KTO KARATAY ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
MEKATRONİK MÜHENDİSLİĞİ ANABİLİM DALI
TEZLİ YÜKSEK LİSANS PROGRAMI

MOBİL ROBOTLARDA GENİŞLETİLMİŞ KALMAN FİLTRESİ KULLANILARAK KAPALI ALAN HARİTALANDIRMA VE NAVİGASYONDA İYİLEŞTİRME

Eren OKUR

Yüksek Lisans Tezi

KONYA
Ocak 2023

MOBİL ROBOTLARDA GENİŞLETİLMİŞ KALMAN FİLTRESİ KULLANILARAK KAPALI ALAN HARİTALANDIRMA VE NAVİGASYONDA İYİLEŞTİRME

Eren OKUR

KTO Karatay Üniversitesi
Lisansüstü Eğitim Enstitüsü
Mekatronik Mühendisliği Anabilim Dalı
Mekatronik Mühendisliği Tezli Yüksek Lisans Programı

Yüksek Lisans Tezi

Tez Danışmanı: Prof. Ali Bülent UŞAKLI

Konya
Ocak 2023

BİLDİRİM

Enstitü tarafından onaylanan Yüksek Lisans tezimin tamamını veya herhangi bir kısmını basılı veya dijital biçimde arşivleme ve aşağıda belirtilen koşullar dahilinde erişime açma iznini KTO Karatay Üniversitesine verdiğimi bildiririm. Bu izinle, Üniversiteye verilen kullanım hakları dışındaki tüm fikri mülkiyet haklarım bende kalacak ve gelecekteki çalışmalar (makale, kitap, lisans, patent vb.) için tezimin tamamının veya bir bölümünün kullanım hakları yalnızca bana ait olacaktır.

Tezimin bütünüyle kendi çalışmam olduğunu, başkalarının haklarını ihlal etmediğimi ve tezimin tek yetkili sahibi olduğumu beyan ve taahhüt ederim. Telif hakkı bulunan ve sahiplerinden yazılı izinle kullanılması zorunlu olan kaynakları, yazılı izin alarak kullandığımı ve istenildiğinde izinlerin suretlerini Üniversiteye teslim etmeyi taahhüt ederim.

Yükseköğretim Kurulu tarafından yayımlanan "Lisansüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge" kapsamında, tezim, aşağıda belirtilen koşullar haricince, YÖK Ulusal Tez Merkezi ve KTO Karatay Üniversitesi Açık Erişim Sisteminde erişime açılır.

☐ Enstitü / Fakülte Yönetim Kurulu kararı ile tezimin erişime açılması mezuniyet tarihimden itibaren 2 yıl ertelenmiştir.[1]

☐ Enstitü / Fakülte Yönetim Kurulunun gerekçeli kararı ile tezimin erişime açılması mezuniyet tarihimden itibaren … ay ertelenmiştir.[2]

☐ Tezimle ilgili gizlilik kararı verilmiştir.[3][4]

19 Ocak 2023

_____

Eren OKUR

[1] MADDE 6(1) Lisansüstü tezle ilgili patent başvurusu yapılması veya patent alma sürecinin devam etmesi durumunda, tez danışmanının önerisi ve enstitü anabilim dalının uygun görüşü üzerine enstitü veya fakülte yönetim kurulu iki yıl süre ile tezin erişime açılmasının ertelenmesine karar verebilir.

[2] MADDE 6(2) Yeni teknik, materyal ve metotların kullanıldığı, henüz makaleye dönüşmemiş veya patent gibi yöntemlerle korunmamış ve internetten paylaşılması durumunda 3. şahıslara veya kurumlara haksız kazanç imkânı oluşturabilecek bilgi ve bulguları içeren tezler hakkında tez danışmanının önerisi ve enstitü anabilim dalının uygun görüşü üzerine enstitü veya fakülte yönetim kurulunun gerekçeli kararı ile altı ayı aşmamak üzere tezin erişime açılması engellenebilir.

[3] MADDE 7(1) Ulusal çıkarları veya güvenliği ilgilendiren, emniyet, istihbarat, savunma ve güvenlik, sağlık vb. konulara ilişkin lisansüstü tezlerle ilgili gizlilik kararı, tezin yapıldığı kurum tarafından verilir. Kurum ve kuruluşlarla yapılan iş birliği protokolü çerçevesinde hazırlanan lisansüstü tezlere ilişkin gizlilik kararı ise, ilgili kurum ve kuruluşun önerisi ile enstitü veya fakültenin uygun görüşü üzerine üniversite yönetim kurulu tarafından verilir. Gizlilik kararı verilen tezler Yükseköğretim Kuruluna bildirilir.

[4] MADDE 7(2) Gizlilik kararı verilen tezler gizlilik süresince enstitü veya fakülte tarafından gizlilik kuralları çerçevesinde muhafaza edilir, gizlilik kararının kaldırılması halinde Tez Otomasyon Sistemine yüklenir.

ETİK BEYAN

KTO Karatay Üniversitesi Lisansüstü Eğitim Enstitüsü Tez Hazırlama ve Yazım Kurallarına uygun olarak Prof. Dr. Ali Bülent UŞAKLI danışmanlığında tarafımdan üretilen bu tez çalışmasında; sunduğum tüm veri, enformasyon, bilgi ve belgeleri bilimsel etik kuralları çerçevesinde elde ettiğimi, tüm değerlendirme, analiz, bulgu ve sonuçları bilimsel usullere uygun olarak sunduğumu, tez çalışmasında yararlandığım kaynakların tümüne bilimsel normlara uygun biçimde atıfta bulunarak kaynak gösterdiğimi, tezimin/projemin kaynak gösterilen durumlar dışında özgün olduğunu bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

19 Ocak 2023

_____

Eren OKUR

Aileme.

TEŞEKKÜR

Tez çalışmam boyunca akademik bilgisi ve donanımı ile bana yol gösteren, her problemde benim yanımda olan destekleyen, motive eden ve tecrübelerini paylaşan kıymetli hocam ve danışmanım Prof. Dr. Ali Bülent UŞAKLI'ya,

Maddi manevi desteklerinin eksikliklerini hissetmediğim annem ve babama teşekkür ederim.

19 Ocak 2023 Eren OKUR

ÖZET

Eren OKUR

Mobil Robotlarda Genişletilmiş Kalman Filtresi Kullanılarak Kapalı Alan Haritalandırma ve Navigasyonda İyileştirme

Yüksek Lisans Tezi

Konya, 2023

Bu tez çalışmasında mobil robot kullanılarak etkin şekilde iç alan haritalandırması ve navigasyon yapılması için Genişletilmiş Kalman Filtreleri (EKF) kullanılmıştır. Günümüzde mobil robotlarda navigasyon işlemleri temizlik, sipariş hazırlanması, yol tarifi gibi amaçlarla kullanılmaktadır. İç alan haritalandırma ve navigasyon sistemlerinin gün geçtikçe gelişmesi sonucunda mobil robotların daha yaygın olarak kullanılacağı öngörülmektedir. Tez kapsamında robotun üzerinde bulunan bilgisayar kendi üzerindeki tüm bilgileri alıp merkezi bir bilgisayara aktarmaktadır. Bu durum birden fazla robota ait verilerin işlenmesine olanak sağlamaktadır. Merkezi bilgisayar robottan gelen verileri EKF ile işleyerek; zaman maliyeti açısından navigasyonda ve artan doğruluk bakımından haritalamayı daha etkin bir şekilde gerçekleştirmiştir. Merkezi bilgisayarda hesaplanan verilere göre robota hareket emirleri gönderilir. Bu verilere göre robot tekerlekleri, mobil robot üzerindeki mikrobilgisayar tarafından işlenerek robot hareketi sağlanır. Tez çalışmasında iç alan haritalandırması ve navigasyon için EKF'nin yön bulma ve konumlandırmada mobil robota sağladığı performans etkisinin tespiti amaçlanmıştır. Bununla beraber mobil robottaki enerji tüketen hesaplama işlemlerinin merkezi bir bilgisayarda yapılması ile enerji verimliliği sağlanması amaçlanmıştır.

Tez çalışması için yapılan mobil robotta EKF kullandığı zaman haritalandırma işlemlerinin doğruluğunda %53,8 iyileşme sağlandığı, navigasyon yapıldığı zaman ise işlem süresinde %30,5 iyileşme sağlandığı gözlemlenmiştir. Tüm işlemler merkezi bilgisayar üzerinde yapıldığında robotun işlem hızında ve maliyetinde verimlilik artışı gözlemlenmiştir. İşlemler merkezi bilgisayarda yapıldığında robotun enerjisini daha verimli kullanması sağlanarak hareketli çalışma süresi %30 arttırılmıştır.

Anahtar Kelimeler

İç Alan Haritalandırması, İç Alan Navigasyon, Genişletilmiş Kalman Filtre, Mobil Robotlar

ABSTRACT

Eren OKUR

Mobile Robot Indoor Navigation and Mapping Performance Improvement with Extended Kalman Filter

Master's Thesis

Konya, 2023

In this thesis, Extended Kalman Filters (EKF) has been used to make efficient indoor navigation and mapping with mobile robots. Today, mobile robots have been used to navigation, cleaning, preparing order and to find path. Improving indoor navigation and mapping, mobile robots will become popular in last years. The microcomputer on the robot reads all the sensor data and sends them to the main server. This situation lets the main server get many robots information. The main server uses all the data with a Kalman filter, then makes all the calculations work for indoor navigation and mapping. This process saves time for navigation and gives more precise map information. After calculations are done on the main server movement, orders are sent to the mobile robot for execution. With the given wheel orders, the mobile robot computer processes all the data and makes the necessary movements. In this thesis, EKF's effect on indoor mapping and navigation performance are researched. With the usage of the main server for computation, robots' energy efficiency is increased.

The mobile robot developed for this thesis, EKF provided 53.8% better efficiency for indoor mapping. Navigation with EKF provided 30.5% better efficiency. When all the work is done with the main server robot, process time increased and the project budget for the mobile robot is reduced. With this method, a mobile robot can use its energy more efficiently and movement time has been increased by 30%.

Keywords

Indoor mapping, Indoor navigation, Extended Kalman Filter, Mobil Robots

İÇİNDEKİLER

TABLOLAR DİZİNİ

ŞEKİLLER DİZİNİ

SİMGELER DİZİNİ

Simge	Açıklama
X(t)	Kalman durum verisi vektörü
F(t)	Dinamik matris değerleri
H(t)	Kalman tasarım matrisi
V(t)	Sensör gürültüsü
z(t)	Ölçüm vektörü
G(t)	Şekillendirme matrisi
W(t)	Sistem gürültüsü
ẋ(t)	Sensör durum verileri zamana göre türevi
u_{k+1}	Değişim vektörü
K_{k+1}	Kalman kazancı
x_{k+1}^{+}	Kalman durum tahmini
P_{k+1}^{-}	Kalman önceki tahmin
H_{k+1}	Tasarım matrisi
P_{k+1}^{+}	Kalman güncel tahmini
P_k	Kovaryans matrisi
P_k^{+}	Sensörden gelen anlık veri
ϕ_{k+1}	Zamana göre oluşan durum verileri
ϕ_{k+1}^{T}	Zamana göre oluşan durum verilerinin güncellenmesi
Q_k	Kovasyans matrisi için gürültü
R_{k+1}	Gürültü kovaryansı
t	Zaman
v(t)	Lineer hız
w(t)	Açısal hız
Q(t)	Orijin konum
x(t)	Orijin konum X
y(t)	Orijin konum Y
S(p)	Jakobian matrisi
$v_k(t)$	Açısal ve doğrusal hız matrisi
ΔS(p)	Yapısal değişimler ve modellenmiş dinamikler
d(t)	Dalgalanma değeri
ṗ(t)	Birinci dereceden kinematik vektörü

p(t)	Robotun birim zamandaki konumu
$e_x(t)$	Robotun t zamanda x konumuna göre hatası
$e_y(t)$	Robotun t zamanda y konumuna göre hatası
$e_\theta(t)$	Robotun t zamanda θ açısına göre hatası
w_m1	Motor 1 için PWM değeri
w_m2	Motor 2 için PWM değeri
speed_lin	Motor lineer hızı
wheel_rad	Motor teker çapı
speed_ang	Motor açısal hızı
wheel_sep	Motorla teker arası uzaklık
E_{est}	Hesaplama hatası tahmini
E_{mea}	Hesaplama hatası
KG	Kalman kazancı
mea	Sensör hesap değeri
$\overline{p_k}$	Tahmin koveryansı
H_k^T	Tasarım matrisi
R_K	Gözlem koveryansı
f(a)	Taylor serisi lineer girdi verileri

KISALTMALAR DİZİNİ

Kısaltma	Açıklama
SLAM	İç Alan Haritalandırma
ROS	Robotik İşletim Sistemi
LIDAR	Lazerli Mesafe Sensörü
IMU	Atalet Ölçü Birimi
EKF	Genişletilmiş Kalman Filtresi
NASA	Amerikan Uzay Ajansı
HP	Yüksek Güç Motor
DC	Doğru Akım
PWM	Darbe Genişliği Modülasyonu
V	Volt
A	Amper
MOSFET	Metal Oksit Yarı İletken Alan Etkili Transistör
USB	Evrensel Seri Veri Yolu
UART	Evrensel Asenkron Alıcı-Verici
BPS	Saniyedeki Bit Sayısı
DOF	Serbestlik Derecesi
PPR	Her Dönüşteki Atış
CPR	Her Dönüşteki Sayım
Li-Po	Lityum Polimer
GUI	Görsel Kullanıcı Programı
RPM	Dakikadaki Devir Sayısı
LDPE	Düşük Yoğunluklu Polietilen
ROS	Robot İşletim Sistemi
2D	2 Boyutlu
RAM	Rastgele Erişilebilir Bellek
AMCL	Adapte Olabilen Monte Carlo Konumlandırması
GPIO	Genel Amaçlı Giriş Çıkış
I2C	Tümleşik Devreler Arası Haberleşme Protokolü
SDA	Seri Veri
SCL	Seri Saat
GND	Toprak

GPS	Global Konumlandırma Sistemi
ECU	Elektronik Kontrol Birimi
LRF	Robot Konumlandırma Filtresi
m	Metre
mm	Milimetre
cm	Santimetre
Hz	Hertz
KHz	Kilo Hertz
C	Santigrat
m^2	Metrekare

1. GİRİŞ

İnsanoğlu hayatını kolaylaştırmak için teknik buluşlar yapmış ve bunları sürekli geliştirmiştir. Teknolojinin gelişmesi ile işlerin daha hızlı yapılması için gelişmiş araç ve makineler üretilmiştir. Elektronik destekli bu makineler zor, tehlikeli veya tekrar gerektiren işleri insan emeği olmadan hızlı ve doğru yapabilmektedir. Makinelerin işlevlerinin artması, kontrol, algılama sistemlerinin gelişmesi ve bilgisayar algoritmalarının üretilmesi ile robotik ortaya çıkmıştır. (Aksoy, 2020)

