol , bir yük taşıma aparatı ve çapa atacak bir mekanizma tasarlamayı planlıyoruz çünkü biliyoruz ki çiftçinin tek sorunu yabani otlar değil tarlada ki meyvenin toplanması da sorun , tarlaya çapa atması da sorun biz aracımızı modüler bir araca çevirerek tek araç 4 aparat ile çiftçinin üzerinde ki yükü büyük ölçüde azaltmayı hedefliyoruz . Araç özelliklerini anlattığımız bu bölümde aracın bu özellikleri nasıl sağlayacağını detaylıca anlattık.

3.2 Aracın Mekanik Özellikleri

3.2.1 Boyutlar ve Gövde

Aracın azami uzunluğu ve genişliği 500x600 mm olup gövdesi yani şasesinin etrafı ise ince alüminyum plakalarla kaplanacaktır. Amacımıza hizmet edecek gerek maliyet gerek ağırlık olarak en verimli imalat yönteminin bu olacağına karar verdik. Çünkü alüminyum plakaların kesilmesi ve delinmesi nispeten daha kolay, kırılgan olmayan malzemelerdir. Böylelikle eğer bir yüzeyin tamamının kapalı olmasına gerek yoksa yüzeyde delikler açarak ağırlığı düşürmek ve bağlantı delikleri delmek de daha kolay olacaktır.



Şekil-5: Araç Gövdesi

3.2.2 Hareket

İKA'nın, arazi koşullarında bulunan çukur ve tümsek gibi engebeleri rahat geçebilmesi adına; İKA'da bulunan 4 tekerleğe 4 ayrı 12 V ile çalışan 37 RPM hızında 8 kg*cm yükleme torku, 2.2 kg*m zorlama torkuna sahip DC motor bağlayarak aracımıza gereken hareket kabiliyeti kazandırılmıştır.

3.2.3 Manevra

Araca manevra kabiliyeti kazandırmak için kızaklama sistemiyle, aracı sağa döndürmek için yalnızca araçta bulunan sol motorlara güç verip eğer sola döndürmek istiyorsak da yalnızca sağdaki motorlara güç vererek araca manevra kabiliyeti kazandırılmıştır. Araç arazi koşullarında çalışacağı ve genişliği, uzunluğundan fazla olduğu için bu yöntemin en verimli yöntem olduğuna karar verdik.

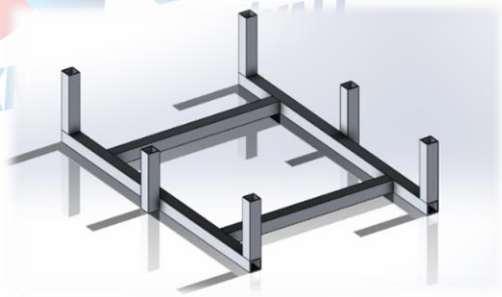
3.2.4 Tekerlekler

Aracın, arazi koşullarında uyum sağlaması ve toprak tepeciklerini rahatça geçebilmesini sağlamak amacıyla çaplarının büyüklüğü 200 mm genişliğinde olan hazır tekerlekler kullanılacaktır. Kendi imalatımızdan daha az maliyetli olacağı için hazır almayı seçtik. Bu tekerlekler, araca yardımcı bir plaka kullanarak bağlanacaktır. Bu bağlantının ayrıntılı bilgisi ve şeması, Bölüm-8 Özgün Bileşenler'de mevcuttur.

Şekil 6: Kullanılacak Hazır Teker

3.2.5 Şase

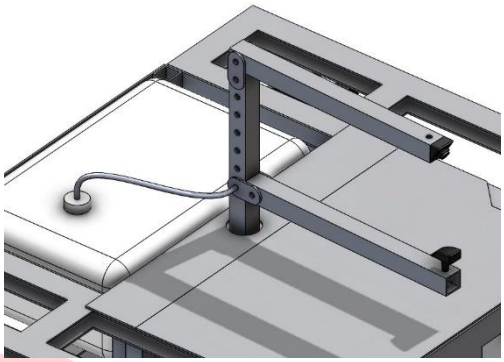
Aracın, kuvvetlere karşı şeklini koruması için alüminyum sigma profillerden oluşan bir şase imal edilecektir. Bütün ağır kuvvetler (depo, motorlar vb.) bu şase üzerinde olacak ve iskelet görevi görecektir. Sigma profil seçmemizin nedeni yine maliyet ve işlenebilirliğe karşı kuvvet dayanımının istediğimiz derecede olmasıdır.



Şekil 7: Araç Şasesi

3.2.6 Namlu

Yabancıleri tespit edebilmesi ve ilaç sıkabilmesi adına, aracın tam ortasında tıpkı tanklardaki gibi ama 180 derece dönebilen bir başlık bulunuyor. Bu başlık bir adet servo motorla döndürülecek olup üzerinde kamera ve bir adet fıskiye bulunuyor. Kameranın daha iyi görüntü işleme ve tespit yapılabilmesi için biraz daha yukarıya, fıskiyenin ise bitkiye daha yakın olması için daha aşağıya konumlandırıldı. Bu kısımla ilgili ayrıntılı bilgiyi Bölüm-7 Yabani Otla Zirai Mücadele Yöntemleri'nde bulabilirsiniz.



Şekil-8: Araç Başlığı

3.3 Elektronik Özellikler

3.3.1 Sistemin Elektronik Tasarımı

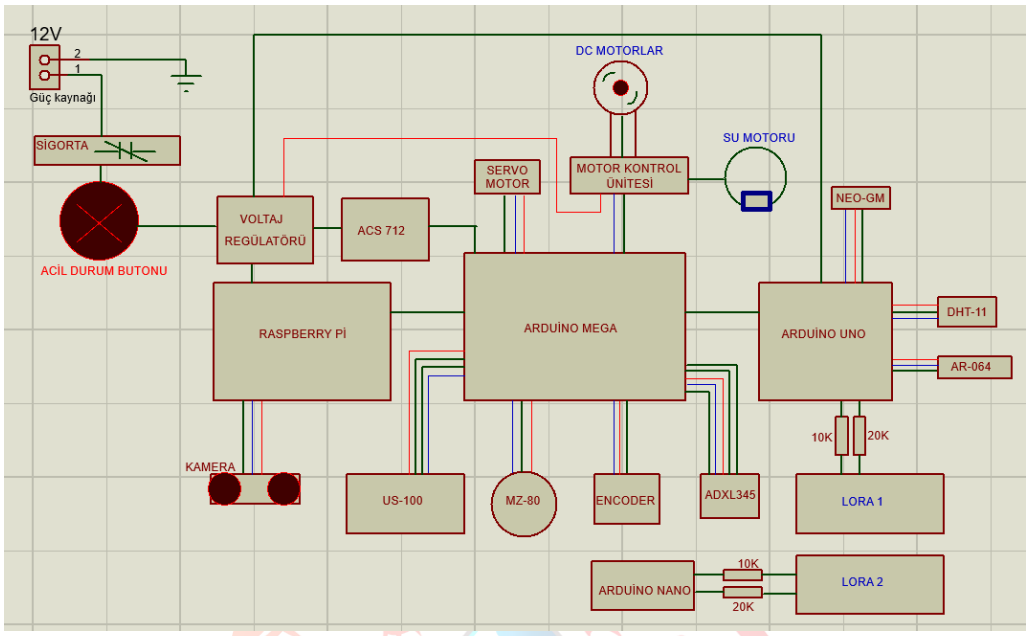
İKA'nın, yarışma alanında gerekli görevleri yerine getirmesi için elektronik aksanların en doğru şekilde seçilmesi ve kullanılması gerekmektedir. Otonom saha taraması için radar sistemimiz, eğim sensörümüz, enkoder ve GPS aktif rol alacaktır.

Arduino Mega, ana kontrolcü olarak radar sistemi ve otonom sürüş için gerekli tüm elektronik parçaların kontrol ve denetlenmesinden sorumludur. Araç engel tespit ettiği durumda duracaktır ve Raspberry Pi, görüntü işleme bilgisayarına etrafımda yabani ot var mı sorusunu soracaktır. Eğer ortamda yabani ot var ise Raspberry Pi işlemcisinden gelen veriye bağlı olarak ana kontrolcümüz servo motorumuzu ve su motorunu aktif hale getirecektir.

Aracın ilaçlama yapabilmesi için arka kısmına sızdırmaz depo yerleştirdik. Aynı zamanda ağırlık dengesini bir nebze dengelemeyi düşünüyoruz. Bu deponun içine ilacı, püskürtme borusuna iletiminde görev alacak 12V DC fırçasız su pompası kullanacağız. İlaç, bu sayede kültür bitkilerine zarar vermeden yabani otu etkisizleştirecektir. Ayrıca depodaki ilaç miktarını ölçmek için AR-064 su seviyesi sensörü kullanacağız. Bu sayede depodaki ilaç seviyesi de kontrolümüz altında olacaktır. Su seviye sensörünün analog değerini, aynı zamanda arayüz programımızın üzerinden takip edeceğiz.

Aracın en kötü ihtimalle yaklaşık 60 dakika çalışacağını varsayarsak bu durumda oluşacak olası işlemci ve pil ısı artışına karşı emniyetli olacağız. Bu doğrultuda kullanacağımız sisteme yakın olacak şekilde DHT-11 sıcaklık sensörü kullanacağız. Bu sıcaklık sensörünü tercih etmemizin sebebi 0-50 derece arasını 2 derece hassasiyetle ölçmesidir. Araçtaki her türlü güvenlik önlemini arayüzden takip edebileceğimiz gibi pil, işlemci sıcaklığını da bilgisayar ortamından takip edebileceğiz.

Aracın otonom saha taramasına yardımcı olmayacak tüm elektronik aksanları, ana kontrolcümüzü çok fazla yormamak için kullanacağımız 3. kart olan Arduino UNOUn'ya vereceğiz. Aynı şekilde Arduino UNO kartına bağlı, LoRa haberleşme sistemi ile arayüze gelecek sensör verilerinin denetimini ve araç kontrolünü manuel olarak sağlayacağız.



Şekil-9: Elektronik Aksan Şeması

3.3.2 Enerji ve Güç Sistemi

Batarya enerjisinin araç sistemine güvenli bir biçimde aktarılabilmesi için tesis edilmiş kablonun, güç aktarımı için gerekli olarak öngörülen maksimum 30A akımı üzerinden güvenli bir şekilde geçirilebilmesi için ACS-712 akım sensörü kullanıyoruz. Bu akım sensörü -30A +30A kadar destekliyor. Ayrıca batarya enerjisinin aracımızın mekanik ve elektronik bileşenlerine aktarımı, güç dağıtım kartı ile gerçekleştirilecektir. Güç dağıtım kartı, bataryanın gücünü 4 adet motor sürücü kartımıza herhangi bir gerilim dönüşümü yapmadan aktaracaktır.

Araç kontrol ünitemizde tasarladığımız devre robotun motorlarına enerji sağlamak için senkronize edilmiş ve düzenlenmiş güç çıkışları da sunmaktadır. Motor sürücü kartlarımız ise bağlı olduğu motorun elektrik direnci ayarının, hız ve tork için gerekli ayarların verisini Ardunio üzerinden kontrol edeceğiz. Elektronik elemanların kullanılabilmesi için gerekli voltaj regülatörlerin kullanılması gerekmektedir.