Robot kelimesi temel olarak Çekçe'de "robota" yani zorunlu çalışan kelimesinden türemiştir. Robotlar elektromekanik sistemlerdir. Aynı zamanda mekatroniğin en önemli ürünlerinden biridir. Robotlar, yapay zekâ ve mekanik sistemleri ile insan hareketlerinin bir kısmını ya da hepsini yapmayı amaçlarlar. Günümüzde endüstriyel çalışmalarda yaygın olarak kullanılmaktadırlar. İnsan tekrarlı işleri yaparken hata yapabilir ancak robot ise bu tekrarlı işlemleri daha hızlı, hatasız, güvenli ve anlık izlenebilir şekilde yapabilir. Bakımları zamanında ve doğru yapılan robotlar uzun süre sorunsuz çalışabilmektedirler. (Skalfist, 2007)

Robotlar iş güvenliğini sağlamak amacıyla insanların zarar görebileceği işlerde özellikle tercih edilmektedir. İnsanlara zarar verebilecek malzemeler üzerinde çok daha güvenli şekilde çalışabilir, yüklenen programla en az hatayla çok hızlı şekilde yapabilirler. Böylece endüstride robotların kullanılmasıyla en az hatayla, maliyetler düşürülecek, kazalar önlenebilecek, yapılan işlemler hızlandırılacak ve daha güvenli bir çalışma imkânı sağlanacaktır. (Okumuş, 2019)

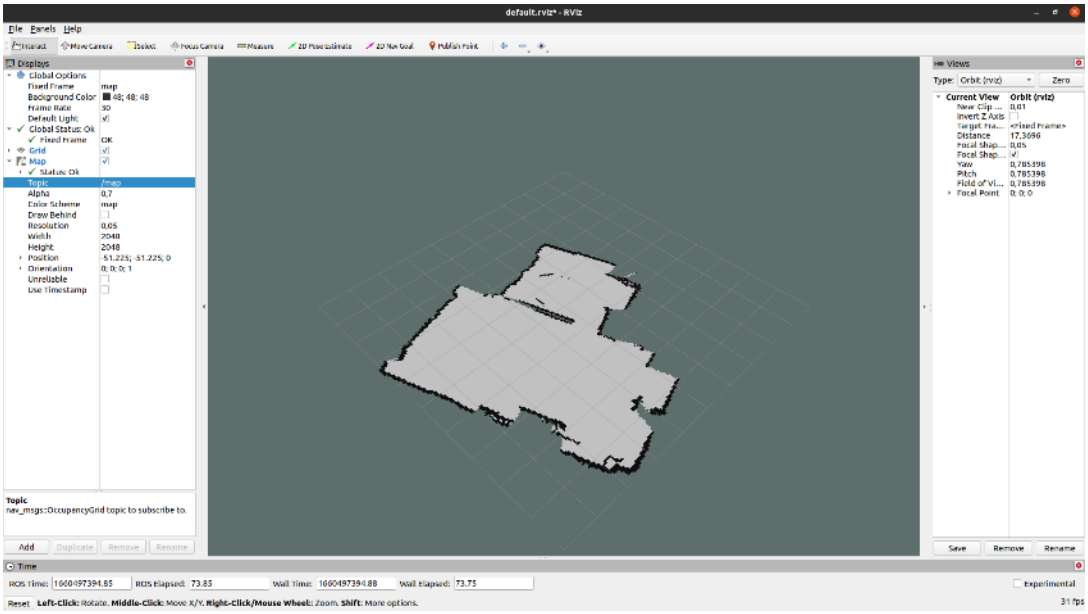
Robotların bazıları bulunduğu noktada sabit olarak çalışıp fiziki ve işlevsel olarak insanlara benzememektedir. Bazı robotlar ise insana benzemekte ve tepkileriyle de insanı andırmaktadır. Bu tip insana fiziki olarak benzeyen robotlara *insansı robot* ismi verilmektedir. Bazı robotlar ayakları veya tekerlekleri yardımıyla hareket edebilmektedir. Hareketli robotlar her ortamda otonom ya da kullanıcı kontrolü ile bir noktadan diğer noktaya gidebilmektedir. Bu şekilde hareket eden robotlara *mobil robot* denilmektedir. Mobil robotlar tasarımlarına göre insansı olarak nitelendirilebileceği gibi insansı özellik taşımayıp işlevsel özellikleri ile insanların yaptığı gezinme işlemini yapabilirler. Bir robotun bulunduğu noktadan başka bir noktaya hareket etmeye başlaması, mobil robot olmasının en temel koşuludur. (Afanasyev, 2015)

Şekil 1. Mobil Robot Örnekleri

(Kaynak: Akınrobotics, 2021)

Mobil robotlar endüstride oldukça yaygın kullanılırken artık evlerimize de girmiş, birçok evin temizliği onlara yaptırılmaktadır. Bu tez çalışmasında geliştirilen mobil robot; iç alanda otonom şekilde hareket edebilen bir robottur. Bununla birlikte mobil robotlar iç alanda oldukları kadar dış alanlarda, hatta uzayda da görevlendirilmektedir. Robotun otonom olarak bir noktadan başka bir noktaya gitmesi onu kontrol eden insana olan gereksinimini azaltmaktadır. Endüstriyel bir depoda robot bir raftan başka bir rafa ürün taşıyabilmekte, uzayda bilim insanlarının test yapabilmeleri için örneklerin bulunduğu noktalara gidebilmektedir. Günümüzde bizler normal hayatımıza devam ederken bir taraftan mobil robotlar otonom şekilde evlerimizi temizlemekte ve işlerimizi kolaylaştırmaktadır. (Li, 2018)

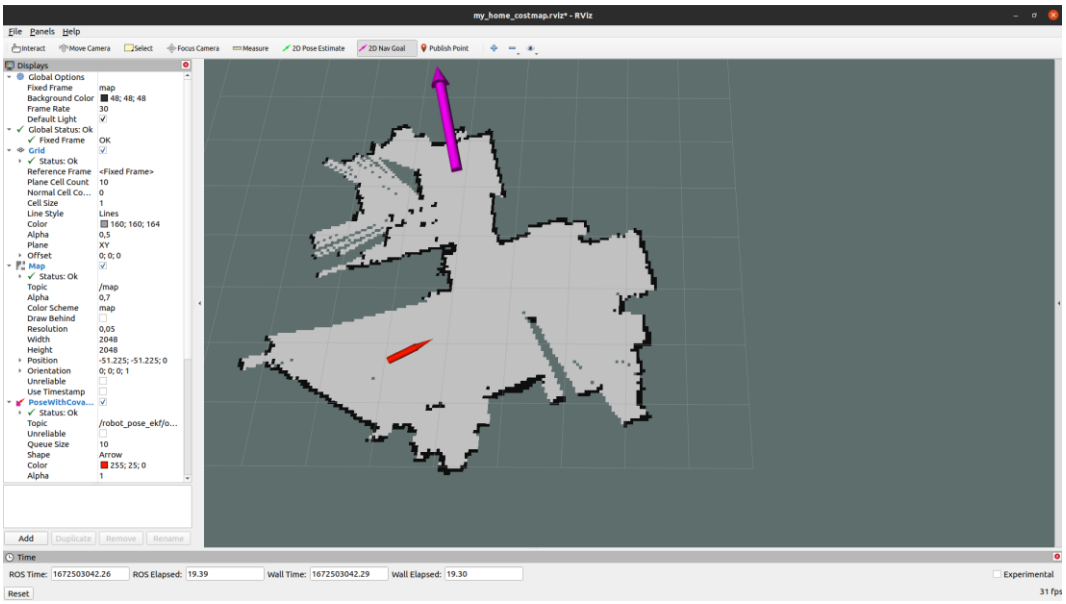
Mobil robotlar otonom hareket edebilme yeteneklerini kullanarak bulundukları iç alanlarda bir noktadan başka bir noktaya giderek işlerini yapabilmektedirler. Mobil robotların bilmedikleri bir ortamda otonom hareket edebilmesi için öncelikle bulundukları ortamı haritalandırmaları gerekmektedir. Bu işlem anlık konumlandırma ve haritalama anlamına gelen SLAM (Simultaneous Localization and Mapping) sayesinde robot tarafından yapılabilmektedir. (Karam, 2020)



Şekil 2. Gerçekleştirilen İç Alan Haritası Örneği

SLAM çalışmalarında, mobil robotun üzerinde bulunduğu ortamı anlamlandırması için ihtiyaca göre sensörlerin bulunması gerekmektedir. Tez çalışması kapsamında LIDAR (Light Detection and Ranging) ve IMU (Inertial Measurement Unit) sürekli olarak etrafı taramakta, robotun çevresinde bulunan ortamdan veri almasını sağlamaktadır. Günümüzde haritalandırma işlemi yapılması için mobil robota insanlarda olduğu gibi alanın tanıtılması gerekmektedir. Bu işlem için mobil robot bir insan yardımı ile bulunduğu ortamda gezer. Bu sırada insan robota girilmemesi gereken yerleri, bulunduğu bölgenin adını, özelliklerini mobil robota tanımlar. (Karam, 2020)

Robot harita çıkardıktan sonra çevresini bildiği için bu alanda hareket ederek bir noktadan başka bir noktaya gidebilmektedir. Robotun bu şekilde iki nokta arasındaki hareketine de *navigasyon* adı verilmektedir.



Şekil 3. Gerçekleştirilen Navigasyon Çalışması Örneği

Mobil robotun SLAM ve navigasyon işlemlerini en etkin şekilde yapabilmesi için robotun kendi üzerinde bulundurduğu sensör verilerini mümkün olduğunca etkin şekilde kullanması gerekmektedir. Mobil robotta kullanılan lazer verileri, teker dönüş sayım bilgileri, ivme ve yön bilgileri sürekli olarak değişmektedir. Bu veriler, robotun işlemlerini doğru yapabilmesi için önemlidir.

Lazer verileri bir eksen üzerinde 360º ölçüm yapabilen LIDAR ile ölçülür. İç alan haritalandırması yapmak için LIDAR 6 - 10 Hz arasında 360º ölçüm yapmalıdır. Bu ölçümleri yapmak için LIDAR üzerinde bir adet lezer alıcı ve verici bulunur. LIDAR ilk olarak bir sinyal gönderir ve belirli bir süre sonra tekrar aynı sinyalin yansımasını alır. Lazer ışığının gönderilmesi ve alınması arasında geçen zaman ile LIDAR'ın yansıma yüzeyine olan mesafesi bulunur. Mobil robotun çevresindeki nesnelerle olan uzaklıkları bu şekilde belirlenir. (Karam, 2020)

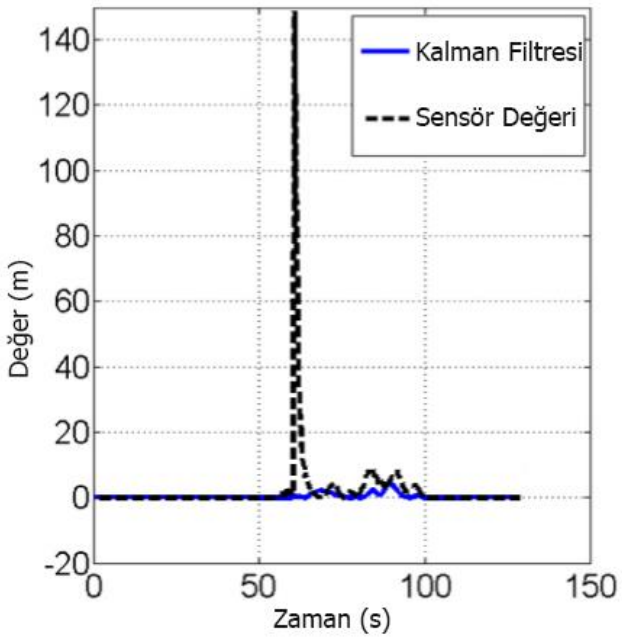
Şekil 4. Mobil Robotun Üzerinde LIDAR Yerleşimi

Robotun motorlarında bulunan ve teker dönüş hareketinin tespit edilmesine yarayan sensöre *enkoder* denilmektedir. Enkoderler mekaniksel hareketi dijital (sayısal) ya da analog sinyale dönüştürebilirler. Robot üzerindeki motorlarda bulunan enkoder motorun dönen ya da hareket eden yüzeylerinde belirli aralıklarda bulunan çentik, delik gibi belirli fiziksel işaretleri takip ederek her uyarı geldiğinde bir sinyal üretir. Mobil robotlarda bulunan enkoder, robotun fiziksel hareketi sinyallere dönüştürerek robotun aldığı mesafeyi ölçmeyi sağlar. (Shen, 2018)

LIDAR ve enkoder sensörleri mobil robotlarda SLAM ve navigasyon işlemlerini yapmak için yeterlidir. Bu sensörlere ilave olarak eklenen 3D kameralar ve GPS (Global Positioning System) modülü; kullanılan algoritmalar yardımı ile haritanın ve

navigasyonun kalitesini arttırmaktadır. Başka bir ifadeyle LIDAR ve enkoder sensörlerine ek olarak derinlik kameraları, radar sistemleri, ivme ölçerler vb. sistemler de kullanılabilmektedir. Mobil robot, tüm sensör verilerini değerlendiren SLAM yazılım algoritmalarını kullanarak gerçek zamanlı haritalandırma ve konumlandırma işlemini yapar. (Chen, 2018)

Sensör verileri sensörün kendi üzerinde, kablolarında ya da bağlı olduğu bilgisayarda kullanılan işletim sistemindeki sorunlar nedeniyle hatalı gelebilmektedir. Sistemde bulunan gürültü ya da geliştirilen yazılımdan kaynaklı olarak da hatalı bilgi oluşabilmektedir. Bu hatalı verilerin haritalama ve navigasyon işlemlerinde kullanılmaması gerekmektedir. Bu hatalı işlemlerin önlenmesi için filtreleme yöntemleri geliştirilmiştir. Sensör filtreleri önceki sensör verileri ile anlık ölçülen sensör verilerini karşılaştırmaktadır. Olmaması gereken bir sensör verisi sisteme girerse filtre bu veriyi sistemin daha önceki verilere yakınlaştırarak tolere eder. Bu tolere işlemi hatalı verinin tüm sisteme zarar vermesini engelleyecektir. (Bader, 2017) Şekil 5'de gösterilen lineer veriler, anlık olarak hatalı gelen mesafe verisini robot üzerinde kullanılan filtrelerle düzeltilmektedir.



Şekil 5. Kalman Filtresinin Sensör Hatasını Düzeltmesi

(Kaynak: Bader, 2017)

Birden çok sensörün olduğu durumda ise bu filtreler hem geçmiş veriye hem de diğer sensörlerden gelen değerlere bakmaktadır. Sistemde bulunan sensörlerin birinde oluşacak bir sorunda kullanılan filtre hem geçmiş verileri hem de diğer sensörlerin verilerini değerlendirerek sisteme doğru verinin gelmesini sağlamaktadır. Bu filtrelerin en önemlisi Radolf E. Kalman tarafından 1960'lı yıllarda geliştirilen Kalman filtreleridir. (Sabatini, 2013)

Kalman filtresi, ilk aşamada sensörden gelen veriler ile bir durum verisi vektörü X(t) elde eder. Durum verisi vektörü sensörün daha önceki verilerinden elde edilen tasarım matrisi H(t) ile çarpılıp sensör gürültüsü V(t) verisine eklenir. Bu işlemler sonucunda hesaplanan ölçüm vektörü z(t) elde edilir. İşlem yapılırken sensörden gelen veriler yerine elde edilen z(t) değeri kullanılması hata ihtimalini tek sensör için düşürmektedir. Kalman filterelemesi için yapılan işlemler Denklem 1 ile ifade edilmektedir. (Sabatini, 2013)

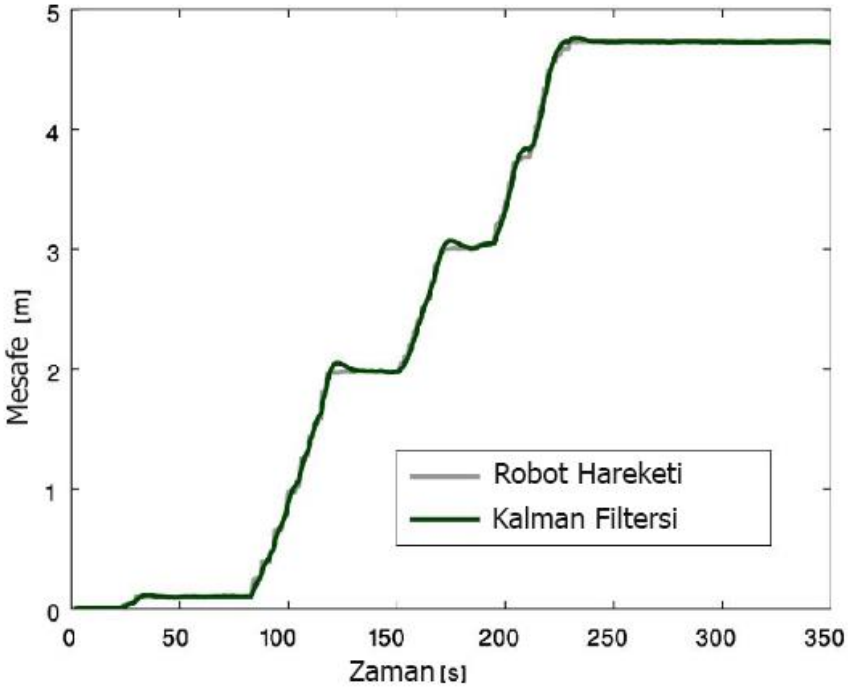
$$z(t) = H(t) * X(t) + V(t) \quad (1)$$

Kalman filterleri sistemin tamamınına uygulandığı zaman sensör durum verileri x(t) öncelikle oluşturulan dinamik matris değerleri F(t) ile çarpılır. Daha sonra veriler şekillendirme matrisi G(t) ile sistem gürültüsü W(t) çarpım değeri ile toplanarak $\dot{x}(t)$ sensör durum verilerinin zamana göre türevi elde edilir. Bu işlem Denklem 2'de gösterilmiştir. (Sabatini, 2013)

$$\dot{x}(t) = F(t) * x(t) + G(t) * W(t) \quad (2)$$

Bu Kalman filtre formülleri lineer veri ile çalışan sistemlerde etkin çalışmaktadır. Bunun nedeni gelen veriyi önceki verilerle karşılaştırmanın bu sistemlerde etkili olmasıdır. Eğer lineer olmayan sistemlerde Kalman filtre kullanılırsa sistem oluşacak değişimleri önceki verilerle karşılaştıracak ve doğru sonuçlar bulamayacaktır. Bu durum robot ilerleme grafiğinin gösterildiği Şekil 6'da gösterilmektedir. Şekil 6'da görüleceği gibi robot sabit şekilde hareket ederken anlık hız değişimlerinden kaynaklanan dalgalanmaları kaldırmıştır. Buna rağmen Kalman filtresi, robot durup beklediği zaman robotu hâlâ

ilerliyormuş gibi göstermektedir. Bu hata oluştuktan sonra sistem zamanla robotun gerçekten durduğunu anlamakta ve robotun doğru konuma gelmesini sağlamaktadır. (Raina, 2007)



Şekil 6. Robot İlerleme Hareketinin Kalman Fitreleri Takibi

(Kaynak: Raina, 2007)

Lineer olmayan sistemlerde Kalman filtrelerinin doğru çalışmaması sorununu çözmek için Genişletilmiş Kalman filtreleri (EKF) geliştirilmiştir. EKF temel olarak lineer olmayan sistemleri lineer yapıp Kalman filresinden geçirmeyi hedefleyen bir yaklaşımdır. Bu işlemi yapmak için sistemin kovaryans ve anlık ortalama değişim değerlerinden yararlanır. EKF iki temel algoritmadan oluşmaktadır. Bu aşamalardan ilki güncelleme ikincisi ise tahmin etme algoritmasıdır. (Raina, 2007)

İlk aşamada sensör durum tahminleri güncellenmektedir. Değişim vektörü u_{k+1} olarak verildiği ve kalman kazancı K_{k+1} olarak verildiğinde bu iki değerin çarpımı ile durum tahmini x_{k+1}^{+} değeri bulunur. Bu durum Denklem 3 ile gösterilmiştir. (Raina, 2007)

$$x_{k+1}^{+} = u_{k+1} * K_{k+1} \quad (3)$$

Güncelleme işleminin son aşamasında tahmin değerleri güncellenmesi amaçlanır. Bu işlem için daha önce tahmin edilen P_{k+1}^{-} değeri ile kalman kazancı K_{k+1} ve tasarım matrisi H_{k+1} değeri ile çarpılır. Elde edilen bu değer tahmin edilen P_{k+1}^{-} değerinden çıkarılarak güncel değer P_{k+1}^{+} bulunur. Yapılan işlemler Denklem 4'te gösterilmiştir. (Raina, 2007)

$$P_{k+1}^{+} = P_{k+1}^{-} - K_{k+1} * H_{k+1} * P_{k+1}^{-} \quad (4)$$

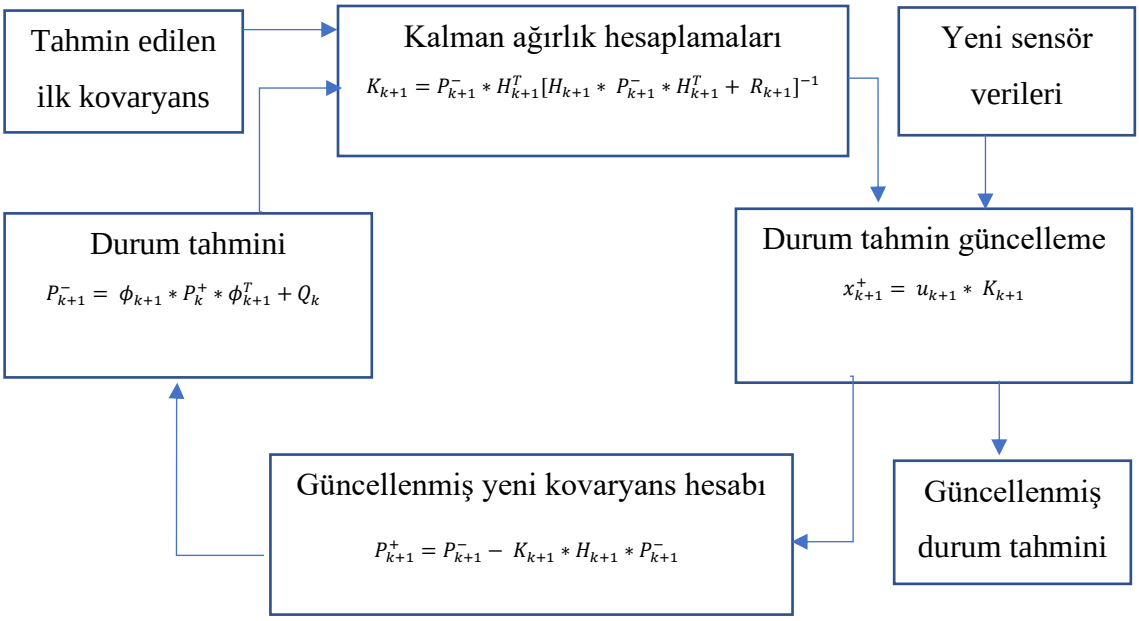
Tahmin algoritması tahmin edilen P_{k+1}^{-} değerini durum vektörünü tahmin ederek kovaryans matrisi P_k değerine göre bulmayı amaçlamaktadır. Sürekli gelen değerler ile oluşturulan P_k^{+} değeri güncellenmektedir. Kavaryans matrisi hesaplanırken şimdiki zaman ile süreç (proses) modelinden oluşan zaman modellerini kullanarak hesaplamalar yapılmaktadır. Bu hesaplamalarda güncel değerler P_k^{+} zamana göre oluşan durum verileri ϕ_{k+1} ve bu verilerin güncellenmiş ϕ_{k+1}^{T}değeri ile çarpılır. Bu işlemlerden elde edilen sonuçlar $Q_k$belirli bir zamanda koveryans matris tarafından belirlenen gürültü ile toplanır. Bu hesaplamalar Denklem 5'te gösterilmektedir. (Raina, 2007)

$$P_{k+1}^{-} = \phi_{k+1} * P_k^{+} * \phi_{k+1}^{T} + Q_k \quad (5)$$

Tahmin uygulmasının son aşamasında Kalman ağırlığı tekrar hesaplanmaktadır. Bu hesaplama için tahmin edilen kovaryasyon verisi ve gürültü kovaryans R_{k+1} değerleri kullanılmaktadır. Denklem 6'da hesaplama için gerekli işlemler gösterilmiştir. (Raina, 2007)

$$K_{k+1} = P_{k+1}^{-} * H_{k+1}^{T}[H_{k+1} * P_{k+1}^{-} * H_{k+1}^{T} + R_{k+1}]^{-1} \quad (6)$$

EKF sisteminde bulunan günelleme ve tahmin aşamaları toplamda dört algoritma ile çalışmaktadır. Bu aşamalar Şekil 7'de gösterilmiştir.



Şekil 7. EKF Kalman Çalışma Döngüsü

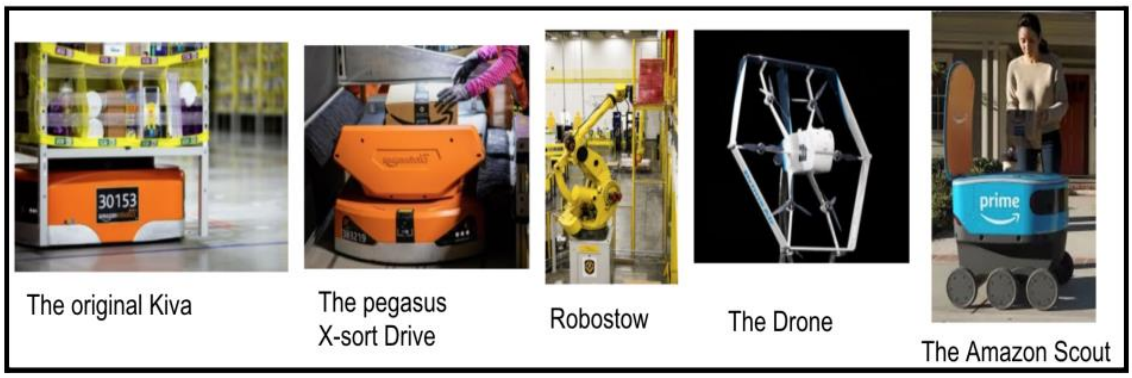
(Kaynak: Raina, 2007)

Mobil robotlarda kullanılan Kalman filtresi, robotun sensörlerinden gelen verileri kontrol ederek hatalı ölçüm ve değerlendirmelerin oluşmasını engellemektedir. Bu filtrenin kullanıldığı sistemde sensör verileri sürekli okunup, doğrulama işlemi yapılmaktadır. Bu düzeltilmiş sensör verileri robotun hafızasına işlenerek gerekli haritalandırma ve navigasyon işlemleri yapılmaktadır. (Koster, 2015)

Mobil robotların bulundukları ortamları haritalandırmaları ve otonom olarak alan içinde hareket edebilmeleri önemli bir gelişmedir. Bu durum robot teknolojisinde çığır açmıştır. Robotlar bu sayede insanların müdahalesi olmadan kendilerine verilen işleri farklı noktalarda yapabilmektedir. Günümüzde otonom mobil robot teknolojileri endüstride kritik bir konuma gelmiştir. (Koster, 2015) Amazon, Ford, Boston Dynamics, NASA, Akınrobotik gibi bir çok şirket bu alanda büyük yatırımlar yapmaktadır. İngiltere'de geliştirilen Starship robotu kaldırımlarda ilerleyip paket dağıtmakta, 20 kg'lık yükleri 5 km uzaktaki hedeflere 30 dakika gibi bir sürede taşıyabilmektedir. Starlink robotlar için

geliştirici firma 2018 yılında 17,2 milyon dolar bütçe ayırmıştır. Amazon da robotlar için büyük yatırım yapan firmalardan biridir. Firma yaptığı yatırımlarla 2018 yılında operasyonel maliyetleri 22 milyon USD kısarak %20 oranında küçülmeyi başarmıştır. Amazon'daki bir depoda robotik sistem kurma maliyetinin 2018 yılında yaklaşık 15 ile 20 milyon USD civarında olduğu tespit edilmiştir. (Wozniakowski, 2018)

Depolarda mobil robotik uygulamalar için 2021 yılı içinde 4,6 milyar USD robotik bütçe ayrılmıştır. Sektörde yerleşen bir çok firma farklı uygulamalar için çok sayıda farklı robot geliştirmektedir. (Girija, 2021) Şekil 8'de Amazon firmasının 2021 yılı itibarıyla yaptığı robotlar gösterilmiştir.