Diğer tarafta da sensörlerin, servo motorların ve kameranın çalışma voltajı 5V'tur. Takımca yapılan toplantılar sonrasında Li-Po pil yerine aracın tüm enerji ihtiyacını hesapladıktan sonra 12V 60 Watt kuru akü kullanılması daha uygun görülmüştür. Aracın yaklaşık toplam dakika hesabı aşağıda verilmiştir. Buradaki 0.7 çarpanı akü ömrünün parametrelerini temsil ediyor. Bu değerleri akü tam şarj durumundayken hesapladığımız için yarışma alanında akü pil durumunu kontrol edeceğiz. Aynı zamanda diğer kontrolcülerimiz için 12V- 5V gerilim düşürücü regülatör kullanılmıştır ve XT60 bağlantı konektörü bulunur. Konektörler akü-kontrol kartı arasındaki güç aktarımı yapacağından bağlantıyı diğer bağlantılara göre daha sağlam tutacağız ve ana kontrol devremizde bu pil konektörüne göre erkek-dişi piller yerleştireceğiz.
Aracın hesaplanan çalışma süresi = $(12\ Ah * .0.7)\ /\ 5A$ = 1,68 saat = 205 dakika .Aracımızı normal koşullarda 205 dk çalıştırabilirz fakat işlemcilerin bu kadar uzun süre çalışmasının çok sağlıklı sonuçlar vermeyeceğini düşünüyoruz bu sebepten araç tek seferde 60 dk çalışacak ; aracımızın aküsünün bu kadar güçlü olma sebebi tarla ortamında aracın her fırsatta şarj edilememesine karşın bir önlemdir .

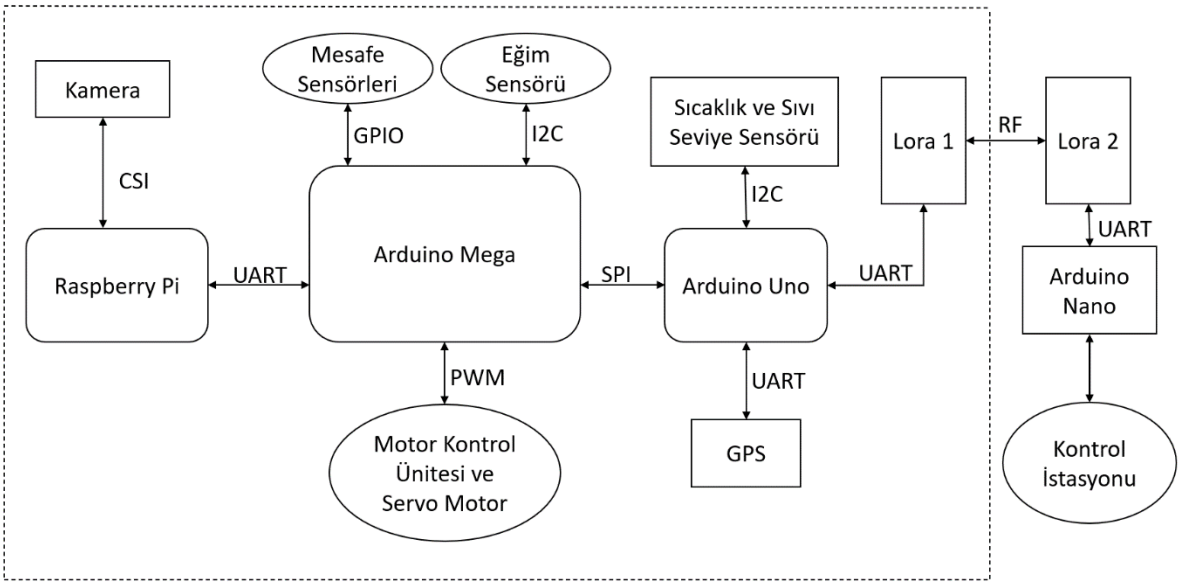
3.4 Yazılımsal Özellikler

İKA üzerinde ve kontrol ünitesinde olmak üzere birçok farklı alanda yazılımsal özellikler barındırmaktadır. Farm-O'nun üzerinde bulunan Araç Kontrol Ünitesi; otonom sürüşü sağlama, yabani ot ve engel algılama, kontrol merkezi ile iletişim kurma özelliklerine sahiptir. Web ve masaüstü destekli Kontrol Merkezi Uygulaması İKA ile iletişimde olup manuel kontrol, İKA'dan gelen konum, ivme ve batarya durumu gibi araç bilgilerini anlık olarak kullanıcıya yansıtma özelliklerine sahiptir.

Yabani ot imhası için tasarlanan algoritmanın en önemli aşaması yabani otun doğru bir şekilde algılanmasıdır. Bu doğrultuda kullanılacak CNN algoritması olarak YOLO (You Only Look Once) seçilmiştir. YOLO hakkında detaylı bilgiye Otonom Sürüş Algoritmaları bölümünde özel olarak yer verilmiştir. Görüntü işleme için Open-CV kütüphanesini Python programlama dili ile kullanımı tercih edilmiştir. Araç kontrolünü tek elden sağlamak ve veri yönetimini sağlayabilmek için testlerde Python'ın pyMata modülüne yer verilmiştir. Ardından Raspberry Pi 4 ile Arduino Mega arasında haberleşmeyi sağlamak için pySerial modülü kullanımı tercih edilmiştir.

3.5 Araç İçi ve Dışı Haberleşme

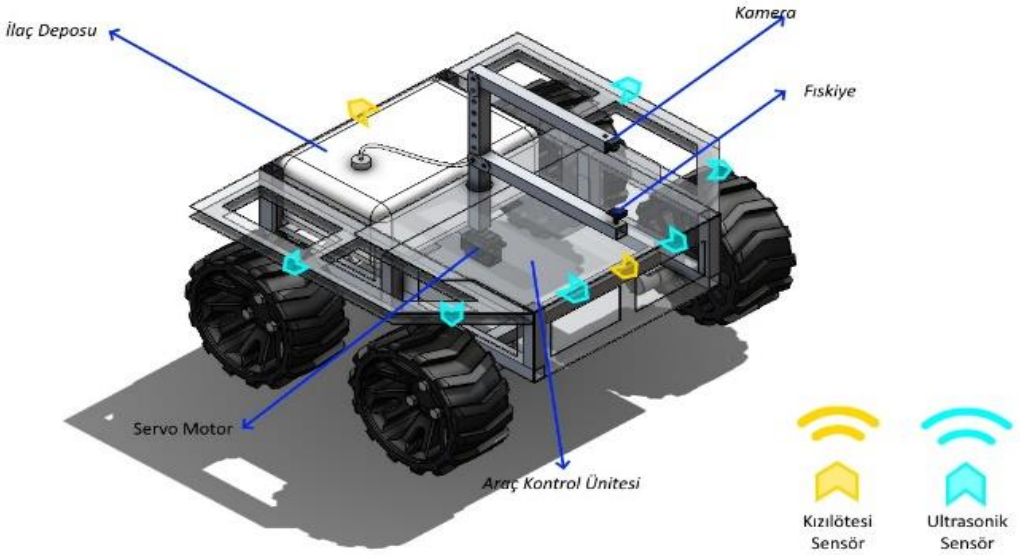
Araç yarışmada bizden beklenen ana 3 görevi yerine getirebilmek için basit , rahat ve sağlıklı bir haberleşme şemasına sahiptir ; araç içinde çokça kullanılan haberleşme protokollerini kullanarak özellikle otonom sürüş aşamasında hiçbir problem ile karşılaşmayacağımızı öngörüyoruz , araç dışı haberleşme kısmında hem sensör verilerinin analizi hem de manuel araç kontrolü için uzun menzile ulaşma kabiliyetine sahip LoRa teknolojisinin tarım arazisinde çok iyi işler çıkaracağından eminiz .



Şekil-10: Haberleşme Şeması

4.Sensörler

Araç otonom hareket edebilme kabiliyetini; üzerinde bulunan 6 adet ultrasonik mesafe sensörü, 2 adet kızılötesi sensör, 1 adet akım sensörü, 1 adet kamera, 1 adet GPS modülü, 1 adet eğim sensörü, 1 adet ilaç seviyesi sensörü, 1 adet sıcaklık sensörü sayesinde otonom hareket edebilecektir. Aracın genel olarak sensör konumları aşağıda gösterilmiştir. Eğim, sıcaklık, akım, GPS sensörleri araç kontrol ünitesinde yer alacaktır.



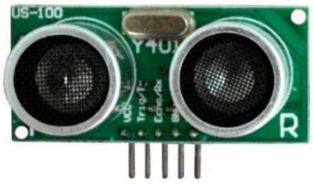
Şekil-11: Sensör Konumları

4.1 Ultrasonik Sensör

Mesafe tespitinde kullanacağımız US-100 sensörü, 2 cm ile 450 cm aralığında çalışan ultrasonik bir sensördür. Genelde tüm mesafe sensörü gibi kullanılan bu sensör de ses dalgası ile uzaklık hesaplama prensibiyle çalışıyor. Ses kanallarından birisi dalga gönderirken diğeri dalganın geri gelme süresinden uzaklık hesabı yapıyor. Aracın ön kısmında 2 adet, sağ-sol çaprazda 2 adet ve sağ-sol yanda 2 adet olacak şekilde toplamda 6 adet US-100 mesafe sensörü kullanacağız. HC-SR04 sensörle çok benzerlik gösterse de hassasiyet kalitesi ve daha sağlıklı çalışmasından dolayı bu sensörü tercih ettik.

Özellikleri:

- Çalışma Voltajı: DC 5 V
- Çektiği Akım: 15 mAh
- Çalışma Frekansı: 40 Hz
- Görme Açısı: 15 ^{0}
- Maksimum Görme Menzili: 4 m
- Minimum Görme Menzili: 2 cm



Şekil-12: Ultrasonik Sensör

4.2 Kızılötesi Sensör

Aracın hem emniyetini hem de otonom sürüşünü kolaylaştırmak adına 80 cm menzile sahip aracın önünde 1 adet ve arkasında 1 adet olacak şekilde 2 adet MZ-80 kızılötesi mesafe sensörü kullanacağız. Böylece US-100 ve MZ-80 mesafe sensörleri ile LiDaR kullanmadan gerekli radar sistemini daha az maliyet daha kolay kontrol mekanizması ile kuracağız.

Özellikleri:

- Dijital çıkışlı, yüksek kaliteli endüstriyel sensördür.
- Arkasındaki pot ile menzili 3-80 cm arasında ayarlanabilir.
- NPN çıkışlıdır.
- 5V ile çalışır, tepki süresi oldukça düşüktür. (2 ms)
- Montajlama aparatıyla rahatça montajlanabilir.

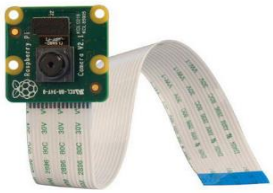
Şekil-13: Kızılötesi Sensör

4.2 Kamera

Araçta, fıskiye sistemine entegre şekilde 180 derecelik dönme ve çapraz açı 130 derece görme kapasitesine sahip bir adet kamera kullanacağız. Aracımızın radar sistemi, bir engel tespit ettiği durumda kameramızın etrafında yabani ot olup olmadığını görüntü işleme yolu ile denetleyecek eğer yabani ot var ise yabani otun bulunduğu noktaya ilaçlama yapılacak.

Özellikleri:

- Yüksek kaliteli görüntü algılama
- Büyük veri işleme kapasitesi
- 8 mega piksel sabit odak noktalı
- 1080p, 720p60 ve VGA90 destekli
- Sony IMX219PQ CMOS görüntü algılayıcı
- 15-pin şerit kablo



Şekil-14: Kamera

4.4 Eğim Sensörü

ADXL345 sensörü, aracın tarım arazisi içerisindeki mutlak pozisyonunun saptanmasında kullanılmaktadır. Bu sensör, 3 boyutlu eksende ivme değerlerini, yüksek hassasiyette analog olarak çıkış veriyor. Parkurda önemli görevlerden biri olan bitki tahrip etmesinden başka zorluklara karşı çıkmak için bu sensörü kullandık. İKA'nın toprak arazide ilerlediği yolun eğimine yönelik veriler oluşturacaktır. Bu veriler doğrultusunda motorlara verilecek güç değerlerinin kontrolü, kontrol kartı tarafından yazılımsal PWM ayarı yapılarak aracın çukurlardan, tepelerden çok rahat bir şekilde geçmesini sağlayacağız.

Özellikleri:

- Giriş Voltajı: 3-5V
- Düşük Güç Tüketimi (40uA)
- Serbest Düşüş Algılama
- SPI ve I²C Arayüzü

Şekil-15: Eğim Sensörü

4.5 GPS Modülü

Aracın yarışma alanı içerisinde konum kontrolü ve takibini GPS modülü sayesinde alacağız. Bu bağlamda araç üstünde bulunan GPS, yarışma alanı içerisinde pist alanına çıkmasını engelleyecektir. Bu veri bilgisini arayüz yazılımımız üzerinden de takip etmeyi planlıyoruz. Yarışmada pist genel GPS kullanımına göre küçük olduğundan araçta kullanacağımız GPS modülünün hassas çalışması gereklidir. Bu sebepten dolayı 5 metrelik hassasiyete sahip GY-NEO6MV2 GPS modülü tercih ettik.

Şekil-16: GPS Modülü

4.6 İlaç Seviye Sensörü

Araçta bitkilerin üzerine fışkırtacak olan ilacı, ilaç deposunda muhafaza edeceğiz. Doğal olarak ilacın yeterlilik seviyesini kontrol altında tutmak zorundayız. Kullanacağımız sensörün çalışma prensibi birbirine paralel olarak çekilmiş iletkenlerin sıvı ile teması sonucu çıkışına analog bir değer veriyor. Bu analog değer belli bir değer altına düşerse ilaç miktarını arayüze aktaracağız. Dijital çıkış veren seviye sensörü kullanmamızın sebebi hassasiyetinin çok düşük olmasıdır. Seviye sensörünü ilaç deposunun iç kısmına koyacağız fakat elektronik kısmın sıvı ile temasını bir koruma ile engelleyeceğiz.

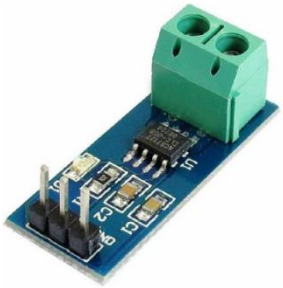
Özellikleri:

- Çalışma Voltajı: 5V
- Çalışma Akımı: <20mA
- Ürün Boyutları: 65x20x8 mm
- Sensör Okuma Alanı: 40x16 mm

Şekil-17 Su Seviye Sensörü

4.7 Akım Sensörü

Araçta kullanacağımız 2 güvenlik sensörlerinden biri manyetik etkiye bağlı lineer akım sensörüdür. Araç kontrol kartının 30 ampere kadar iki yönlü akım girişine izin verecek şekilde bu sensörü seçtik. Aynı zamanda hata payı %5'nin altında olmak üzere sağlıklı bir analog çıktısı veriyor. Sensörün çalışma mantığı akım hattını sensörün elektroniğinden elektriksel olarak izole edebilmesidir. (2.1 kV RMS'e kadar) Bu sensör aynı zamanda devredeki diğer tüm elektronik bileşenleri olası bir hata durumunda koruyacaktır.

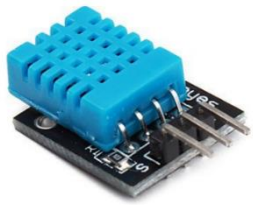


Şekil-18: Akım Sensörü

Yalnızca sigorta kullanarak tüm gücün kesilmesini sağlayabilirdik. Fakat İKA, toprak arazide çalışacak ve motorlarımız dur kalk yaparken fazla akım çekebilir ki bu da normaldir. Sigortanın bu durumlarda atmasından korktuğumuz için hem aracın emniyetini daha sağlıklı seviyeye çıkaracağız hem de aracın hiçbir sıkıntı yokken ilerleyişini durdurmasını engelleyeceğiz.

4.8 Sıcaklık Sensörü

Aracın bir diğer güvenlik sensörü ise DHT11 sıcaklık ve nem algılayıcı sensördür. Sensörün başlıca özellikleri yüksek güvenilirliktedir ve uzun dönem çalışmalarda dengelidir. 8 bit mikroişlemci içerir, hızlı ve kaliteli tepki verir. 0 ile 50°C arasında 2°C hata payı ile sıcaklık ölçer. Yarışma alanında aracın yaklaşık 30-40 dakika çalışacağını varsayarsak kullanacağımız kart ve güç ünitesinin sıcaklığını görmek isteriz. Eğer ki kritik sıcaklık noktasını geçtiğimizi arayüz üzerinden tespit edersek aracı güvenli bir şekilde durduracağız.



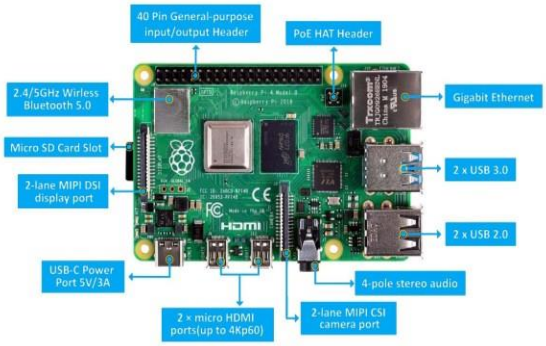
Şekil-19: Sıcaklık Sensörü

5. Araç Kontrol Ünitesi

Araç üzerinde otonom sürüş ve araç kontrolü için kullanılan donanımlar Raspberry Pi 4, Arduino Mega ve Arduino UNO olmak üzere 3 birimdir.

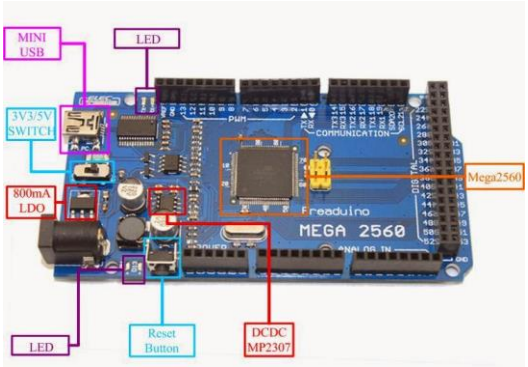
Raspberry Pi 4 4GB Teknik Özellikleri

- 1.5 GHz dört çekirdekli ARM Cortex-A72 CPU
- 4GB LPDDR4 RAM SKU | 2GB Modeli | 8GB Modeli
- VideoCore VI Grafikleri
- 4kp60 HEVC video
- Gerçek Gigabit Ethernet
- 2 × USB 3.0 ve 2 × USB 2.0 bağlantı noktaları
- 2 × mikro HDMI bağlantı noktası
- 5V/3A çalışmasını destekleyen USB-C güç girişi



Şekil-20: Raspberry Pi 4

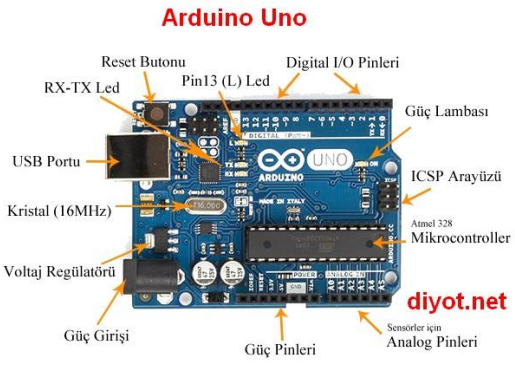
Arduino Mega 2560 Teknik Özellikleri



Şekil-21: Arduino Mega

- ATmega 2560 mikro işlemci denetleyicisine sahiptir.
- CH340 USB- Serial dönüştürücüsüne sahiptir.
- Dijital Giriş / Çıkış 54 adet pin deliği
- Analog Giriş / Çıkış 16 adet pin deliği
- Giriş Voltajı DC 7V ~ 12V
- Çıkış Voltajı DC 5V / 3.3 V
- 256 KB Flash hafızasına sahiptir.
- 16 Mhz Çalışma frekansı
- /ICSP çıkışı ve reset bulunmaktadır.
- PWM (Sinyal Genişlik Modülasyonu) Çıkışı 14 adet

Arduino UNO Teknik Özellikleri



Şekil-22: Arduino UNO

- Mikrodenetleyici: ATmega328
- Çalışma Gerilimi: 5V
- Giriş Gerilimi (önerilen): 7-12V
- Giriş Gerilimi (limit): 6-20V
- Dijital G/Ç Pinleri: 14 (6 tanesi PWM çıkışı)
- Analog Giriş Pinleri: 6
- Her G/Ç için Akım: 40 mA
- 3.3V Çıkış için Akım: 50 mA
- Flash Hafıza: 32 KB (ATmega328)
- SRAM: 2 KB (ATmega328)
- EEPROM: 1 KB (ATmega328)

5.1 Servo Motor

Servo motor üstünde bulunan bir parçayı 180 veya 360 derece döndürmeye yarayan son derece kullanışlı bir bileşendir. İKA üzerinde yabani ot imha sistemi için 1 adet servo motor kullanılmaktadır. Servo motor için örnek Arduino kodu yandaki gibidir:

```
#include <Servo.h>

Servo myservo;

int potpin = A0;
int val;

void setup() {
  myservo.attach(9);
}

void loop() {
  val = analogRead(potpin);
  val = map(val, 0, 1023, 0, 180);
  myservo.write(val);
  delay(15);
}
```

Şekil-23: Servo Motor için Örnek Arduino Kodu

5.2 Ultrasonik Mesafe Sensörü

Ultrasonik mesafe sensörünün çalışma prensibi polislerin kullandığı radar sistemleri ile aynıdır. Mesafeyi ölçmek için cisimlere bir taraftan ses dalgası gönderip diğer taraftan almaya dayalıdır. Bu bilgilere dayanarak mesafe ve zaman formülleri birimleri ile aşağıdaki gibidir:

Sesin Hızı:	Zaman:	Mesafe (santimetre cinsinden):
V_s = 343.2 m/s V_s = 0.034 $cm/\mu s$	$Zaman = mesafe/$hız $t = S/V_s = 10/0.034$ = 294.11 µs	$S = t * 0.034/2$

Bağlantılar yapılırken sensörün V_{cc} ucu 5V' a, GND ucu Arduino'nun GND'una diğer uçlar ise Arduino'nun PWM kısımlarından iki tanesine bağlanmalıdır. (Kod için 9 ve 10 numaralı uçlara bağlanmıştır.) Aşağıdaki koddan da anlaşılabileceği gibi yukarıdaki formüller kullanılarak zaman (duration değişkeni) ve mesafe (distance değişkeni) elde edilmiştir.