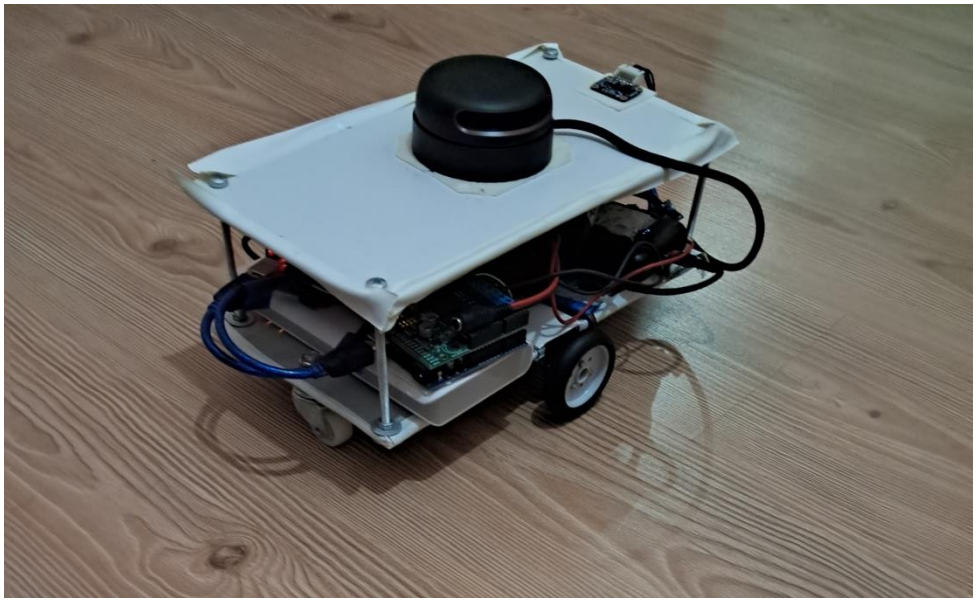


Şekil 8. Örnek Mobil ve Endüstriyel Robotlar

(Kaynak: Girija, 2021)

Tez kapsamında yapılan mobil robot otonom hareket için gerekli olan LIDAR, enkoder, IMU gibi sensörlerle donatılmıştır. Robot üzerinde bir bilgisayar ve hareketini sağlamak için iki motor bulunmaktadır. Robotun enerji ihtiyacını sağlamak için bir adet 3 A , bir adet 2,1 A güç bankası ve bir adet 12 A lityumion batarya kullanılmıştır.

Robotun sensör verileri yine robot üzerindeki bir bilgisayar tarafından gömülü sistemlerden okunur. Bu veriler uzaktaki işlemci gücü daha yüksek bir sunucuya gönderilmektedir. Bu güçlü bilgisayarda bir test ortamı oluşturulmuştur. Robot tüm verileri kendisi işleyebileceği gibi tüm verilerini uzak bilgisayara da gönderebilmektedir. Ayrıca robotun harita çıkarması ve navigasyon yapması için gereken algoritmalar hem EFK kullanılarak hem de EFK kullanılmadan çalışacak şekilde hazırlanmıştır. Şekil 9'da tez kapsamında gerçekleştirilen robot görülmektedir.



Şekil 9. Tez Kapsamında Yapılan Mobil Robot

Tez kapsamında geliştirilen mobil robot iç alan haritalandırması ve navigasyon işlemlerini yapmaktadır. Bu alt sistemlerin daha etkin şekilde çalışması için robotun uzaktaki sunucu olarak çalışan merkezi bilgisayar üzerinden yönetilmesi sağlanmıştır. İşlem kapasitesi yüksek uzak sunucu kullanımı sayesinde yapılan işlemler çok daha hızlı olmuştur. Bu sunucu sayesinde robotun bataryası daha uzun süre kullanılabilir hâle gelmiş yani enerji verimliliği sağlanmıştır. Sunucu işlemleri daha fazla kaynak ile yaptığı için anlık olarak daha doğru sonuçlara ulaşmıştır.

Robotta kullanılan Kalman filtresi ile sensörlerden okunan verilerin daha güvenilir olması sağlanmıştır. Bu filtre ile robot çalıştırıldığı zaman; navigasyon ve haritalama işlemlerinin doğruluğu artırmış ve hesaplama süreleri düşmüştür.

EKF ile mobil robotlarda navigasyon ve haritalama işlemleri farklı şekilde incelenmiştir. Daha önceki araştırmalarda bu tez çalışmasında yapılan işlemlerin karşılaştırması Tablo 1‘de verilmiştir.

Tablo 1. Tez Konusundaki Çalışmaların Karşılaştırılma Tablosu

Yazar	Yıl	Konu	Açıklama
Sasiadek	2008	Navigation Of An Autonomous Mobile Robot Using Ekf-Slam And Fastslam	Çalışmada konumlara erişmede EKF ile yapılan SLAM işlemlerinde büyük başarı gözlemlenmemiş. Tez çalışmasında doğruluk oranında % 0,7 başarı gözlemlenmiştir.
Luka	2010	EKF-Based Localization of a Wheeled Mobile Robot in Structured Environments	EKF ve EKF’sız sonuçlar birbirine yakın çıkmıştır. Tez çalışmasında EKF ile %30,5 iyileşme gözlemlenmiştir.
Yengin	2019	A Novel Data Association Technique To Improve Run-Time Efficiency Of Slam Algorithms	EKF Slam ile %43 ve %52 zaman kazancı sağlanmıştır. Tez çalışmasında zaman eşit olmuş harita alanlarında iyileşme sağlanmıştır.
Quoc	2019	Employing Extended Kalman Filter with Indoor Positioning System for Robot Localization Application	Robot konumlandırma işleminde X ve Y ye göre EKF daha etkin çıkmıştır. Tez çalışmasında navigasyon işlemine ek olarak haritalama

			işlemide yapımıştır. Sürece değil sonuca odaklanılmıştır.
Li	2015	Localization of Wheeled Mobile Robot Based on Extended Kalman Filtering	20 metrede ortalama 2 cm lik kazanç tespit edilmiştir. Tez çalışmasında 10 metrede %2 kazanç bulunmaktadır.
Yu	2019	Research on Accurate Positioning of Indoor Objects Based on ROS and 3D Point Cloud	Hata oranı % 6,7 olarak tespit edilmiştir. Tez çalışmasında hata payı %2 olarak bulunmuştur.
Xiang	2019	Localization and Mapping Algorithm for the Indoor Mobile Robot Based on LIDAR	Çalışmada da EKF'nın daha verimli çalıştığı doğrulanmıştır. Tez çalışmasında hata ve doğruluk oranları da paylaşılmıştır.
Tomoiaga	2016	Indoor Mapping Using Low Cost LIDAR Based Systems	Çalışmada yaklaşık 2 m fark bulunmuştur. Tez çalışmasında 0,6 m fark bulunmuştur.
Silva	2019	Mapping and Navigation for Indoor Robots under ROS: An Experimental Analysis	pioner_slam 2'de 0,9 ve 3'te 0.8'e fark tespit edilmiştir. Tez çalışmasında ortalama 0.6 m fark bulunmuştur.
Shen	2018	Research and Implementation of SLAM	Çalışmada hız ile ilgili testler yapılmıştır. SLAM verimleri birbirine yakın çıkmıştır.

		Based on LIDAR for Four-Wheeled Mobile Robot	Tez çalışmasında EKF ile yapılan SLAM % 0,7 iyileşme sağlanmıştır.
Pajazati	2014	SLAM – Map Building and Navigation via ROS	Çalışmada sadece similasyon yapılmıştır. Tez çalışmasında protatip mobil robotta yapılmıştır.
Omara	2015	Indoor Mapping using Kinect and ROS	Çalışmada ROS kullanılmış fakat EKF kullanılmamıştır. Tez çalışmasında server bağlantısı ve EKF kullanılmıştır.
Okumuş	2019	Cloud Based Indoor Navigation for ROS-enabled Automated Guided Vehicles	Çalışmada ROS ve server EKF olmadan kullanılmışıtır. Tez çalışmasında ekstradan EKF'de kullanılmıştır.
Megalingam	2018	ROS based Autonomous Indoor Navigation Simulation Using SLAM Algorithm	Çalışmada ROS ile Navigasyon ve SLAM yapılmıştır. Tez çalışmasında uzak bilgisayar ve EKF kullanılmamıştır. EKF ile yapılan testlerde ortalama 10 sn fark gözlemlenmiştir.

2. İÇ ALAN HARİTALANDIRMA VE NAVİGASYON

Mobil robotlar ilk defa bir ortama girdiği zaman bulundukları ortam hakkında bilgi sahibi olmazlar. Mobil robotun bu ortamda iş yapabilmesi için öncelikle gezebileceği alanı öğrenmesi gerekmektedir. Robot bulunduğu bu ortamı anlamlandırmak için üzerindeki sensörleri kullanarak 2 veya 3 boyutlu bir sanal düzlem oluşturur. Bu oluşturulan düzlem ilk aşamada robotun sensörlerinin tarayabildiği alan kadar sınırlıdır. Robot bulunduğu çevrede hareket etmeye başladığı zaman bu düzlem daha da şekillenmeye başlayarak bir haritaya dönüşmektedir. Robot bu şekilde ilk haritalandırma işlemini tamamlamış olacaktır.

Robot bulunduğu ortamın haritasını oluşturduktan sonra kendisini bu harita üzerinde konumlandırması gerekecektir. Konumlandırma aşamasında daha önceki harita ile anlık elde ettiği sensör verilerini kullanır. İlk olarak kendisini harita üzerinde bulunduğu noktaya yakın ya da üzerinde olan bir noktada konumlandırır. Bu ilk belirli konuma göre hareket ettikçe sürekli kendi konumunu belirlemeye devam eder. Anlık konumunu ilk referans noktasına göre takip etmek için üzerinde bulunan sensörleri kullanır.

Robot anlık konumunu belirleme işlemlerini yaptıktan sonra iki nokta arasında hareket eder. Bu işlem öncelikle bulunduğu nokta ile gitmesi istenilen nokta arasında en kısa yolun bulunması ile başlar. Robot bu yolu harita üzerindeki engelleri dikkate alarak planlar ve yol boyunca sürekli konumunu ve engelleri kontrol ederek istenilen noktaya ulaşır.

Elektronik konusunda öncelikle ele alınacak konu hareketli motorların ve sürücülerin özellikleri ile gömülü yazılımlardır. Bundan sonra robotta kullanılacak sensörler seçilmeli ve bu sensörlerden değerlerin okunabilmesi için senkronize gömülü programlar koşulmalıdır. Tüm bu cihazların güç tüketimleri hesaplanarak uygun batarya seçilmesi de elektroniğin en önemli konusudur. Robotik projelerinde enerji konusu önemlidir. Robottaki sensörler ve kullanılacak bilgisayarlar hem çok pahalı hem de hassas cihazlardır. Bu cihazlara uygun enerji verilmediği taktirde sistem çalışmamakta ya da sistemdeki cihazlar arızalanmaktadır.

Robot mekaniği özellikle dinamik parçaları olan robotlar için önemli bir konudur. Mobil robotlara takılan hareketli parçalar olduğu zaman robotun anlık konumuna ek olarak bu parçaların da uzaydaki konumlarının hesaplanması gerekir. Bu hareketli parçaların da

robota çarpmasını önleyecek bir tasarım yapılmalıdır. Mekanik tasarımda robotun tüm parçaları belirlenmeli ve bu parçaların montajı konusu üzerine durulmalıdır. Robotun hareketli parçaları için dinamik, durağan parçaları için de statik konumlandırma belirlenmelidir. Bu belirlenen bilgilerin yazılım ile odometri hesaplanması için paylaşılması gerekir.

Yazılım tüm mekanik ve elektronik parçaların nasıl çalışmasına karar veren yapıdır. Robot yazılımı tüm sensör bilgilerini okuyarak bazı kararlar alır. Bu kararlara uygun olarak tüm hareketli parçalara emirler verir. Robotun kontrolü de bu emirlerle yapılır. Robot harita çıkarmak için tüm sensörlerinden gelen verileri kullanarak SLAM algoritmalarını çalıştırır. Bu sırada robotun hareket etmesi gerekmektedir. Bunun sebebi sadece robotun bulunduğu yerdeki sensör bilgisinin robotun tüm çalışma alanının haritasının çıkarması için yeterli olmamasıdır. Aynı şekilde robot navigasyon ile bir noktadan başka bir noktaya giderken de sürekli konum ve yol hesabı yapmaya devam etmek durumundadır. Konumunu ve gideceği yolu hesapladıktan sonra motorlara gideceği yöne doğru ilerleme emri göndermektedir. Yazılım ile bu şekilde aktif hesaplama ve emir görevleri bulunurken aynı zamanda tüm sensörlerin okunması ve bu verilere uygulanacak filtrelemelerin uygulamaları yapılır.

2.1. Mobil Robot Elektronik Tasarımı

Mobil robotların iç alan haritalandırma ve navigasyon işlemlerini yapabilmeleri için üzerlerinde LIDAR, IMU, enkoder gibi sensörlerin bulunması zorunludur. Bu sensörler robotun üzerindeki kısıtlı güç ile kontrol edilebilmesi için oldukça kararlı güç kaynaklarına bağlanmalıdır. Ayrıca robotun ağırlığını taşıyarak hareket ettirebilecek uygun güçte motorlar kullanılmalıdır. Motorlar diğer bileşenlerle kıyaslandığında daha yüksek enerji kullanan ekipmanlardır. Motorları beslemek için uygun güç kaynağı bağlanmalıdır.

Motorların beslemeleri mümkün olduğunca elektronik kartlardan ayrılmalıdır. Motorlar bulundukları sistemler için büyük elektronik gürültü kaynaklarıdır. Ayrıca motorların enerji ihtiyaçları kullanıma göre değiştiği için çalışan sistemde enerji dalgalanmaları yaşanabilir. Motorların bu enerji ihtiyaçları doğrultusunda yüksek enerji sağlayan ve anlık değişimlerden etkilenmeyen kaynakların seçilmesi gerekmektedir. Bu kaynaklara elektronik sistemler zorunlu olmadıkça bağlanmamalı ve kablolar gereken akımı

sağlayacak şekilde seçilmelidir. Bu sistemlerde ısınma da olacağı göz önüne alınarak mümkün olduğunca güvenlik önlemleri alınmalıdır.

2.1.1. Motorlar ve Elektromekanik Sistemler

Motorlar robot üzerindeki gerilim ve ağırlığın büyük bir kısmını oluşturur. Çalışırken sistemde reaktif akım dolaşmasına ve elektronik sistemlerin zarar görmesine sebep olabilirler. Çalışma zamanlarında fazla zorlanırlarsa hem kendisinin hem de onu süren devrenin fazla ısınmasına sebep olabilirler. Bu gibi nedenlerden dolayı motor seçimi öncesinde ihtiyaçlar belirlenmeli ve bu ihtiyaçlara uygun motor seçimi esas olmalıdır. Robotik projelerde robotun ağırlığına göre yüksek performans ya da düşük performans motorlar seçilebilmektedir. Motor yüksek performanslı olursa robot üzerindeki bilgisayar, bataryalar, sensörler ve mekanik aksam gibi ağırlıkları daha rahat taşıyabilecektir. Fakat bu durum daha yüksek akım ihtiyacı doğuracak ve kuvvetli bir batarya seçimi gerektirecektir. Tüm bu koşullar dikkate alınarak tez çalışmasında geliştirilen mobil robot üzerinde 2 adet motor (Pololu 2273) kullanılmıştır (Şekil 10).

Şekil 10. Kullanılan DC Motor

(Kaynak: Pololu, 2022)

Motorlar 6 ila 12 Volt DC gerilim arasında çalışmaktadır ve yüksek güç tüketimine sahip, farklı gerilimler ve akım değerlerine ihtiyaç duyan ekipmanlardır. Gerilim yükseldikçe motorların kalkışı için gereken akım miktarı düşmektedir. Gerilim düştükçe, tersine motorun kalkması için gereken akım yükselmektedir. Yüksek güç motorları için gerekli motor gereksinimleri Tablo 2'de gösterilmiştir.

Tablo 2. Kullanılan Yüksek Güçlü DC Motorun Enerji – Güç Tablosu

Voltaj (V)	Kalkış Akımı (A)	Yüksüz Motor Hızı (RPM)	Kalkış Torku (oz-in)	Dişli Tipi
6	6,5	280	90	34:1
12	5,6	290	120	34:1

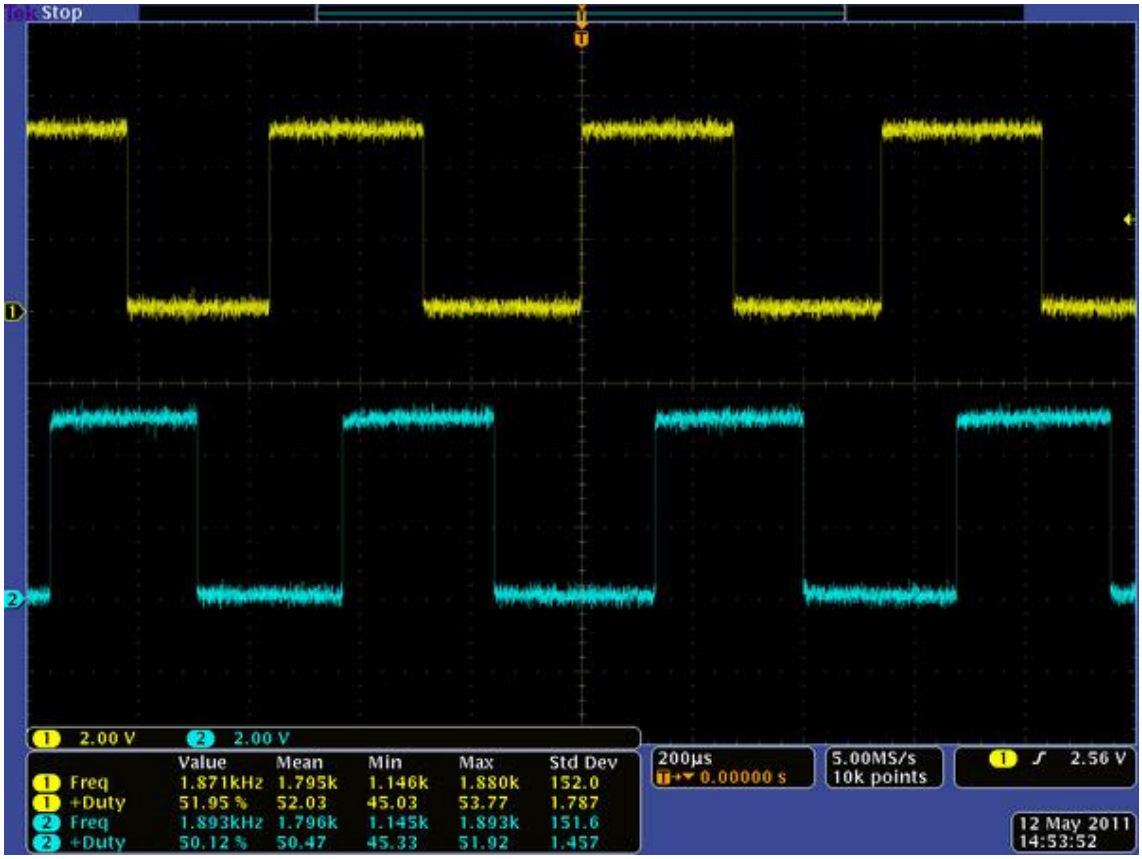
(Kaynak: Pololu, 2022)

Kullanılan motorların daha az akım çekmesi için 12V değerinde bataryalar seçilmiştir. Robotun iki motoru da hareket ederken 12 volt gerilim üzerinde bataryalardan 11,2 A akım geçmektedir. Bu yüksek akım gereksinimi nedeniyle doğru batarya seçimine dikkat edilmiştir. Motorun yüksek güçte olması daha hızlı ve torkla kalkmasına olanak vermektedir.

Motorların kullanımı sırasında motora bataryanın tüm gücü verilerek aniden hareket etmesi istenmez. Motorun yavaşça kalkarak istenen hızda gitmesi için motor sürücü sistemine ihtiyaç duyulmaktadır. Motor sürücüler darbe genişlik modülasyonu (PWM) olarak ifade edilen teknik ile motor için gerekli olan en uygun akımın istenilen hız için aktarılmasını sağlar. PWM kontrolü DC motorların endüvi içerisinde yaptığı hareketi kontrol için geliştirilmiştir.

PWM tekniği ile motora belirli aralıklarla açma kapama isteği gönderilir. Bu açma kapama frekansı motorun ne kadar hızlı ve kuvvetli döneceğini ayarlamak için kullanılır. Motora hızlı aç/kapa bilgisi gönderilirse motor sanki üzerinde sürekli gerilim varmış gibi tüm gücü ile dönebilecektir. Motora aksine sadece kalkabileceği frekansta sinyal verilirse motor kalkıp gidebileceği en düşük hızda ilerleyecektir. Motora yüksek frekansta sinyal verildiğinde motor aynı gerilimde daha az akım çekerek, kalkacak ama çalışma süresinde

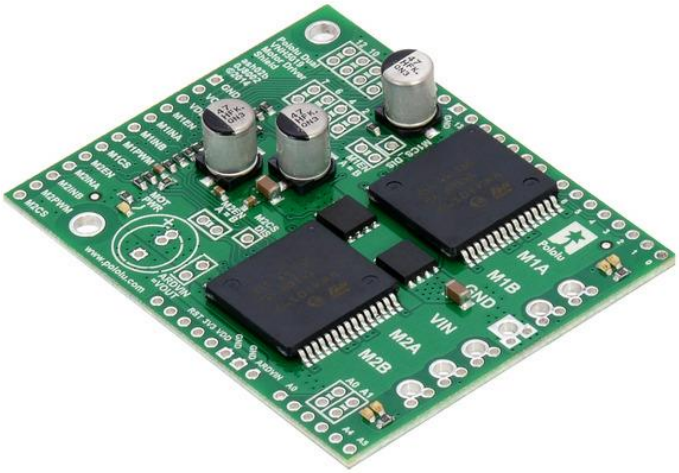
daha fazla akım çekecektir. Motora düşük frekansta sinyal verilirse motor aynı gerilimle yüksek akımda kalkacak fakat çalışma süresinde daha az akım çekecektir. Pololu 2273 motorun çalışması için gerekli akım grafiği Şekil 11'de belirtilmiştir.



Şekil 11. DC Motora Ait Enkoder Sinyalleri

(Kaynak: Pololu, 2022)

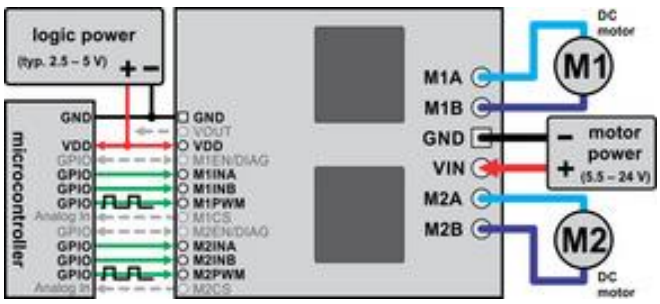
Motorlarda PWM kontrolleri motor sürücüler tarafından kontrol edilmektedir. Motor sürücüye motorun hızı verildiği zaman sürücü bu hız için gerekli PWM sinyallerini motora vermektedir. Robotta motor sürücü olarak her iki motorun da çekebileceği en yüksek akıma uygun olan sürücü seçilmesine önem verilmelidir. Motorlar 12V gerilimde 11,2A akıma kadar çıkabilmektedir. Motor sürücüsü aynı zamanda projede kullanılacak mikroişlemcilerle de etkin şekilde çalışabilecek durumda olmalıdır. Bu gereksinimleri sağlayabilmek için motor üreticisi Pololu'nun Arduino kartlar için ürettiği dual VNH5019 motor sürücü tercih edilmiştir. VNH5019 motor sürücü Şekil 12'te gösterilmiştir.



Şekil 12. Motor Sürücü

(Kaynak: Pololu, 2022)

Bu motor sürücü doğrudan projede kullanılan mikroişlemci üzerine takılabilmektedir. Gücünü mikroişlemciden farklı olarak doğrudan motor bataryasından alabilmektedir. Motor sürücü sadece bir mikroişlemci kullanarak iki farklı motoru sürebilmektedir. Motorlar 12V için toplamda 11,2A akım çekerken bu sürücü 12V için tek motorda 12A ile 30A kadar akım verebilmektedir. İki motor aynı anda akım çektiğinde ise motor sürücü 24A ile 60A arasında akımı verebilecek kapasiteye sahiptir. Bu akım kapasitesi motor sürücünün robotlarda kullanılan motorlar için uygun olduğunu göstermektedir. Motor sürücünün mikro işlemciye ve motorlara bağlantısı Şekil 13'te gösterilmektedir.



Şekil 13. VNH5019 Motor Sürücü Mikroişlemci Bağlantısı

(Kaynak: Pololu, 2022)

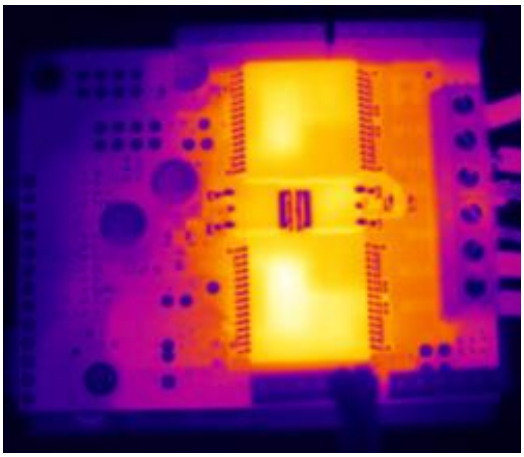
Motor sürücü 5V ile çalışabilecek kapasitedeyken aynı zamanda sahip olduğu koruyucularla 41V çalışabilmektedir. Motor sürücü aynı zamanda motordan gelen 16V geri besleme voltajına da karşı direnebilmektedir. PWM frekansı 20 kHZ çıkabilmekte bu da motorun daha az akımla sessiz kalkışına imkân verebilmektedir. Motor çalışma özellikleri Tablo 3'te paylaşılmıştır.

Tablo 3. DC Motor Sürücü Çalışma Tablosu

Çalışma voltaj aralığı (V)	5,5 - 24
Her ayak için MOSFET direnci (mΩ typ.)	18
PWM frekansı (kHz)	20
Akım duyarlılığı (V/A tipik)	0,14
Yüksek akım koruması (V)	24 – 27
Mantıksal yüksek voltaj koruması (V)	2,1
20A'de ısınma süresi (s)	20
15A'de ısınma süresi (s)	90
Kısa devrede çalışma akımı (A)	12

(Kaynak: Pololu, 2022)

Yüksek akım çektiklerinde motorlar ve motora bağlı devreler de ısınır. Motor sürücü devresi ve kontrol sistemleri bu nedenle mümkün olduğunca robotun bilgisayarı ve diğer elektronik devresinden izole olmalıdır. Motor sürücü yeterince zorlandığında 153°C sıcaklığa erişebilmektedir. Bu robot için oldukça tehlikeli bir durum olduğu için hava akımı ve soğutma konusuna dikkat edilmesi gerekmektedir. Motor çalışma sırasında çekilmiş termal görüntüler Şekil 14'te paylaşılmıştır.

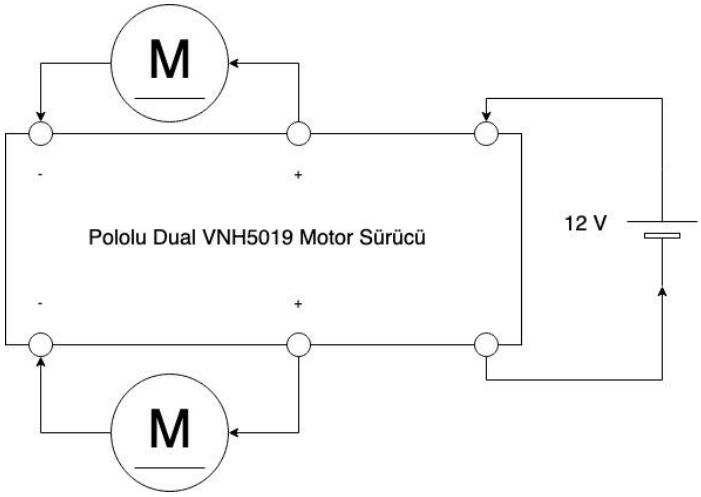


Şekil 14. VNH5019 Motor Sürücü Isınma Grafiği

(Kaynak: Pololu, 2022)

Motora bağlı enkoderin de elektriksel bağlantısına dikkat edilmelidir. Enkoder elektronik bir devre olduğu için hassas bir yapıdadır. Bu iki sistem aynı motor üzerinde olmasına rağmen mümkün olduğunca izole elektriksel yapıya sahip olması gerekmektedir. Motorlar çalıştıklarında harmonik gerilim üretmekte ve bu devrelere uzun çalışma sürelerinde zarar vermektedir. Motor devrelerinin bu zarardan etkilenmemesi için mümkün olduğunca diğer sistemlerden izole edilmesi gerekmektedir. Motor sürücüsü ile motor kontrol devresinin gücünün de sistemden ayrılması; motor kontrol devresi ile karar verici bilgisayarı koruyacaktır.