Yan taraftaki kod Arduino Mega'ya yüklendikten sonra kontrol için mesafe sensörüne herhangi bir cismi yaklaştırıp uzaklaştırarak mesafenin serial monitördeki değişimi gözlenmelidir. Eğer cismin yakınlaştırılıp uzaklaştırılmasına göre serial monitörde herhangi bir değişim olmazsa bu sensörün bozuk olması veya kodda bir hata olmasına bağlanabilir.

```
static const int trigPin = 9;
static const int echoPin = 10;
void setup() {
  pinMode(trigPin,OUTPUT);
  pinMode(echoPin,INPUT);
  Serial.begin(9600);
}
void loop() {
  digitalWrite(trigPin,LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin,HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin,LOW);

  duration = pulseIn(echoPin,HIGH);
  distance = duration * 0.034 / 2;
  Serial.print("Distance:");
  Serial.println(distance);
  Serial.print("Duration:");
  Serial.println(duration);
```

Şekil-24: USM Kullanımı Örneği

5.3 GPS Modülü

GPS modülü yardımıyla uydudan alınan veriler yardımıyla İKA'nın bulunduğu enlem ve boylam konumu elde edilmektedir. GPS modülünün çalışıp çalışmadığı kontrol etmek için aşağıdaki kod yeterli olacaktır:

Bu koda göre eğer GPS modülünden veri geliyorsa serial monitöre veri yazımı gerçekleştirilecektir. Serial monitöre yazdırılan değerlerden GPGGA ile başlayan satırda sonunda N (North) veya S (South) harfleri bulunan 1 adet ondalıklı sayının değeri, aracın konumunun Kuzey veya Güney olarak hangi paralellerde olduğunu göstermektedir. Aynı şekilde sonunda W (West) veya E (East) harfleri bulunan başka bir ondalıklı sayı ise aracın hangi Batı veya Doğu meridyenlerinde olduğunu göstermektedir. Bu bilgilerin doğruluğu telefondaki konum gibisi yardımıyla kontrol edilebilir.

```
void setup{
  Serial.begin(230400);
  Serial1.begin(9600);
}
void loop(){
  if(Serial1.available() == 1)
     Serial.write(Serial1.read());
}
```

Şekil-25: GPS Verilerinin Okunması

5.4 DC Motor Sürme Kontrolü

Arduino Mega 2560 mikrodenetleyicisi için DC motor sürme işlemi digitalWrite standart kütüphane fonksiyonu ile gerçekleştirilir. Bu fonksiyon ilk parametre olarak motorumuzun bağlı olduğu pili alırken ikinci parametresine LOW veya HIGH makroları geçilebilir. HIGH makrosu motoru döndürürken LOW makrosu motorumuzu durdurur. Motorun kontrolü de yine fonksiyonun HIGH makrosu ile çalıştırılıp LOW makrosuyla durdurulmasıyla yapılabilir.

Biz sadece bu proje için değil motorla yapılan herhangi bir projede kullanılmak üzere bir motor çalıştırma fonksiyonu yazdık. Bu fonksiyonun çağrısı ancak 4 argüman ile gerçekleştirilebilir. Çünkü kendisi 4 adet parametre değişkenine sahip bir fonksiyondur. Bu fonksiyonun ilk iki parametresi motorların durmasını, ileri veya geri dönmesini sağlamaktadır. Fonksiyona geçilen argüman 1 olursa motorun ileri dönmesi, 0 olursa motorun durması, -1 olursa ise motorun geri dönmesi işlemleri yapılmaktadır. Fonksiyonun diğer iki parametre değişkeni ise sırasıyla sol ve sağ motorların hızlarını argüman olarak almaktadır. Burada 0 minimum, 255 ise maksimum hız olarak değerlendirilir. Sözüne edilen fonksiyon yukarıdaki gibidir.

```
void motor_run(int motor_left_direction, int motor_right_direction, int speed1, int speed2) {
  switch(motor_left_direction){
     case 0:
        digitalWrite(left_motor_pin1, LOW);
        digitalWrite(left_motor_pin2, LOW);
        break;
    case -1:
        analogWrite(enA,speed1);
        digitalWrite(left_motor_pin1, LOW);
        digitalWrite(left_motor_pin2, HIGH);
        break;
    case 1:
        analogWrite(enA,speed1);
        digitalWrite(left_motor_pin1, HIGH);
        digitalWrite(left_motor_pin2, LOW);
        break;
  }
  switch(motor_right_direction){
     case 0:
        digitalWrite(right_motor_pin1, LOW);
        digitalWrite(right_motor_pin2, LOW);
        break;
    case -1:
        analogWrite(enB,speed2);
        digitalWrite(right_motor_pin1, LOW);
        digitalWrite(right_motor_pin2, HIGH);
        break;
    case 1:
        analogWrite(enB,speed2);
        digitalWrite(right_motor_pin1, HIGH);
        digitalWrite(right_motor_pin2, LOW);
        break;
  }
}
```

Şekil-26: Basit Motor Kontrol Komutları

5.5 LoRa ile Haberleşme Kontrolü

Özellikle günümüzde Internet of Things (Nesnelerin İnterneti) ile birlikte verilerin bir yerden başka bir yere aktarılması günümüzde çok önemli bir hale geldi. LoRa da aktarımda ciddi bir yere sahiptir. LoRa kelimesi aslında Long Range'ten gelmektedir. Buradan da anlaşılabileceği gibi LoRa teknolojisi uzak mesafeli veri transferini yapar ve bunu yaparken az güç tüketir. Ancak buna karşın verinin boyutları da daha küçük olmaktadır. LoRa veri almak için yandaki test kodu kullanılabilir:

```
#include <SoftwareSerial.h>

SoftwareSerial fixSerial(10, 11); // RX, TX

#define M0 7
#define M1 6

void setup() {
  Serial.begin(9600);
  fixSerial.begin(9600);

  pinMode(M0, OUTPUT);
  pinMode(M1, OUTPUT);
  digitalWrite(M0, LOW);        // E22 Modülleri için Aktif edin
  //digitalWrite(M0, HIGH);       // E32 Modülleri için Aktif edin
  digitalWrite(M1, HIGH);
}

void loop() {
  if (Serial.available()) {
    fixSerial.write(Serial.read());
  }

  if (fixSerial.available()) {
    Serial.write(fixSerial.read());
  }
```

Şekil-27: LoRa Kullanımı

5.6 Far-Mo Araç Arayüzü

Aracın anlık konum, hız, batarya durumu vb. verileri araçtan ara yüze; ara yüzdeki kontrol tuşlarından çıkan veriler ara yüzden araca olmak üzere haberleşme sistemi, çift taraflı olarak çalışmaktadır.

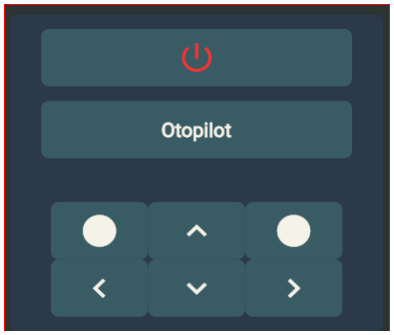
Şekil-28: İKA Arayüz Tasarımı

Ara yüzü geliştirirken Android, IOS, Windows gibi platformlarda uygulama geliştirebildiğimiz için Flutter geliştirme kitini kullandık. Uygulamamız Windows uygulaması olup araçla bağlantılı olan cihazın ara yüzünde o aracın verileri gözükmektedir. Ara yüz kolay kullanılabilmesi ve anlaşılabilmesi açısından olabildiğince sade dizayn edilmiştir. Araçtan gelen veriler, LoRa' dan SQL veri tipinde gelen veriler, üs bilgisayarında kullanılan Python programıyla Flutter`ın anlayacağı biçime dönüştürülerek ara yüze gönderilecektir. Aynı şekilde ara yüzden gelen veriler, bu program ile SQL veri tipine dönüştürülerek LoRa' ya gönderilecektir.

Aracın GPS sensöründen alınan veriler ve tarlanın köşelerinin koordinatları; ara yüz tarafında işlenerek Open Street Map üzerinden tarlanın sınırları, aracın anlık konumu ve aracın hareket güzergâhı gösterilecektir.

Ara yüz aracın anlık hız, ivme, rotasyon, batarya durumu ve deposundaki ilacın miktarını kullanıcıya göstermektedir. Log ekranı ile aracın anlık olarak ne durumda olduğu gösterilecektir.

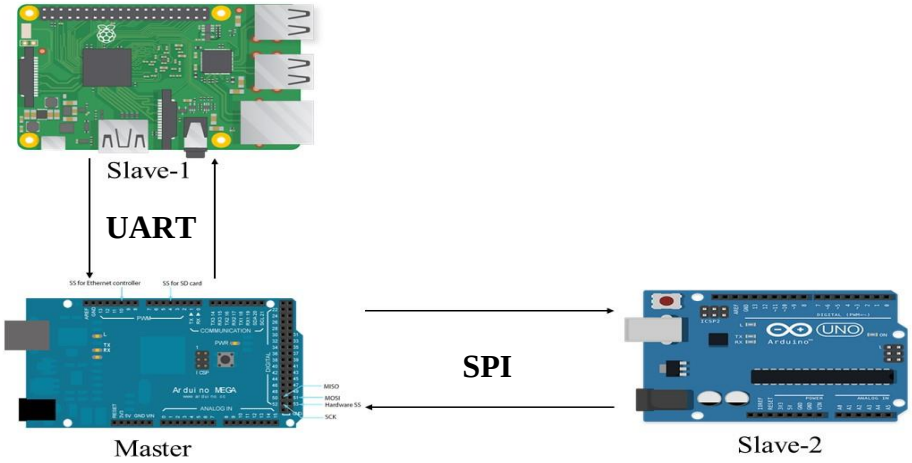
Ara yüzün yan tarafta bulunan bölümünde, aracın kontrol bölümü, aracın hangi kontrol modunda olduğunu gösteren ve modlar arasında geçiş yapılmasını sağlayan buton bulunmaktadır. İKA, gerekli durumlarda aracın hangi modda olduğunu gösteren tuşa basarak aracın ara yüz üzerinden kontrol edilmesi sağlanacak. Olağanüstü bir durumda ya da aracın durması gerektiğinde üstte bulunulan kapatma tuşuna basarak aracın durması sağlanacaktır.



Şekil-29: Arayüz İKA Kontrol Bölgesi

6. Otonom Sürüş Algoritmaları

Raspberry Pi 4; görüntü işleme uygulamaları ve nesne algılama için kullanılmıştır. Arduino Mega; motor sürme işlemi, mesafe sensörlerinin verilerini toplamak, sprey mekanizmasının kontrolü ve otonom sürüş karar mekanizması için kullanılmıştır. Arduino UNO; LoRa kablosuz haberleşme sistemini kontrol etmek, doğrudan ara yüze gidecek olan verileri sensörlerden toplamak ve Arduino Mega ile bağlantı oluşturmak için kullanılmıştır. Bu üç birimden Arduino UNO ve Mega SPI, Arduino Mega ve Raspberry Pi 4 UART ile master-slave çerçevesinde çalışmaktadır. (Şekil-30)



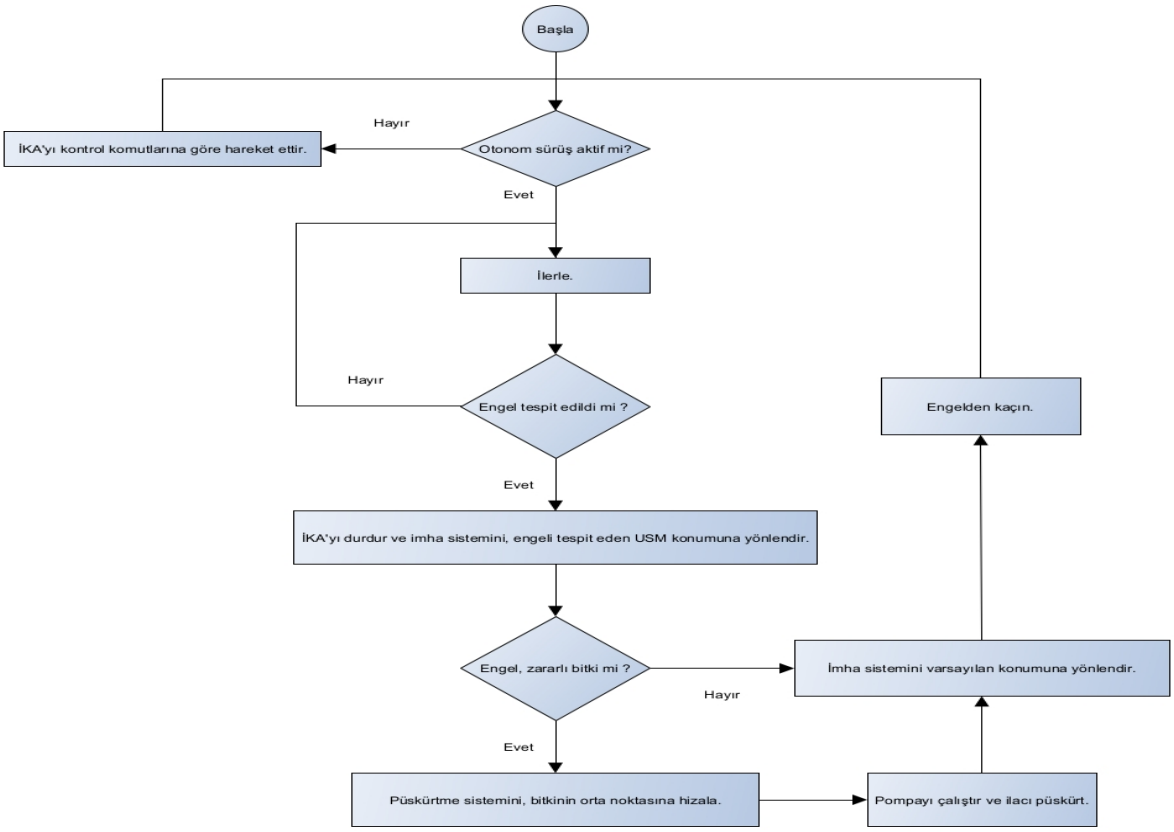
Şekil-30: Birimler Arası SPI ve UART Haberleşmesi