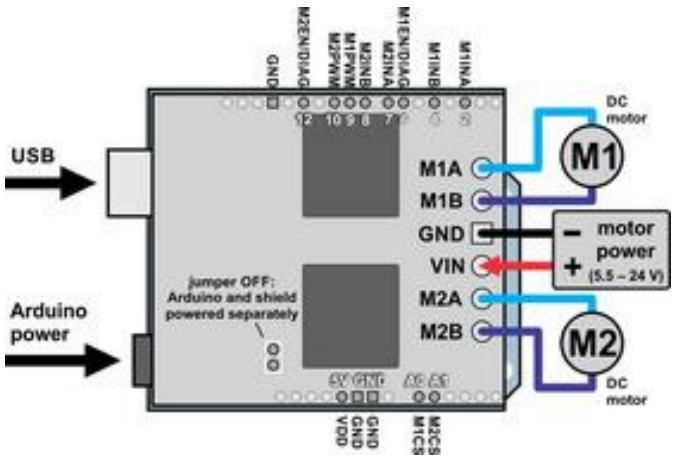
Motor, sürücüsü ve bataryası kendi içinde izole bir yapıya sahiptir. Motor elektriksel olarak diğer elektronik sistemlerden bağımsızdır. Motor sürücünün güç bağlantıları Şekil 15'te gösterilmiştir.



Şekil 15. VNH5019 Motor Sürücü Güç Bağlantıları

(Kaynak: Pololu, 2022)

Motor sürücü görüldüğü gibi sadece batarya ve motorlara bağlıdır. Bu nedenle başka hiçbir sistem ile etkileşimi bulunmamaktadır. Bu metot ile robot genelinde elektriksel koruma çok daha rahat olmaktadır. Mikro işlemci ile motor sürücü arasındaki bağlantı motor ile bağlantısını etkilememektedir. Motor sürücü gücünü hiçbir şekilde ana elektrik sisteminden almamaktadır. Bu elektrik sistemi Şekil 16'da gösterilmiştir.



Şekil 16. VNH5019 Motor Sürücü Arduino Bağlantıları

(Kaynak: Pololu, 2022)

2.1.2. Sensörler

Mobil robotta iç alan haritalandırma ve navigasyon işlemleri yapılabilmesi için çok sayıda sensör bulunması gerekmektedir. Bu sensörler robotun kendisi ile çevresindeki diğer nesneleri ayırmasına olanak sağlamak amacıyla kullanılmaktadır. Robotta kullanılacak sensörler seçilmeden önce navigasyon ve haritalamada gerekli sensörlerin listesi çıkarılmalıdır.

Navigasyon ve haritalamada robotun bulunduğu konumdan ne kadar ilerlediğini anlamak için robot tekerlerinin dönüş sayısının ve tekerlek çapının bilinmesi gerekmektedir. Bu veriyi de motora takılacak olan bir enkoder sensörü verebilecektir. Motor dönüş sayıları elimizde olduğuna göre şimdi robotun çevresindeki nesnelere olan uzaklığını algılaması gerekmektedir. Bu konuda uzaklık ölçme sensörleri kullanılmalıdır. Robot için konumunu en iyi algılamasını sağlayacak olan sensör LIDAR olarak geçmektedir. Bu sensör yüksek firekansta 360 derece uzaklık verisi almaktadır. LIDAR çalıştığı anda robot bulunduğu konumdaki tüm nesneleri hızlıca çıkarabilmektedir. Son olarak Kalman Filtresinde kullanılması için IMU kullanılması gerekmektedir. IMU robotun ivme hareketlerini, konum bilgisini ve hızını tespit edebilmektedir. Bu üç sensör robotun doğru navigasyon ve haritalandırma yapması için yeterlidir. Robot bu sensörler dışında bilgisayarında ve elektronik sistemlerinde bulunan ısı, voltaj ve akım koruma sensörlerine sahipdir. Bu sensörler elektronik sistemlerin güvenliği içindir. Sensörler ana devre üzerindeki sistemde otomatik olarak kullanılmaktadır.

Robot üzerinde bulunan sensörlerin bağlantıları ve kontrolleri önemlidir. Bu sensörler güvenli şekilde bağlanmalı ve mümkün olduğunca elektriksel gürültü ve hatadan uzak olmalıdır. Sensörlerin elektriksel gereksinimleri göz önüne alınarak tüm batarya kullanımları planlanmalıdır. Özellikle Lidar 360 derece ölçüm verisi almak için sürekli dönmesi gerekmektedir. Bu motor sistemden sürekli akım çekmekte ve elektronik gürültü üretmektedir. Proje yapılırken bu durum göz önünde bulundurulmalıdır. Eğer LIDAR bilgisayar üzerinden çalıştırılacaksa o zaman bilgisayara yeterince gücün verildiğine emin olmak gereklidir. LIDAR bağlantısı motor ve elektronik olarak ayrılabiliyorsa mümkün olduğunca ayrılmalıdır. LIDAR hareket ederken hareketinde bozulma olmamasına özen gösterilmelidir. Kabloların düzgün, dönüş hareketinin engelsiz olması LIDAR'ın en etkin şekilde çalışmasını sağlayacaktır.

LIDAR teknolojisi lazer ile mesafe ölçümüne dayanmaktadır . Lazer geliştirilmeden önce 1930'lu yıllarda mesafe ölçüm sistemleri, atmosferik uzaklıkları ölçmek için kullanılmaktaydı. 1960 yılında lazerin geliştirilmesi ve 1980'li yıllarda etkin şekilde kullanılması ile lazer ile mesafe ölçme teknikleri gelişmiştir. LIDAR'ın ölçtüğü uzaklık için d ile ışık hızı c ve toplam ölçüm süresi t ile ifade edildiğinde; Denklem 7 ile uzaklık hesabı yapılabilmektedir.

$$d = \frac{1}{2}tc \tag{7}$$

LIDAR ile uzaklık ölçümü kolaylıkla yapılmakla birlikte bu sistemin bazı dezavantajları da bulunmaktadır. LIDAR üzerinde kullanılan ışığın gücüne göre ölçüm yapabilmektedir. Lazer ışığının kaynaktan çıkması ve tekrar LIDAR üzerinde bulunan ışık sensörüne gelmesine göre ölçüm yapmaktadır. Eğer ortamda bulunan maddeler ışığı emen bir yapıda olursa bu durumda lazer ya yanlış ya da hiç ölçüm yapamayacaktır. Ölçüm yapılan ortamın ışığı farklı yönlere yansıtması da ölçümü kısıtlayabilmektedir. Bu gibi etmenler LIDAR ile doğru ölçüm yapabilmesi önündeki en büyük engeldir. LIDAR ile ölçüm yapılacak alanda bu cihazın ölçüm yapacağı materyallerin bu kriterlere uyduğuna emin olmak gerekir.

LIDAR sensörü iki ana unsurdan oluşmaktadır. Bu unsurların ilki LIDAR içindeki lazeri fiziksel olarak çeviren motordur. Motor gücünü bilgisayardan ya da güç bankasından aldığı için 5V üzerinden çalışmaktadır. Motorun içerisinde harmonik oluşabileceği için mümkün olduğu kadar elektronik devrelerinden farklı bir güç kaynağı ile beslenmesi gerekmektedir. LIDAR motor dışında bir de üzerinde lazer alıcı ve verici bulunan bir elektronik karta sahiptir. Bu elektronik kart sürekli olarak dönmekte ve üzerinde bulunan lazerler yardımı ile kendisi ile ölçüm yapabildiği en uzak mesafeye kadarki uzaklığı bulmaktadır. Elektronik devrenin elektriği manyetik alan üzerinde oluşmaktadır. Elektronik kartın altında bulunan sabit kısımda sarılı olan tellere elektrik verilmekte bu verilen elektrik kartın üzerinde bulunan parçadaki elektrik sargılarını beslemektedir. Burada oluşan elektriksel akım, kartın tüm elektriksel ihtiyacını karşılamaktadır. Bu sarımların oluşturduğu devre oldukça hassas bir yapıdadır. Bu nedenle kartın yanmaması için buradaki devreye mümkün olduğunca kararlı bir elektrik akımı verilmeli.

Proje kapsamında öncelikle Slamtek A1 LIDAR kullanılmıştır. A1 LIDAR 2 Hz ile 10 Hz arasında seçilebilen 3 farklı modla 360 derece dönerek 8000 örnek alabilmektedir.

LIDAR çevresindeki nesneleri 12 metreye kadar ölçebilmektedir. Motor dönüş hızı 2 Hz, 5,5 Hz ve 10 Hz olarak ayrılmaktadır. Bu hız PWM kontrolü ile yapılabilmektedir. LIDAR'ın motor dönüş hızı ve örneklem sayısı yükseldiği zaman çok da yüksek çözünürlükle detaylı ölçümler yapabilmektedir. A1 LIDAR bilgisayara bağlandığı USB kablosu üzerinden gereksinimi olan 5 V enerjiyi almakta ve bu enerji ile hem motoru döndürmekte hem de mesafe ölçümü yapmaktadır. LIDAR kendisine güç verildiği anda hem dönmeye hem de ölçüm yapmaya başlamaktadır. LIDAR verileri robot işletim sistemi (ROS Robot Operating System) üzerinden okunabilmekte ve 8 Hz hızında veriyi yayınlayabilmektedir. Kullanılan bu LIDAR'ın en önemli avantajı hesaplı ve kolay bulunur olmasıdır.

A1 LIDAR'da sadece tek bir USB bağlantı noktası bulunduğu için hem motor hem de elektronik devrenin tüm enerji ihtiyacı bu kablo üzerinden karşılanmak zorundadır. Robot üzerinde enerji taşınabilir güç kaynağı (powerbank) ile sağlanmaktadır. Piyasada çok çeşitli bulunmakla birlikte yüksek akım sağlayabilen taşınabilir güç kaynağı seçeneği azdır.

Tez kapsamında kullanılan Slamtek A1 LIDAR ilk aşamada bilgisayar taşınabilir güç kaynağı bağlanmış, buradaki USB çıkışından da LIDAR'a güç verilmiştir. Bu yöntemde ilk aşamada veriler okunmuştur. Hatta verileri RVIZ üzerinden kısmen de görme imkânı da oldu. Fakat belli bir süre çalıştıktan sonra LIDAR'dan veri gelmemeye ya da gelen veri her zaman sıfır olmaya başladı. Robot üzerinde bilgisayar olarak kullanılan Raspberry kendi üzerinde güç sorunları yaşandığını olay günlüklerinde listenledi. Raspberry'nin kararlı çalışması için en az 3 A akım verilmesi gerekirken Raspberry modülüne LIDAR dahil 1,5 A akım verildiği için LIDAR okuma kabiliyetini kaybetmiştir. Bu durum LIDAR sensörlerinin hassasiyeti konusunda dikkatli olunması gerektiğini ortaya koymuştur. A1 LIDAR ile ilgili bir diğer husus da açıldığı anda üzerinden veri okunmasa bile dönerek akım tüketmesi olmuştur. Özellikle bilgisayar açılırken normal zamandan çok daha fazla enerji çekmekte LIDAR da sürekli dönmeye çalıştığı için bilgisayarın açılış enerji ihtiyacını artırmaktadır.

A1 LIDAR ile ciddi sorunlar yaşadıktan sonra alternatif LIDAR araştırılmıştır. Slamtek firmasının bir üst sürümü olan A2 LIDAR'ların enerji bakımından motor ve sensörün ayrımına daha fazla dikkat edildiği ve sadece üzerinden veri okunurken açıldığı tespit

edilmiştir. Bu nedenle ikinci LIDAR seçimimde A2M8 LIDAR tercih edilmiştir. Ayrıca Raspberry Pi enerji gereksinimini sağlayabilecek 3 A çıkış veren çok daha kuvvetli bir taşınabilir güç kaynağı seçilmiştir. A2M8 LIDAR, iki ayrı USB ile veri okuma ve motor kontrol işlemlerini yapmaktadır. Bu nedenle veri okuma kablosunu doğrudan yüksek güçle beslenmiş olan Raspberry Pi modülüne diğer motor kablosunu da farklı bir taşınabilir 2 A güç kaynağına bağlanmıştır. Bu enerji gereksinimleri sağlandıktan sonra A2M8 LIDAR verimli şekilde çalışarak harita çıkarma ve navigasyon işlemlerinde kullanabilir duruma gelmiştir.

A2M8 LIDAR, UART bağlantı tipi ile USB üzerinden haberleşmektedir. A2 LIDAR için USB adaptöründe iki seçenek olarak UART hızları verilmektedir. Adaptör üzerinde bulunan bir anahtar ile istenirse 115.200 BPS ya da 256.000 BPS olarak seçilebilmektedir. LIDAR 0,2 m ile 12 m arasındaki tüm nesneleri ortalama 10 Hz hızda dönerek tespit edebilmektedir. LIDAR 5 V ile çalışan sistem kartlarında 450 – 600 mA arasında akım dolaşmaktadır. LIDAR güç tüketimi 2,25 W ile 3 W arasındadır. ROS ile A2 LIDAR verilerinin okunması için LIDAR tipi seçilmeli ve LIDAR frekansı ile UART hızı belirlenmelidir. İki farklı kablo olması LIDAR enerjisinin taşınabilir güç kaynağı tarafından sağlanmasına olanak sağlamaktadır. LIDAR'ın biri kısa biri uzun iki kablosu bulunmaktadır. Kısa kablo sensör bilgilerini okumak içindir. Bu kablonun kesinlikle bilgisayara takılması gerekmektedir. Uzun kablo ise istenirse bilgisayara istenirse harici bir kaynağa takılabilmektedir. Her iki kablo farklı enerji sistemlerine bağlı olabilmektedir. LIDAR ile yapılan testlerde harici kaynaktan 0,6 A akım çekildiği görülmüştür. Eğer harici kaynak kullanılmayacaksa kullanılan bilgisayarın toplam enerjisine bu akımın ekleneceği unutulmamalı ve buna göre bilgisayarın beslenmesi gerekmektedir.

Proje kapsamında UART bağlantı hızı 115.200 BPS olarak ayarlanmıştır. 8 kHz hızında veri almak için ayarlar yapılmıştır. LIDAR enerjisi harici bir güç kaynağından verip veri kablosu Raspberry Pi 4 bilgisayara aktarmaktadır. LIDAR sadece veri okunmaya başladığı zaman çalışmakta ve güç kaynaklarından akım çekmektedir. LIDAR çalışırken akım çekmesi özelliği nedeniyle bilgisayar açılırken kendi enerjisini korumasını sağlamaktadır. LIDAR dönmesinin gövdeyi sallamaması için üst gövdenin sıkı monte edilmesine özen gösterilmelidir.

Robot üzerinde bulunan bir diğer sensör de IMU dur. IMU sensörleri tek bir sensörden oluşmayan birden fazla sensörün sonuçlarını veren elektronik ölçüm sensörüdür. IMU ölçüm tiplerine göre 3, 6, 9 veya 10 DOF olarak ayrılabilmektedir. Bu değerlendirme ölçüm miktarına doğrudan bağlıdır. 9 DOF bulunan bir IMU'yu örnek alırsak bu sensörde 3 kanal ivme ölçer, 3 kanal jiroskop 3 kanal manyetometre bulunmaktadır. Tüm bu kanalların toplamı sensörün özelliği olan 9 DOF değerini sağlamaktadır. İvme ölçerler bulundukları ortamdaki titreşimleri, hareketlenmeleri ve ivmelenmeleri ölçmektedir. İvme ölçer sensörün içinde bulunan materyaller hareket anında büzülmektedir. Büzülme hareketi sensör içindeki elektriklenme ile dijital sinyalleri oluşturmaktadır. IMU tek bir hat üzerinden farklı sensörlerin işlenmesine olanak sağlamaktadır. IMU günümüzde robotlarda, otonom kara, deniz ve uzay araçlarında yaygın olarak kullanılmaktadır.

IMU ile hızın zamana göre değişim miktarı ya da hızın zamana göre türevine göre ölçülen ivme ölçülebilmektedir. İvme vektörel bir nicelik olup cismin hem yönünün hem de hızının zamana göre değişimini vermektedir. İvme formülü cisme uygulanan F hareket yönünde uygulanan vektörel kuvvetinin yine aynı cismin skaler olan m kütlesine bölünmesi ile bulunmaktadır. Bu durum Denklem 8'de gösterilmiştir.

$$a = \frac{F}{m} \tag{8}$$

İvme ölçer, bir cismin sahip olduğu ivmeyi ölçmek için kullanılmaktadır. İvme ölçer bu hesaplamaları yaparken içerisinde bulunan belirli özelliklere sahip kütlenin sensör içindeki konumuna bakarak hesaplama yapmaktadır. Sensörün çalışma prensibini içinde bir kütle ve sabit dirençli bir yay bulunan silindirik muhafaza kap üzerinden işler. Kaptaki yay ve kütle ilk andaki konumu 0g olarak değerlendirir. Kap hareket ettirildiğinde kütlenin üzerine harekete göre oluşan kuvvetin etkisi ile yay esnemektedir. Bu esneme ile kütlenin yerinin değişimine göre ivmesi ölçülebilmektedir. İvme ölçer bu cismin konumu referans alınarak 3 eksende 3 ayrı analog sinyal üretebilmektedir.

İvme sensörleri temel olarak analog ve sayısal çıkışlı olmak üzere iki kısımda incelenmektedir. Analog ivme sensörleri, oluşturdukları verileri hiçbir işlem yapmadan doğrudan sürekli olarak gerilim değişikleri ile çıktı verirler. Sayısal ivme ölçerlerde ise

gerilim değişimlerine bakılmaz sensör kendi içinde gerekli hesaplamaları ve modülasyonları yaparak doğrudan bilgisayarlarda kullanılabilecek sayısal değerleri üretmektedir. Sensör seçilirken göz önünde bulundurulması gereken diğer konular da eksen sayısı, hassasiyeti, bant genişliği ve empedans olarak belirtilmektedir. Sensör eksen sayısı sensör tasarımına göre 2 veya 3 eksen olabilecektir. Uygulamaya göre 2 eksenli sensörler kullanılabilir. Piyasada genellikle 3 eksenli sensörler satılmaktadır. Sensörün hassasiyeti tüm sensörler için önemli bir konudur. Yapılacak uygulamaya uygun ölçüm kabiliyetine sahip sensör seçilmelidir. Bant genişliği sensörden alınacak ölçüm hızını ifade etmektedir. Daha yüksek bant genişliğine sahip sensörler çok daha fazla ölçüm yapacak ve gelen verilerin çözünürlüğü daha fazla olacaktır. İvme sensörünün empedansı sistemde ölçümleri okuyacak diğer ekipmanlara uygun olması gerekmektedir. Yüksek veya düşük empedansa sahip sensörlerden okuma yapmak sistemdeki diğer cihazlar için zor olacağı için bu değerlerin doğru seçilmesi gerekmektedir.

Jiroskop dengeleyici, dönme ve yönelme hareketlerini tespit ederek üzerinde bulunduğu cihazın yalpalama hareketlerinin engellenmesi veya yönelme işlemlerinin kontrolü için kullanılmaktadır. Jiroskop dengeleyici kuvvet veya açısal momentumun korunması prensibi ile çalışarak kendi içinde hızla dönerek bu momentumu elde etmektedir. Mekanik Jiroskoplarda ölçüm yapılırken yerçekimi etkisini azaltmak için dik şekilde konumlanması önerilmektedir. Jiroskoplar birbirleri üzerine perçinlenmiş iki halka arasında dönen bir çemberden oluşmaktadır. Ortadaki çember normal koşullarda hızla döndüğü için dönmenin ya da yönelmenin olduğu tarafın tersine doğru dönmesi prensibine göre çalışır. Bu şekilde iki çember kullanıldığında bir çember öbür çemberin üzerine perçinlenerek gimbal oluşturulur. Bu iki çemberin dönmesi ile 3 eksenli bir ölçüm yapılabilir hâle gelmektedir.

Günümüzde mikro elektro mekanik sistemlerin (MEMS) gelişmesi sonucu bu sensörler çok küçük boyutlara inmiştir. Bu yeni sensörler eski mekanik sensörlerden farklı olarak titreşim etkisi ile işlem yapmaktadır. Sensörler üzerinde bulunan T şekilli iki yapısal elementin ortasında bir algılayıcı kol ve bu üç parçayı birbirine bağlayan bir stator bulunmaktadır. Sensör üzerindeki açısal momentum veya dengeleyici kuvvetin değişmesi ile algılayıcı kol ve T şekilli çubuklar farklı hareketler yapmaktadır. Oluşan farklı hareketlere göre dönüş belirlenmektedir.

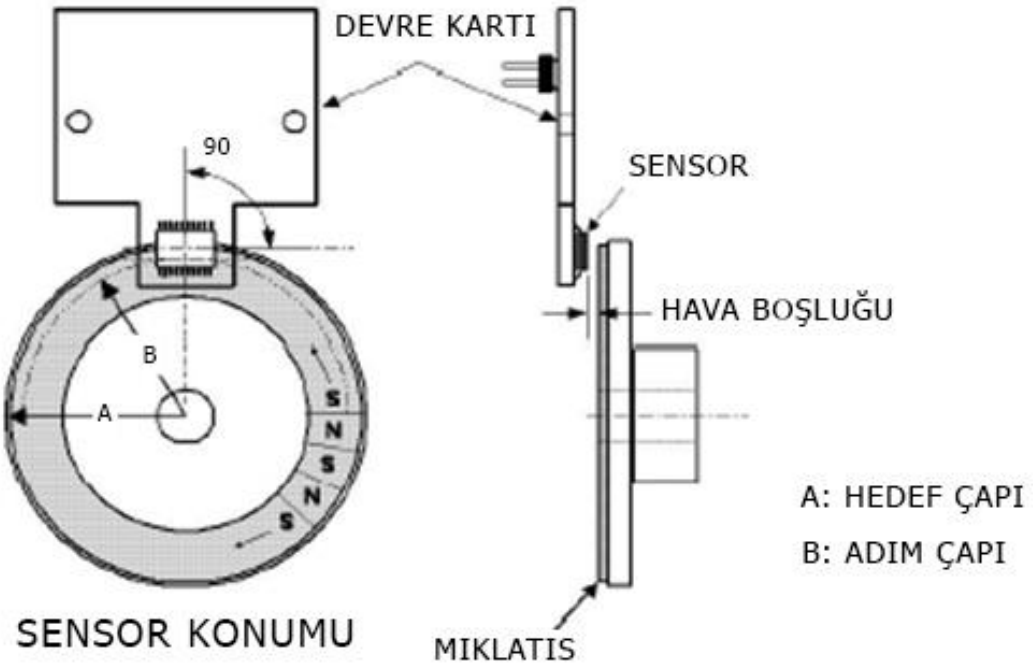
Teknolojinin gelişmesi ile bu şekilde oluşturulan sensörlere birçok farklı alternatif de üretilmiştir. Bu alternatiflerin bazıları yüzük lazer jiroskop, fiber optik jiroskop, quantum jiroskop, titreşim jiroskoplarıdır. Projede devre kartında titreşim ile ölçüm yapan jiroskop kullanılmıştır. Elektronik sistemlerde bu tip jiroskopların kullanımı yaygındır.

Manyetik alanı ölçmek için kullanılan manyetometre vektörel olarak manyetik alanın gücünü ve yönünü ölçebilmektedir. Manyetik alan sensörü pusulalara benzer şekilde dünyanın manyetik alanına göre sensörün konumunun bulunmasını sağlamaktadır. Manyetometreler pusula gibi tek bir çubuğun kuzeye yönelmesi prensibi ile çalışır. Manyetik alanın ölçümü için elektronik kartlarda sıklıkla kullanılan iki yöntem vardır. Bunlardan birincisi manyetik alana tepki gösteren bir manyetik sıvı kullanıp bu sıvının şekil değişimlerinin yorumlanmasıdır. Bir diğeri de elektriksel kuvvetlere direnç gösteren bir materyalin direncinin ölçülmesi ile yapılmaktadır. Projede kullanılan sensör, elektriksel rezistans ile ölçüm yapmaktadır.

Proje kapsamında ilk kullanılan IMU DFROBOT tarafından üretilen gravity-BMI 160 6 eksenli modelidir. Bu modelde veriler Raspberry üzerinden kolayca okunabilmektedir. Bu IMU 6 eksenli olduğu ve BMI160 serisi IMU için ROS paketleri tespit edilememiştir. Tez çalışmasında, Çin'de kaynaklı GY-BNO055 tipi IMU denenmiştir. Bu IMU'da veri okuma için gerekli donanım doğrudan IMU ile gelmediği için PULL-UP işlemi yapılması gerekmiştir. PULL-UP sensör için gerekli enerjinin doğrudan bilgisayar üzerinden alınamadığı durumlarda kullanılmaktadır. Temel prensibi Bilgisayarın sensöre bağlandığı kabloya gerilimi yükseltmek için belirli bir dirence bağlı güç kaynağı ile destek verilmesi gerekmektedir. Bu enerji olmadan sensör çalışmamaktadır. Bu sensörde çeşitli sorunlar yaşadıktan sonra Adafruit firmasının BNO055 ile geliştirmiş olduğu IMU sensör tercih edilmiştir. Bu sensör diğer sensöre göre daha kararlı olduğu için veri okumada sensör kaynaklı sorun yaşanmamıştır.

Enkoderler motorun dönüş sayısını ölçmek için geliştirilmiş sensörlerdir. Enkoder delikli bütün panelin üzerinde bulunmaktadır. Panel üzerinde elektronik devreler ve sayım ekipmanları bulunmaktadır. Enkoder paneli şafta doğrudan bağlı olduğu için şaftın döndüğü kadar dönmektedir. Enkoderin üzerinde sayım yapılabilmesi için motora paralel

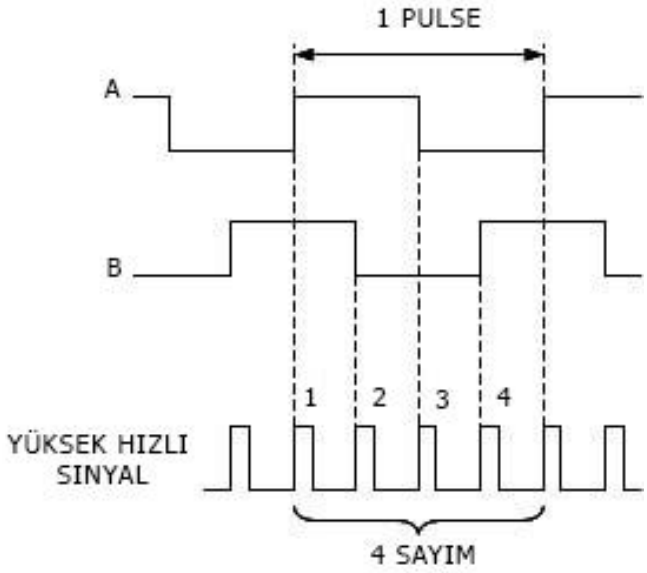
sıralanan küçük mıknatıslar bulunmaktadır. Mıknatıs yerleşimleri Şekil 17'de gösterilmiştir.



Şekil 17. Enkoder Üzerinde Mıknatıs Yerleşimi

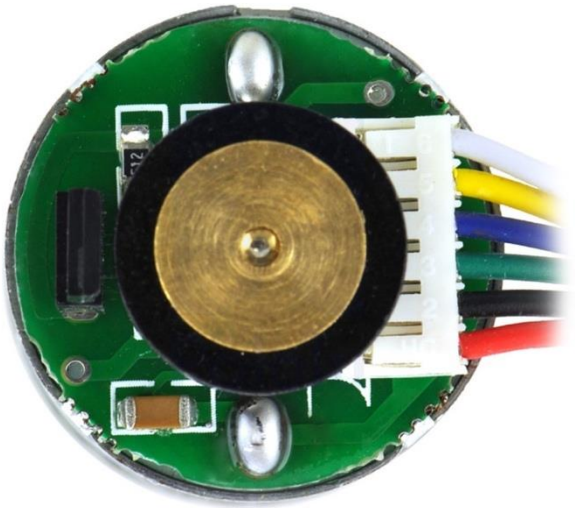
Panel dönerken üzerindeki mıknatıslar sıra ile bir hal sensörünün karşısına geçerler. Bu hal sensörü kendisinin önünden geçen bu mıknatıslardan dolayı bir akım oluşturur. Enkoder üzerindeki devreler bu akımları bir değer olarak görürler. Bu değerler daha sonra da enkoderi okuyan sistemlere doğrudan işlemden geçmeden gönderir. Robot üzerinde kullanılan enkoder sensörleri motor ile gelmiştir. Bu motora doğrudan bağlantıları yapılmıştır.

Enkoderde çift kanal Hal efekt sensörü bulunmaktadır. İki kanaldan gelen veriler her dönüş için sayım yapmaktadır. İki kanal arasındaki farklarla enkoder dönüş sayısı belirlenmektedir. Enkoderin her dönüşündeki sayım miktarına CPR adı verilmektedir. Enkoderin dönüşündeki sayımların değişmesine pulse denilmektedir. Mikroişlemciler bu pulseleri okuyabilmektedir. Eğer mikroişlemci pulse olarak işlem yapacaksa bu durumda her dönüşteki pulse PPR değerini dikkate alması gerekmektedir. Enkoder A ve B kanalları ile PPR ve CPR oluşumları Şekil 18'de gösterilmiştir.