6.1 Yazılım süreçleri

Kullanılacak otonom sürüş algoritmasının basit temellere dayanıyor olması sebebiyle herhangi bir ara katman yazılımı tercih edilmemiştir. Master olarak kullanılan Arduino Mega'nın üzerinde çalıştırılan ana fonksiyon, tüm süreci yöneten ve denetleyen fonksiyondur. Ana fonksiyon kullanılarak denetlenebilirlik ve haberleşmenin düzeni sağlanmakla birlikte veri aktarımı için harcanan tepki süresi en aza indirgenmeye çalışılmıştır.

LoRa ile ana üsse gelen veriler, SQL veri tabanında saklanıyor olup aynı zamanda Flutter SDK'sı yardımıyla oluşturulan kullanıcı ara yüzünde kullanıcı ile buluşmaktadır. Ayrıca aynı yol üzerinden İKA'yla iletişim kurulup ara yüzden gelen veriler manuel ve otomatik kontrol arasında geçiş yapmayı mümkün kılmaktadır.

6.2 Otonom sürüş



Şekil-31: Ana Fonksiyonunun Sadeleştirilmiş Akış Şeması

Otonom sürüş için işlenen verilerin çoğunu, ultrasonik mesafe sensörü ve kızılötesi mesafe sensörü verileri oluşturmaktadır. Arduino Mega tarafından toplanan ultrasonik mesafe sensörü ve kızılötesi verileri ile otonom sürüş Böcek Algoritması & Temizlik Robotu biçiminde tasarlanmıştır. İKA'nın çevresini kaplayan sensörler aracılığıyla aracın konumlandırılması sağlanmıştır. Arduino Mega tarafından toplanan mesafe verileri bir dizi içerisine toplanmıştır. Ardından döngüler yardımı ile bu dizideki veriler denetlenir. Anormal bir veri alındığında veri indeks numarasına göre İKA motor kontrolünü sağlar. Bu algoritmanın testi ROS & Gazebo aracılığı ile simüle edilmiştir. Simülasyon ortamında test edilen algoritmayı, Arduino Mega'ya uygun olacak biçimde programlayıp ana fonksiyona eklenmiştir. Tüm bu durumlar çerçevesinde aracı genel olarak kontrol eden en önemli fonksiyon ana fonksiyondur. Bu ana fonksiyon içerisinde nesne algılama, kontrol modu ve diğer sensör verileri gibi İKA'nın durumunu belirleyen koşullamalar içermektedir. Bu süreç içerisinde Farm-O yabani ot tespit ettiğinde araç durur ve imha prosedürlerini uygular. Şekil-31'de ana fonksiyonunun sadeleştirilmiş akış şeması verilmiştir. Gerekli durumlarda kullanılmak üzere otonom sürüşe son verip manuel sürüşe geçilebilmesi için ara yüze anahtar eklenmiştir. Bu anahtar verisi LoRa aracılığı ile Arduino UNO'ya gelmektedir. Ardından SPI ile Arduino Mega'ya aktarılıp ana fonksiyon üzerinden işlemler gerçekleştirilmektedir. Manuel kontrole geçen araç için kontrol basit ok tuşları ile ifade edilmiştir.

6.3 Yabani Ot Tespiti Yazılımı

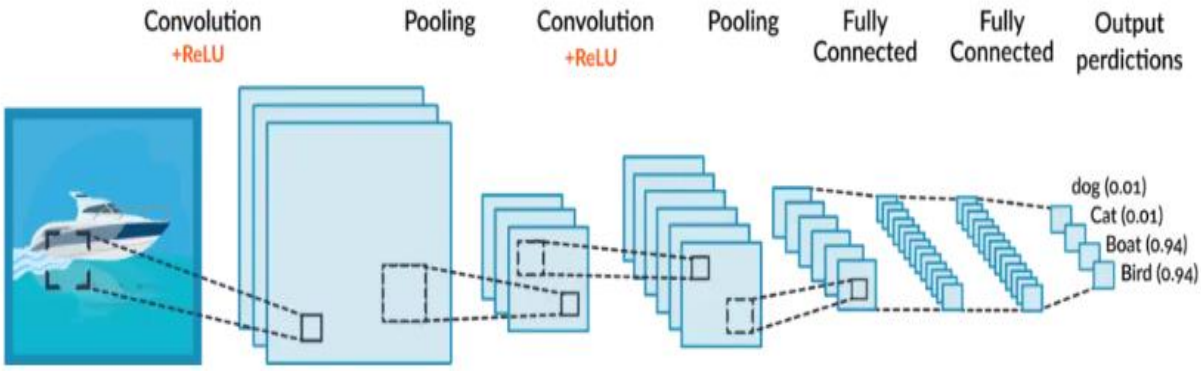
Yabani otların kamera tarafından algılanması için derin öğrenme tekniği kullanılacaktır. Görüntü işleme ve otonom sürüş algoritmaları Python programlama dili ile yazılmıştır. Görüntü işlemede bitkinin tespiti ve görüntünün işlenebilmesi için OpenCV ve NumPy kütüphanelerinden yararlanılmıştır. Model eğitimi için YOLO mimarisi seçilmiştir. YOLO seçilme sebebi ise diğer mimarilere göre eğitilen modellere nazaran daha çok doğruluk, daha yüksek FPS (Frames Per Second) sağlaması ve diğer algoritmalardan daha hızlı olmasıdır.

6.3.1 YOLO Nedir?

YOLO, konvolüsyonel sinir ağlarını (CNN) kullanarak nesne tespiti yapan bir algoritmadır. Açılımı "**You Only Look Once**", yani "**Sadece Bir Kez Bak**". Bu adın seçilmesinin nedeni algoritmanın nesne tespitini tek seferde yapabilecek kadar hızlı olmasıdır. YOLO algoritması çalışmaya başladığında görüntülerdeki veya videolardaki nesneleri ve bu nesnelerin koordinatlarını aynı anda tespit eder. YOLO'nun çalışma mantığını anlamak için ilk önce CNN (Convolutional Neural Network, Evrişimsel Sinir Ağları) terimini anlamak gerekir.

6.3.2 CNN Nedir?

CNN genellikle görüntü işlemede kullanılan ve girdi olarak görselleri alan bir derin öğrenme algoritmasıdır. Farklı operasyonlarla görsellerdeki feature'ları (özellikleri) yakalayan ve onları sınıflandıran bu algoritma farklı katmanlardan oluşmaktadır. Convolutional Layer, Pooling ve Fully Connected olan bu katmanlardan geçen görsel, farklı işlemlere tabii tutularak derin öğrenme modeline girecek kıvama gelir. CNN modelleri oluştururken, unstructural (düzensiz) veri ile uğraştığımızdan klasik makine öğrenmesi algoritmalarına kıyasla veri ön işleme kısmında çok uğraşmamaktayız.



Şekil-32: CNN

6.3.3.1 Convolution Layer

Convolutional (evrişim katmanı) CNN algoritmalarında görüntüyü ele alan ilk katmandır. Bilindiği üzere görseller aslında içlerinde belirli değerler taşıyan piksellerden oluşan matrislerdir. Evrişim katmanında da orijinal görsel boyutlarından daha küçük bir filtre görselin üzerinde gezer ve bu görsellerden belirli özellikleri yakalamaya çalışır.

6.3.3.2 ReLU

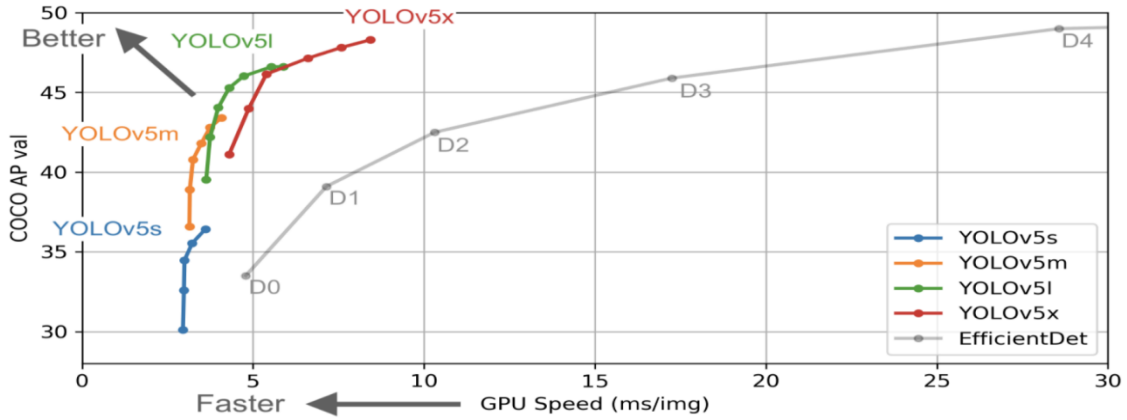
ReLU (Rectified Linear Unit) f(x) = max(0, x) şeklinde çalışan doğrusal olmayan bir fonksiyondur. Örnek vermek gerekirse -25 değerini alan bir ReLU fonksiyonu çıktı olarak 0'ı, 25 değerini alan bir fonksiyon ise 25'i verir. Ana amacı negatif değerlerden kurtulmak olan ReLU CNN'lerde oldukça önemli bir konumdadır.

6.3.3.3 Pooling

Evrişimli katman gibi pooling (havuzlama) katmanı da boyutsallığı azaltma amacındadır. Bu sayede hem gereken işlem gücü azalır hem de yakalanan gereksiz özellikler yok sayılarak daha önemli özelliklere odaklanılır. CNN modellerinde genellikle kullanılan iki farklı pooling tekniği vardır. Bunlardan biri Max (Maksimum) diğeri de Average (Ortalama) pooling'tir.

6.3.3.4 Fully Connected Layer

Bu katmanda diğer katmanlardan geçen resimler matristen vektör haline getirilir. YOLO'nun sürümleri arasından YOLOv5 seçilmiştir. Bu sürümü seçmemizdeki asıl neden diğer sürümlerinden görüntüyü daha hızlı işleyebilmesidir. (Şekil-33)



Şekil-32: YOLO Modellerinin Kıyaslaması

Model eğitimi için ilk başta Teknofest tarafından gönderilen yabani ot örneği için veri seti (dataset) oluşturulmuştur. Bu veri seti için yaklaşık 1200 tane yabani ot fotoğrafı içeren bir klasör oluşturulmuştur. Bu fotoğraflar gönderilen yabani otun toprağa yerleştirilip günün farklı zamanlarında çekilmiştir. Veri setinin daha zengin olması için Data Augmentation (Veri Zenginleştirme) algoritması kullanılmıştır.

6.3.4 Data Augmentation Nedir?

Data Augmentation algoritması veri setinde var olan fotoğraflar üzerinden küçük değişiklikler yaparak (zoom, rotate, padding, cropping, grayscaling, contrast, random erasing) veri seti için yeni veriler ortaya çıkarır. Bu algoritma makine öğrenmesinde doğruluğu önemli ölçüde arttırmaktadır [1]. Bu algoritma sayesinde daha kolay ve daha etkili fotoğraflar üretilebiliyor.

Şekil-34: Veri Seti İçin Örnek Görsel

Şekil-33: Veri Etiketleme İşlemi

Çekilen fotoğraflara daha sonra makesense.ai sitesinde label'lar eklenmiştir ve bir sonraki aşama yani eğitim aşamasına geçilmiştir. Bu aşama için gereken .txt uzantılı dosyalar bu aşamada alınmıştır.

Model eğitimini gerçekleştirmek için Google Colab (Colaboratary) ortamı kullanılmıştır. Google Colab, yapay zekâ ve derin öğrenme projeleri üzerinde çalışanlar için etkileşimli, bulut tabanlı, kullanımı kolay, ücretsiz, kurulum gerektirmeyen ve ortak çalışmaya dayalı bir programlama ortamıdır. Bu aşamada .cfg ve. weights dosyası elde edilmiştir. Sonraki aşama ise Python dilinde OpenCV ve NumPy kütüphanelerinden yararlanarak görüntü işleme kodlarının yazılması ve testler yapılmasıdır.

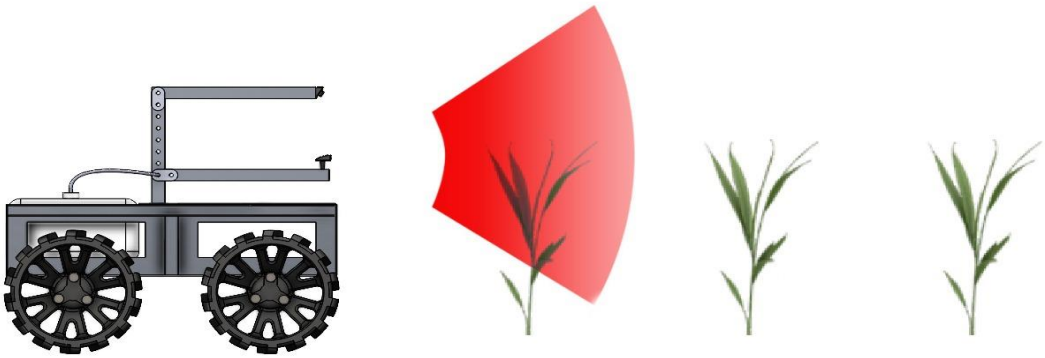
Şekil-34: Model Test Sonucu

6.4 Yabani Ot İmha Algoritması

Mesafe verileriyle algılanan engele doğru yönelen kamera, engelin yabani ot olup olmadığını yabani ot tespit algoritmasını kullanarak kontrol etmektedir. Eğer engel yabani ot ise yabani otu çerçeveleyen dörtgenin merkez noktası hesaplanır. Ve bu noktanın X parametresi ile kameranın orta noktasının X parametresi eşitlenene dek servo motorun o yöne hareket etmesi sağlanır. Bu işlem Raspberry Pi 4 ve Arduino Mega arasındaki veri aktarımıyla gerçekleşir.

7. Yabani Ot Zirai Mücadele Yöntemleri

7.1 Fıskiye Sistemi



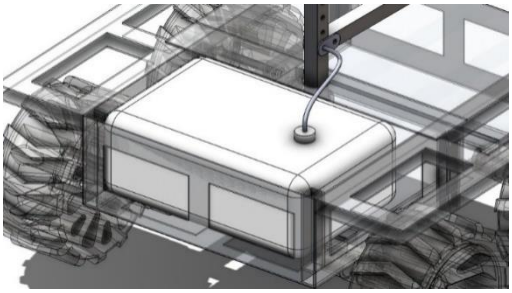
Şekil-35: Fıskiye Şeması

Depoda bulunan ilacı Topran marka Opel Meriva A Silecek Su Fıskiye Motoru olan pompamız yardımıyla sigma profilden imal edeceğimiz ve servo motor yardımı ile yönünü değiştirebileceğimiz, namlunun ucunda bulunan uç kısmındaki nozzle'ı 360 derece açıda istenilen eksende ayarlanabilir olması ve suyu sprey şeklinde homojen bir biçimde atması sebepleriyle tercih ettiğimiz araç silecek fıskiyesine gönderip, bu fıskiye ile püskürtmeyi sağlayacağımız sistemdir.

- Araç silecek fıskiyesinin bağlı olduğu namlunun yüksekliği basit bir şekilde ayarlanabilir.
- Sistemi çeviren servo motor 180 ° dönme aralığına sahip olup 10,00 kg-cm tork ve 0,20sn/60° hız ile dönmektedir
- Fıskiye sisteminde yüksekliğin ayarlanabildiği servo motora bağlı olan sigma profili boyutu 270 mm'dir.
- Bitkileri ilaçlamak için imal ettiğimiz namlu 270 mm'dir.
- İlaçlamanın yapıldığı namluyu uzun tutarak rüzgâr gibi çeşitli dış etmenlerin ilaçlamayı negatif etkilemesini engelledik.

7.2 İlaç Deposu

Aracın tespit ettiği yabani otları ilaçlamada kullanacağı kimyasal maddeyi sakladığı 300x200x110 boyutlarındaki haznedir, 4 litre sıvı hacimli bir ilaç deposu kullanılacaktır, deponun ortasına delikli bir plastik koyarak suyun geçişleri kırılıp aracın dengesini bozması engellenecektir.



Şekil-36: İlaç Deposu

8. Özgün Bileşenler

8.1 Elektronik Özgünlükler

8.1.1 Lora Haberleşme Modülü

Yaptığımız araştırmalar sonucunda LoRa haberleşme modülünü bizim kadar aktif kullanan bir takım ile karşılaşmadık; araçta LoRa'nın temel görevi gerekli verilerin arayüze sağlıklı bir şekilde aktarılması ve aracın kablosuz şekilde manuel olarak kontrol edilmesi, düşük güç tüketimi ile veri akışını sağlaması, radyo frekansının kullanılması, diğer alternatiflere nazaran daha güvenilir olduğunu düşünmemiz ve uzun mesafelerden araç kontrolünün sağlanması (3km-5km) ve benzeri sebeplerden dolayı LoRa teknolojisini İKA'da kullanmaya karar verdik.

8.1.2 Akım Sensörü

Araçta bulunan bazı elektronik aksanların özellikle motorların toprak arazisinde çalışacağından dolayı biraz fazla akım çekmesi gerekir fakat gereğinden fazla akım çekerse hem aracın güvenliği için hem de tarlanın tahrip olmaması için gücün kesilmesi gerekir.

Bu sebepten dolayı diğer takımlardan farklı olarak sigorta kullanmak yerine sistemin daha düzgün çalışmasını denetleyecek bir akım sensörü kullanmaya karar verdik.

8.2 Yazılımsal Özellikler

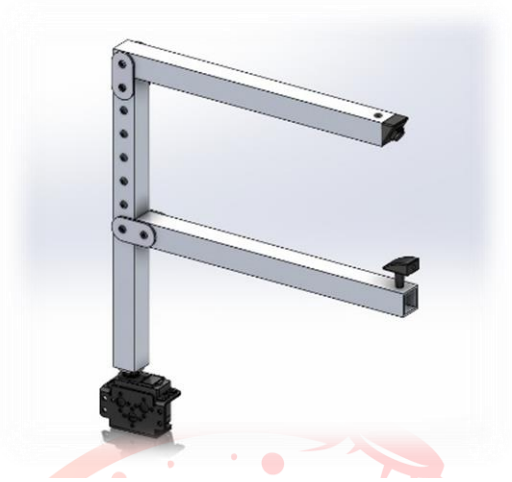
İKA'nın kullanıcıya doğrudan yansıyan özgün özelliği ara yüzü programlarken kullandığımız Flutter geliştirme kitidir. Flutter Google'ın geliştirdiği Multi-Platform destekli, açık kaynak kodlu bir geliştirme kitidir [4]. Flutter Widget yapısı, açık kaynak koduna sahip olması ve Multi-Platform desteğine sahip olması bizim Flutter'ı tercih etmemizdeki en önemli sebeplerdendir. Bahsedilen bu özellikler sayesinde İKA'nın IOS, Web, Android, Desktop üzerinde benzer arayüzler ile desteklenebilmesi yazılımımızın özgün özelliğidir.

8.3 Mekanik Olarak Özgün Bileşenler

Aracın tamamı tasarım ekibimiz tarafından özgün olarak bilgisayar ortamında çizildikten sonra imal edilmeye başlanmıştır.

1. Kullandığımız deponun ortasına delikli bir plastik ekleyerek suyun deponun içerisinde geçişlerini yavaşlattık. Böylece depodaki sıvının aracın dengesine etkisini en az düzeye indirmeye çalıştık.

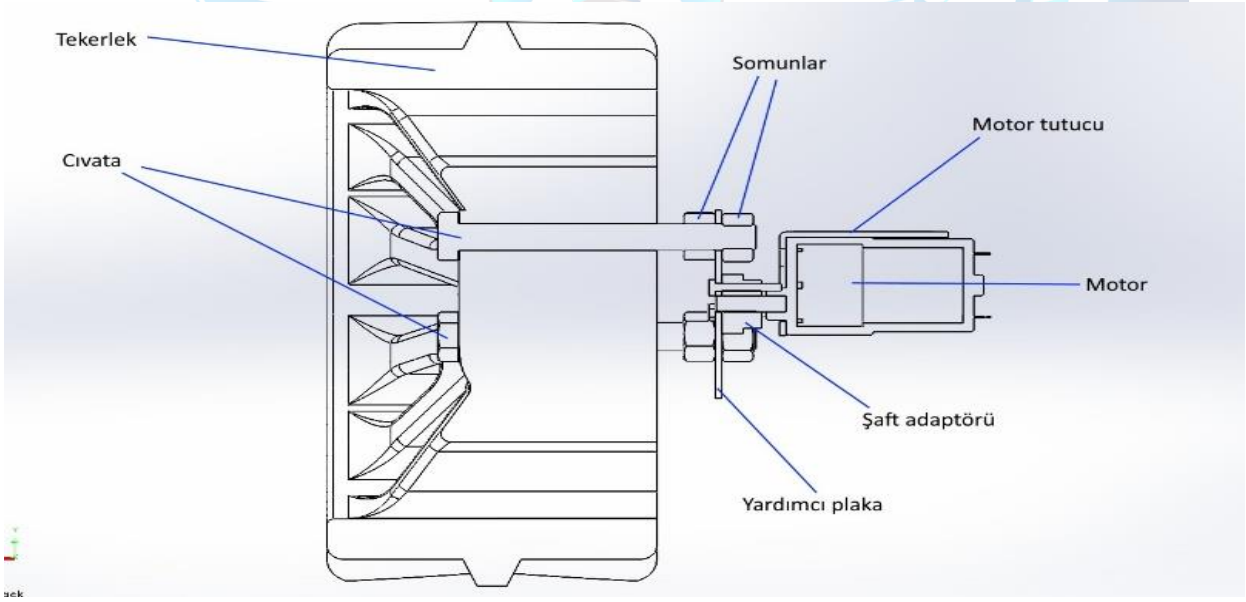
2. Fıskiye motorumuzda aynı şekilde araç fıskiyesi motoru olup suyu ilerlere iletme sorunuyla karşılaşmadan istenilen yere ilaç ulaştırılacaktır.



Şekil-37: Fıskiye Tasarımı

3. Aracın fıskiye sisteminde sigma profilden oluşan namlu kısmının yüksekliğinin değiştirilebilmesi için sigma profilde delikler açılmıştır ve namlu buradaki istenen deliğe kolay bir şekilde el ile monte edilebilir.

4. Teker bağlantısı için yardımcı bir plaka bulunuyor ve tekerimize hâlihazırda açılmış 4 deliği kullanarak 4 yerden tekeri tutan bir bağlantı yöntemi kullandık.



Şekil-40: Teker Bağlantısı

5. Araç için özel bir şase tasarladık. Bunun sebebi ise araca gelen kuvveti yayarak mukavemeti arttırmak oldu.

9. Güvenlik Önlemleri

MEKANİK GÜVENLİK ÖNLEMLERİ			
OLABİLECEK HATA	OLASILIK 1-DÜŞÜK 2-ORTA 3-YÜKSEK	ETKİSİ 1-DÜŞÜK 2-ORTA 3-YÜKSEK	**ÖNLEM**
Araba ile iletişimimiz koparak istenmeyen hareketler yapması.	**1**	**3**	Kolay ulaşılabilir bir acil durum butonu ile direk güç kesilerek araç durdurulur.
Kimyasal ilaç deposundan bir sızıntı sonucu elektronik aksanların zarar görmesi.	**2**	**3**	Aracın deponun altında kalan kısmını sızıntı durumunda sıvıyı dışarı atabilmesi için mazgal şeklinde imal edilecek.
Yağış veya aracın ıslanması durumunda elektronik aksanların zarar görmesi.	**3**	**3**	Elektronik aksanların konumlandırması bu hususta dikkate alınarak açıkta kalanlar için ise bir kapak imal edilecektir.

ELEKTRONİK GÜVENLİK ÖNLEMLERİ			
OLABİLECEK HATA	OLASILIK 1-DÜŞÜK 2-ORTA 3-YÜKSEK	ETKİSİ 1-DÜŞÜK 2-ORTA 3-YÜKSEK	**ÖNLEM**
Güç kaynağının veya mikro işlemcilerin fazla ısınması .	**2**	**3**	Güç kaynağının veya mikro işlemcilerin ısısını ölçmek için bir adet ısı ölçer sensör yardımıyla bilgi alarak riskli durumlarda aracın durdurulması.
Motorların fazla akım çekerek yanması	**2**	**3**	Akım rölesi kullandık bunu tercih etme sebebimiz aracın bazı yokuşları çıkarken motorlara biraz daha yüklenmek istemesi ile birlikte sigortanın bizi şaşırtabileceğini düşünüyoruz.

YAZILIMSAL GÜVENLİK ÖNLEMLERİ			
OLABİLECEK HATA	OLASILIK 1-DÜŞÜK 2-ORTA 3-YÜKSEK	ETKİSİ 1-DÜŞÜK 2-ORTA 3-YÜKSEK	**ÖNLEM**
Otonom sürüş algoritması bazı durumlarda karar verebilir veya yanlış karar verebilir	**2**	**1**	Aracımızın verilerini aldığımız ara yüz aracılığı ile manuel sürüş yapılabiliyor.
Aracın otonom sürüş algoritmasını denerken araç kendisine ve çevresine zarar verebilir.	**3**	**2**	Aracımızı yola çıkarmadan önce otonom sürüş algoritmasını bir simülasyon ortamında test ederek fiziksel mekanda denediğimiz zaman olabildiğince az hata ile karşılaşmayı hedefliyoruz.

10. Simülasyon ve Test

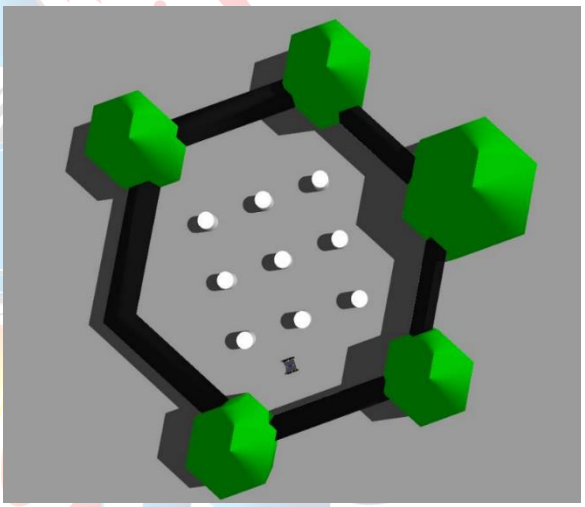
Simülasyon ortamında aracımız için seçtiğimiz her parçanın mukavemeti hesaplanmıştır, algoritma test edilmiştir ve aracın elektronik parçalarının hem işlemcilerle hem güç kaynağımız ile entegre kullanılması için gerekli değerlendirmeler yapılmıştır. Tedarik etmiş olduğumuz elektronik parçaların ilk aşamada güvenlik testleri, kısa devre testleri, çektiği akım ve harcadığı güç laboratuvar ortamında test edilmiştir. İkinci aşamada ise elimizde bulunan sensörlerin, motorların ve işlemcilerin bir test aracında entegrasyonu tamamlanmıştır. Gerekli yazılımsal ve elektronik testlerin büyük bir kısmını laboratuvar ortamında geri kalan testleri ise simülasyon ortamında gerçekleştirmiş bulunmaktayız.

10.1 Redüktörlü DC Motor testleri

Aracımızda 12V, 37RPM, Zorlama Torku: 2.2kg x m, Yükleme Torku: 8kg x cm ve çektiği akım 130 ± 40mA olan 4 adet DC motor kullanacağız. Motorlarımız için RPM testi ve voltaj testi uyguladık çıkan sonuçların teorik değerleri karşıladığını gözlemledik.

10.2 Simulasyon Ortamı Gazebo

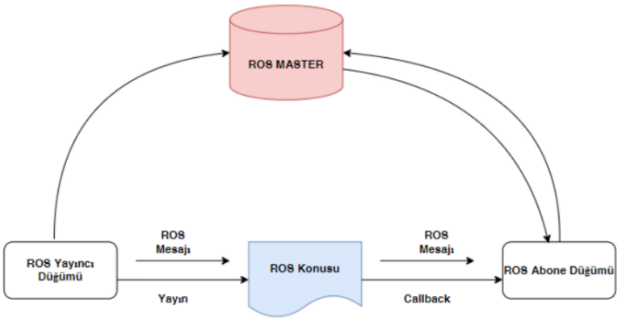
Algoritma simülasyon testi için simülasyon ortamı olarak Gazebo tercih edilmiştir. Gazebo'nun tercih edilme sebeplerinden biri ROS entegrasyonuna sahip olmasıdır. Gazebo, ROS paketleri ve Gazebo eklentileri sayesinde robot projelerinin simülasyonu için gerekli koşulları ve kolaylığı sağlamaktadır. Dinamik simülasyon, gelişmiş grafikler, sensör kitaplıkları, eklentiler, hazır robot modelleri, uzak sunucu desteği ve komut satırı araçları gibi özelliklere sahiptir. İKA algoritma testi için diff-drive eklentisine sahip Diff-Drive [1] isimli açık kaynaklı projeden yararlanılmıştır. Simülasyonun testi, turtlebot3_world dünyasında yapılmıştır. (Şekil-41)



Şekil-38: Turtlebot3_world

10.3 ROS (Robot Operating System)

ROS, adının aksine bir işletim sisteminden ziyade görselleştiricidir. ROS, publisher/subscriber (yayıncı/abone) modelinde düğümler ile haberleşme üzerine kuruludur. Bu düğümlerin arasında bağımsız ROS Master bulunmaktadır. (Şekil-42)



Şekil-39: ROS Çalışma Prensibi [2]

Ros Master'a kaydedilen düğümler arasında gerçekleşen haberleşme, konu (topics) yayını yapan düğümler ve konuya abone olan düğümler arasında gerçekleşir. Bir

konuya birden fazla düğüm abone olabilmektedir. Bu düğümler dilden bağımsızdır ve farklı dillerde yazılan düğümler ile haberleşme sağlamanın avantajı göz önünde bulundurulmalıdır.

Giderek gelişen sensör teknolojisi sayesinde robotlarda kullanımı artan sensörlerin verilerini yönetebilmek için ROS görselleştiricisi gibi ara katman yazılımı kullanmak hata ayıklamayı kolaylaştırmaktadır.

İKA üzerinde birçok farklı sensör kullanılsa bile bu verilerin tümünü işleyecek ve ROS'u stabil çalıştırabilecek donanıma sahip olunmadığı için tercih edilmemiştir. Simülasyon ortamında gerçekleştirilen testlerde yayını yapılan konular *rostopic list* ile listelenmiş olup çıktı şekil-43'te verilmiştir.

```
mericsetan@mericsetan-HP-Pavilion-Gaming-Laptop-15-cx0xxx:~$ rostopic list
/Diff_Drive/diff_drive_controller/cmd_vel
/Diff_Drive/diff_drive_controller/odom
/Diff_Drive/diff_drive_controller/parameter_descriptions
/Diff_Drive/diff_drive_controller/parameter_updates
/camera/depth/image_raw
/camera/depth/points
/camera/rgb/camera_info
/camera/rgb/image_raw
/camera/rgb/image_raw/compressed
/camera/rgb/image_raw/compressed/parameter_descriptions
/camera/rgb/image_raw/compressed/parameter_updates
/camera/rgb/image_raw/compressedDepth
/camera/rgb/image_raw/compressedDepth/parameter_descriptions
/camera/rgb/image_raw/compressedDepth/parameter_updates
/camera/rgb/image_raw/theora
/camera/rgb/image_raw/theora/parameter_descriptions
/camera/rgb/image_raw/theora/parameter_updates
/clicked_point
/clock
/depth/camera_info
/gazebo/link_states
/gazebo/model_states
/gazebo/parameter_descriptions
/gazebo/parameter_updates
/gazebo/performance_metrics
/gazebo/set_link_state
/gazebo/set_model_state
/initialpose
/joint_states
/move_base_simple/goal
/rosout
/rosout_agg
/scan
/tf
/usm/usm_LFM
/usm/usm_LF
/usm/usm_LM
/usm/usm_MB
/usm/usm_RFM
/usm/usm_RF
```

Şekil-43: Rostopic List Çıktısı

Listelenen konuların içerisinden birini görüntülemek için *rostopic echo /[konuAdı]* komutu çalıştırıldığında yayınlanan verileri görmek mümkündür. Solda verilen şekil-44, ultrasonik mesafe sensörünün örnek yayınıdır. Ayrıca Gazebo aracılığı ile sensör verilerinin gerçek dünya koşullarında benzetilmiş olup gürültülü bir şekilde yayınlandığı gözlemlenmektedir.[3]

```
header:
  seq: 17
  stamp:
    secs: 163
    nsecs: 694000000
  frame_id: "usm_LF"
radiation_type: 0
field_of_view: 0.5
min_range: 0.009999999776482582
max_range: 4.0
range: 0.5592813491821289
```

Şekil-44: Örnek Sensör Mesajı

10.4 Simülasyon Algoritması

Algoritma simülasyonu için Python programlama dili kullanılarak hem abone hem de yayıncı biçiminde kodlama yapılmıştır. Otonom sürüş algoritmaları bölümünde anlatıldığı üzere İKA, mesafe verilerine göre sürüşünü gerçekleştirmektedir. Bu sebepten dolayı mesafe verileri bir dizi içerine kaydedilmiştir. (Şekil-45) Mesafe verilerinde anormallik tespit edildiğinde konumu bilinen sensöre göre İKA'ya cmd_vel konusunda Twist() tipinde mesaj (hız mesajı) gönderilip hareket etmesi sağlanmıştır.

```python
def ultrasonicMF(self, mid_front):
  self.allUsm[3] = round(mid_front.range,3)


def ultrasonicRF(self, right_front):
  self.allUsm[4] = round(right_front.range,3)


def ultrasonicRM(self, right_mid):
  self.allUsm[5] = round(right_mid.range,3)
  self.autoCtrl()


def rosBasla(self):
  #USM sensör verilerine abone ol
  rospy.init_node("ultrasonic_sensor")
  rospy.Subscriber("usm/usm_LF",Range,self.ultrasonicLF)
  rospy.Subscriber("usm/usm_LM",Range,self.ultrasonicLM)
  rospy.Subscriber("usm/usm_MB",Range,self.ultrasonicMB)
  rospy.Subscriber("usm/usm_RFM",Range,self.ultrasonicMF)
  rospy.Subscriber("usm/usm_LFM",Range,self.ultrasonicRM)
  rospy.Subscriber("usm/usm_RF",Range,self.ultrasonicRF)
  rospy.Subscriber("usm/usm_RM",Range,self.ultrasonicRM)


  self.pub = rospy.Publisher("Diff_Drive/diff_drive_controller/cmd_vel",Twist,queue_size=10)
  self.hiz_mesaji = Twist()

  rospy.spin()
```

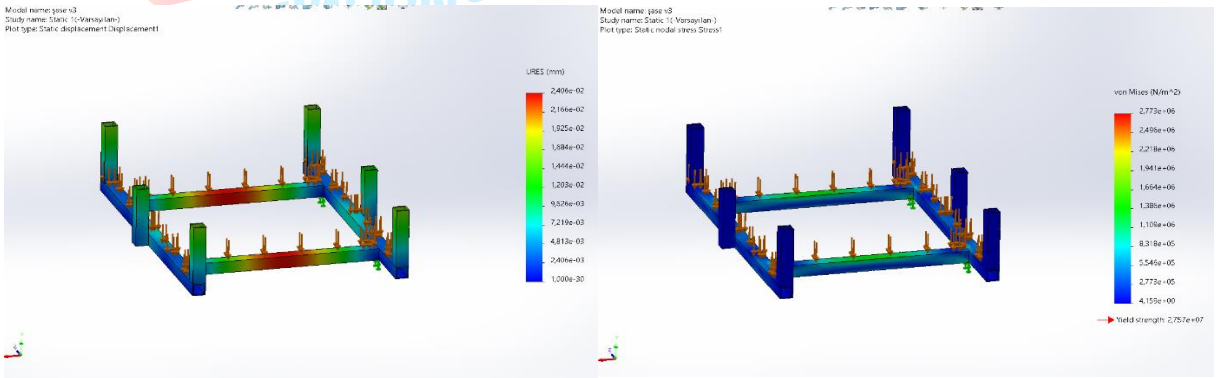
Şekil-45: Mesafe Verilerine Abone Hız Verilerine Yayıncı Düğümü (Python)

10.5 Mukavemet Testleri

Aracımızın önemli parçalarını SolidWorks' ün simülasyonundan yararlanarak dayanıklılık testini gerçekleştirdik. Bu testlerde stres testini aracımızın ağırlığına denk gelen 15 kg ile test ettik. Yer değiştirme testinde ise aracımızın ağırlığının 2 katına eşit olan 30 kg ile testi gerçekleştirdik. Bunun sebebi aracımız hareket halindeyken alabileceği darbelere önlem amaçlıdır.

10.5.1 Şase Mukavemet Testi

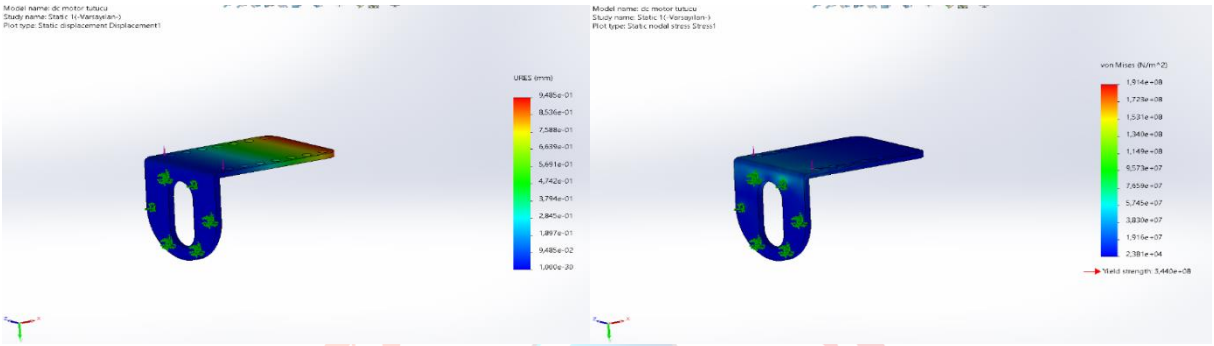
Aracımızın iskeleti olan şase bütün donanımların ağırlığını taşıyacağı parçadır. Şasemiz sigma profillerin gerekli ölçülerde kesilip vida ile birleştirilerek üretilecektir. Şasemiz motor tutucuların bulunduğu yerden sabitlenerek teste tabi tutulmuştur. Stres testinde şaseye uygulanan 147 Newton sonucunda şasenin aracın ağırlığının 10 katına kadar dayanabildiği, yer değiştirme testinde ise şaseye uygulanan 294 Newton sonucunda şasenin en kırmızı bölgesinde 0,02 mm gibi küçük bir değerde deformasyona uğradığı gözlenmiştir.



Şekil-46: Şase Mukavemet Simülasyonu

10.5.2 DC Motor Tutucular Mukavemet Testi

Motor tutucular araca hareket kabiliyeti kazandıran motorların şaseye bağlanmasında görev alan parçadır. Parçamız seçilirken materyaline ciddi önem verilerek seçilmiştir. Stres testinde motor tutucuların birine düşen aracımızın ağırlığının ¼ 'ü olan 37 Newton uygulanmış ve sonucunda motor tutucunun üstüne düşen yükün 1,8 katına kadar dayanacağı, yer değiştirme testinde ise motor tutuculara aracın ağırlığının ½ katı olan 74 Newton uygulanmış ve sonucunda maksimum 0,9 mm değerinde deformasyona uğradığı gözlenmiştir.



Şekil-47: DC Motor Tutucu

10.6 Prototip Araç ile Yapılan Testler

Aracın tüm testlerini, mühendislik hesaplarını gerek simülasyon ortamında gerek kâğıt üzerinde tamamladık fakat biz mühendislik öğrencileri biliyoruz ki; fiziksel, uygulamalı testler yapmadan bu araç gerekli yetkinliklerde çalışmayacaktır; bu sebepten fotoğrafta görmüş olduğunuz 2. Prototipimiz (Şekil-49) olan mini İKA'da şu testleri yaptık;

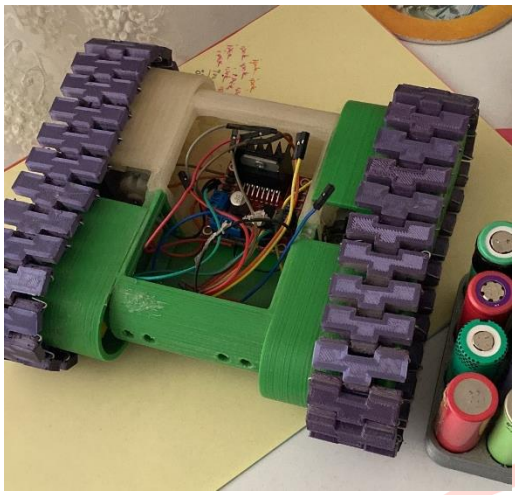
10.6.1 Otonom Sürüş İçin Yapılan Testler

- Mesafe sensörleri engel gördüğünde motorlarımız nasıl reaksiyon veriyor ?
- Eğim sensörümüz bulunduğu ortama ne kadar sürede kalibre oluyor ve eğim sensörüne bağlı olarak motorlarımızın tepkisi nedir ?
- Encoderden gelen verilere bağlı aracın maksimum hızı durma süresi ve durma safhasında aldığı yol vb. değerler hesaplandı. Bu hesaplanan değerler beklentimiz karşılıyor mu ?

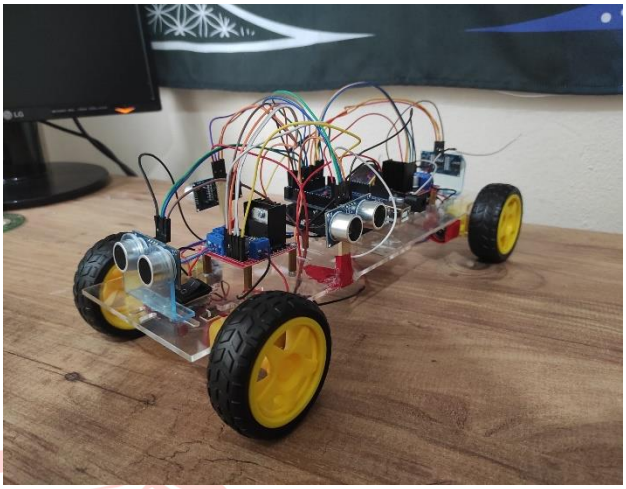
Gibi soruları prototip aracımız üzerinden kendimize sorduk ve otonom sürüş algoritmasını test etme fırsatını bulduk.

10.6.2 Yabani Ot Tanımak İçin Yapılan Testler

Eğittiğimiz modeli Raspberry Pi işlemcimize gömmemizin ardından kamera ve servo motor bağlantılarını sağladık ve ilk safhada eğitim için kullandığımız fotoğraflar ile servo çalışma mekanizmasını test ettik. Ardından kameramızın gördüğü yeni görüntüler ile servo motorumuzun çalışmasını gözlemledik; yaptığımız ilk testlerden oldukça olumlu sonuç almamıza rağmen aracımızın her koşulda yabani otu tanıması için gerekli çalışmalarımıza devam etmekteyiz.



Şekil-48: Prototip 1



Şekil-49: Prototip 2

11. Kaynakça

[1] https://github.com/parambeernegi/robot_arm_ros/tree/master/diff_drive

[2] https://dergipark.org.tr/tr/download/article-file/1857266

[3] https://classic.gazebosim.org/tutorials?tut=sensor_noise&cat=sensors#IMUnoise

[4]https://flutter.dev/?gclid=Cj0KCQjwheyUBhD-ARIsAHJNM-P1OkW8z-nRS5254zzuZbZIK6rHm_UtfWBDOqd52XIfNkNy1HhVQ50aAhRlEALw_wcB&gclsrc=aw.ds

[5] https://fixaj.com/

[6]https://batuhandaz.medium.com/yolo-algoritmas%C4%B1na-genel-bak%C4%B1%C5%9F-ve-r-cnn-deep-learning-3cdb0a6e8455

[7] https://teknoloji.org/cnn-convolutional-neural-networks-nedir/

[8] https://www.projehocam.com/arduino-spi-haberlesme/.

[9] https://www.tarnet.com.tr

[10] http://wiki.ros.org/navigation/Tutorials/RobotSetup

[11] https://yasincapar.com/tr/calismalar/cae-2/solidworks-simulation-tr/

[12] https://www.translatorscafe.com/unit-converter/tr-TR/calculator/dc-power/

(2021, Teknofest, Tarımsal İnsansız Kara Aracı Yarışması, Space and Beyond Technology)