Şekil 18. İki Kanallı Enkoderden Sayım ve Pulse Hesaplanması

PPR değeri A kanalında sinyal boyunun bir ve sıfır değerleri arasında olması bir pulse olarak ifade etmektedir. 1 PPR değeri içinde 4 adet CPR değeri bulunmaktadır. CPR tek sayımın bulunması için iki adet kanala ihtiyaç vardır. 1 CPR değeri A kanalının durum değişimi ile B kanalının durum değişimi arasındaki aralığı göstermektedir.



Şekil 19. Motor Enkoderi

(Kaynak: Pololu, 2022)

Tez kapsamında yapılan mobil robot için kullanılan enkoder Şekil 19'da gösterilmiştir. Kullanılan enkoder motor ile birleşik olarak satılmakta ve manyetik bir enkoderdir. Bu enkoder ve motor POLOLU firması tarafından üretilip birleştirlmiştir. Enkoder doğrudan şaft ile birleşik ve kabloları tek bir devre kartı ile hem motora hem de enkodere gittiği için çok daha kararlı çalışma sağlamaktadır. Enkoderin CPR değeri 48 olarak verilmiştir. Bu değer ile her dönüşte 1632 sayım yapabilmektedir.

2.1.3. Güç Tüketimleri ve Bataryalar

Robotun güç tüketimi, robotun çalışması için çok önemlidir. Hem işlemlerin doğru yapılması hem de robot üzerindeki cihazların zarar görmemesi için gereklidir. Motor üzerindeki sensörler, motorlar ve bilgisayar oldukça hassas cihazlardır. Çok hassas oldukları için çalışma zamanında birbirlerini etkileyebilmektedirler. Motorlar sistemden çok ciddi akım çekmektedir. Motorlar ayrıca sisteme reaktif akım ve harmonikler de verebilmektedir. Motorun yüksek akım çekmesi sensörlere ve bilgisayara daha düşük veya dalgalı akım gitmesine olanak sağlayacaktır. Dalgalı akım sistemdeki sensörlerin yanlış çalışmasına ve hatalı veri göndermesine imkân sağlayacağı gibi uzun vadeli çalışmada sensörün de bozulmasına sebep olabilecektir. Bu sorun bilgisayarlar için de geçerli olmakla birlikte çoğu bilgisayar kendi içlerinde koruma sağlamaktadır. Bilgisayarlar kendilerini korurken çoğunlukla çevre birimlerin yani sensörleri taktığımız mikroişlemcilerin enerjilerini kesmektedir. Bilgisayarın sağladığı bu kesinti aynı zamanda sensörler için de sorun oluşturabileceği için tüm önlemlerin alınması gerekmektedir. Tez kapsamında geliştirilen robotta güç tüketen sistemler olarak bir adet Raspberry Pi, iki adet arduino mikroişlemci, iki adet enkoder, iki adet motor, bir adet IMU ve bir adet LIDAR bulunmaktadır. Bu gücü karşılamak üzere bir adet 12 V – 30 A Li-Po batarya, 5 V – 3 A, 5 V – 2 A taşınabilir güç kaynağı kulanılmıştır.

Mikroişlemci olarak iki adet Arduino UNO R3 kullanılmıştır. Bu mikroişlemciler fazla enerji tüketmemektedir. Arduino güç tüketimi tek işlemci için yaklaşık 5 V – 100 mA olarak belirtilmektedir. Arduino güç ihtiyacı ve haberleşme ihtiyaçları için tek bir USB kablosunu kullanabilmektedir. Enkoderler güçlerini harici olarak ya da kendilerini okuyan mikro işlemcilerden alabilmektedir. Enkoderler genellikle düşük güçle çalışırlar. Projede kullanılan enkoderin 5 V - 10 mA enerji gereksinimi vardır. Bu enkoder gücünü Arduino modülden sağlamaktadır.

Arduino mikroişlemci ile motorlara güç verilirken dikkat edilmesi gerekmektedir. Burada dikkat edilmesi gereken en önemli husus motorların Arduino'nun sağlayabildiği güç ile sürülebiliyor olması veya olmamasıdır. Bazı motorlar düşük kapasiteli olduğu için motor sürücüsü doğrudan ardunino üzerine takılarak motor sürülebilir. Eğer motor Arduino'nun sağlayabileceğinden çok daha fazla güce ihtiyaç duyuyorsa bu durumda sürücü üzerinden harici güç verilmesi gerekmektedir. Harici güç gerektiren motorlar alındığı zaman motora harici güç verebilecek bir sürücü seçilmesi gereklidir.

Tezde kullanılan motorlar yüksek performanslı olduğu için yük durumunda 6 A akım çekebilmektedir. Bu akım değeri Arduino'nun sağlayabileceğinden çok daha yüksek olduğu için motorun bu şekilde dönmesi olanaksızdır. Bu sorunu çözmek için çok daha yüksek enerji çekebilen ama kontrol için Arduino'yu kullanan bir sürücü seçilmiştir. Motorlar 6 A akım çekebildiği için 12 A üzerinde akım kapasitesine sahip sürücü ve güç kaynağı kullanılması sağlıklı bir motor kontrolü için gereklidir.

Motorlar ilk hareketleri sırasında normalde çektiklerinden çok daha fazla akım çekmekte bu da sistemde anlık akım düşümlerine sebebiyet vermektedir. Bu akım değişimlerinin yanında bir de motorlardan gelen harmonik ve reaktif akımlar da elektronik parçalara zarar verebileceklerdir. Bu sorunlarla karşılaşmamak için sistemde motorlar ile elektronik sistemleri mümkün olduğunca ayırmak gerekmektedir. Bu iki sistemi ayırmak için araya yüksek kapasitörler ve dirençler ile kontrol sağlanabileceği gibi doğrudan farklı güç kaynakları kullanılarak da sistemler izole edilebilir. Tez projesi kapsamında sistemlerin doğrudan izolasyonu tercih edilmiştir. Motorları besleyen güç kaynağı yeterince güçlü olmasına rağmen, motordan başka bir sistemi beslemek için kullanılmamıştır.

Projede kararlı akım sağlayan ve güçlü olan Li-Po piller ve Polulou dual VNH5019 sürücü kullanılmıştır. Bu pil ve sürücü seçilmeden önce farklı batarya ve sürücüler denenmiştir. İlk olarak bu motorlar daha önceden Adafruid'in dört motor sürebilen L293D ile kontrol sağlayan motor sürücü kartı ile kontrol edilmeye çalışılmıştır. Bu kartın gücü her bir kanaldan 0,6 A olduğu için motorları kaldırmaya yetmemiştir. Bu karttan sonra L298N ile kontrol edilmeye çalışılmıştır. Fakat bu kartın da her bir kanaldan verebileceği en yüksek akım 2 A olduğu için motoru çalıştıramamıştır. Motor ayrıca röle ile yönetilmeye çalışılmış fakat bu sistem de röleler sürekli ya tam ya da hiç akım veremediği için PWM kontrolü yapılamamıştır. Polulou motorlara uyumlu aynı anda iki

motor sürebilen VNH5019 sürücüsü kullanılmıştır. Yeni sürücü gerekli güç gereksinimlerini karşılamış ve sorun çözülmüştür.

Tez için geliştirlen mobil robotun bataryası konusunda ilk önce motorlar da diğer sensörler gibi taşınabilir güç kaynağı ile beslenmeye çalışılmıştır. Kullanılan motorlar çok güçlü olduğu için robotun hareket edemediği gözlemlenmiştir. Bu sorundan sonra 12 V – 6 A akım veren bir kuru akümülatör kullanılmıştır. Bu akümülatör tek motor için yeterince güçlü olmasına rağmen hem çok ağır hem de iki motorun aynı anda kalkması durumunda iki motora da yeterli akımı sağlayamamıştır. Bu işlemlerden sonra Li-Po pil ile motorun sürülmesi için 12 A üzeri akım verebilen bir pil seçilerek sorun çözülmüştür. Yüksek verimli motorların kalkışları oldukça atik ve kıvrak hareketler yapabilmekte ancak güç tüketimleri sorun olabilmektedir. Bu nedenle Li-Po veya aynı derecede güç verebilen kaynakların kullanılması gerekmektedir.

Robotta bilgisayar olarak kullanılan Raspberry Pi 4'ün çalışması için en az 5 V – 2,5 A akıma ihtiyaç duyulmaktadır. Bu akım bilgisayar üzerinde kullanacak sistemleri kapsamamaktadır. Bu nedenle güç gereksinim hesapları yapılırken Raspberry pi için daha yüksek akım verilmesi gerekmektedir. Raspberry tüm sensörleri yönettiği için bu sensörler harici güç kaynağı kullanmadıkları taktirde tüm gücünü Raspberry üzerinden alacaktır. Projede kullanılan Adruino kartlar ve IMU oldukça düşük akıma ihtiyaç duymaktadır. Bu nedenle akım gereksinimleri tolere edilebilirken LIDAR'ın akım ihtiyacı üzerindeki motor yüzünden çok yüksektir. Bu nedenle LIDAR'ın enerjisini ayırmak ya da Raspberry'ye LIDAR'ın ihtiyacını karşılayacak güçte akım verilmelidir. Bilgisayarlar hassas cihazlar olduğu için güç kaynakları da gürültüden ve dalgalanmalardan uzak olmalıdır. Bilgisayarın güç kaynağı için 3 A sabit amper verebilen Philips powerbank kullanılmıştır.

2.2. Mobil Robot Mekanik Tasarımı

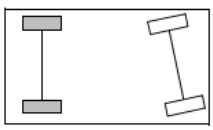
Mekanik tasarım robot açısından oldukça önemli bir konudur. Robotun elektronik ve yazılım sistemleri tam olarak çalışsa da mekanik olarak doğru tasarım yapılmazsa navigasyon ve haritalama işlemlerinin yapılması zorlaşacaktır. Robotta mekanik olarak dikkat edilmesi gereken konular tekerlerin istenilen yöne gidebilmesi ve bu hareketi sırasında sürtünme ve zorlanmaların en aza inmesidir.

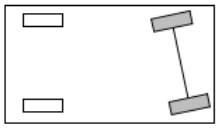
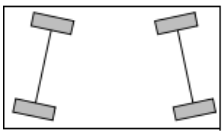
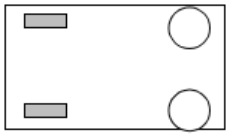
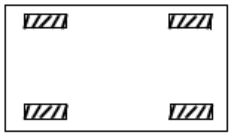
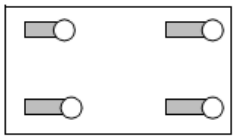
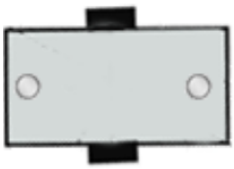
Mobil robotun mekanik tasarımda öncelikle dikkate alınması gereken diğer konu gövdedir. Gövde tipi çeşitli şekillerde olabildiği için robotun görevine uygun olarak seçilmesi gerekmektedir. Her gövde tipinin farklı uygulamalar için avantajları ve dezavantajları bulunmaktadır. Gövde ayrıca parçaların yerlerine yerleşmeleri, mekanik dayanımları, tasarımları önemlidir. Robotun gövdesinin çarpmalara karşı dayanıklı olması gerekmektedir. Gövde robotun içinde veya üzerinde bulunan hassas ekipmanların korunmasından sorumludur. Eğer görevini tam olarak yerine getiremezse robot içindeki pahalı elektronik aksam zarar görebilecek, ciddi maddi kayıplar yaşanabilecektir. Robotun gövdesinin seçilmesinde ve tasarlanmasında ayrıca estetik unsurlar da göz önünde bulundurulmalıdır. Gövdeye uygun tekerler kullanılmalıdır.

Günümüzde mobil robotlar için farklı gövde tipleri kullanılabilmektedir. Bu değişikliklerin temel nedeni yapılacak uygulamaya uyum sağlanmasıdır. Çoğu temizlik robotu uygulamasında mümkün olduğunca çok alanda temizlik yapabilmek ve çarpışmalardan robotun çok daha hızlı kurtulmasına olanak vereceği için yuvarlak veya elips gövde tasarımları tercih edilmektedir.

Robotlarda sıklıkla kullanılan bir diğer gövde tipi de dört köşeli gövde tipidir. Bu gövde sisteminin tercih edilmesinin amacı robotun dar noktaları görebilmesi, kapılardan kolaylıkla geçebilmesidir. Bu çok kullanılan tiplere alternatif olarak uygulamaya göre farklı gövdelerde robotlar da vardır. Bu tipteki robotları teker yerleştirmelerine göre 7 tipe ayırmak mümkündür. Bu tiplere göre ayrım yapılması sırasında hangi motorun eksende hareket ettiği, tekerlerin konumları, hangi motorun bir aktüatöre bağlı olduğu, tekerlerin tiplerine göre ayrım gösterilmektedir. Tablo 4'te kullanımlar gösterilmiştir.

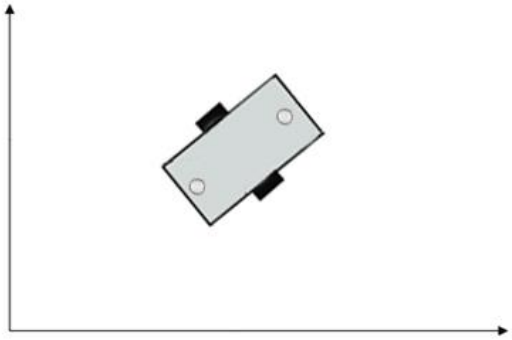
Tablo 4. Teker Yerleşimlerine Göre Mobil Robotlar

Teker Yerleşimi	Açıklama
	Araçta arkasında iki motorlu teker önünde ise yönelmeyi sağlayan iki teker bulunur. Tekerler yönelme için eşit dönüş yapmazlar. Arkadan çekişli arabalar bu yapıda üretilmektedir.

	Araçta önünde iki adet yönelen ve motora bağlı olan teker bulunmaktadır. Arkadaki iki teker ise sabit bulunmaktadır. Bu sistemi önden çekişli arabalar kullanmaktadır.
	Araçta dört yönelen ve motora bağlı teker bulunmaktadır. Bu sisteme arazi araçları örnek gösterilebilir.
	Aracın arkasında birbirinden bağımsız iki adet motorlu teker, ön tarafta ise iki adet sarhoş teker bulunmaktadır.
	Araçta birbirinden bağımsız hareketler yapabilen dört adet teker bulunmaktadır.
	Araçta birbirinden bağımsız hareket edebilen 4 adet teker bulunmaktadır. Bu tekerler castor teker olarak adlandırılan robot hareketlerinin doğrudan teker ile yapılabildiği tekerlerdir.
	Aracın orta kısmında birbirinden bağımsız iki teker motorlara bağlıdır. Aracın başında ve sonunda iki adet sarhoş teker bulunmaktadır.

(Kaynak: Siegward, 2014)

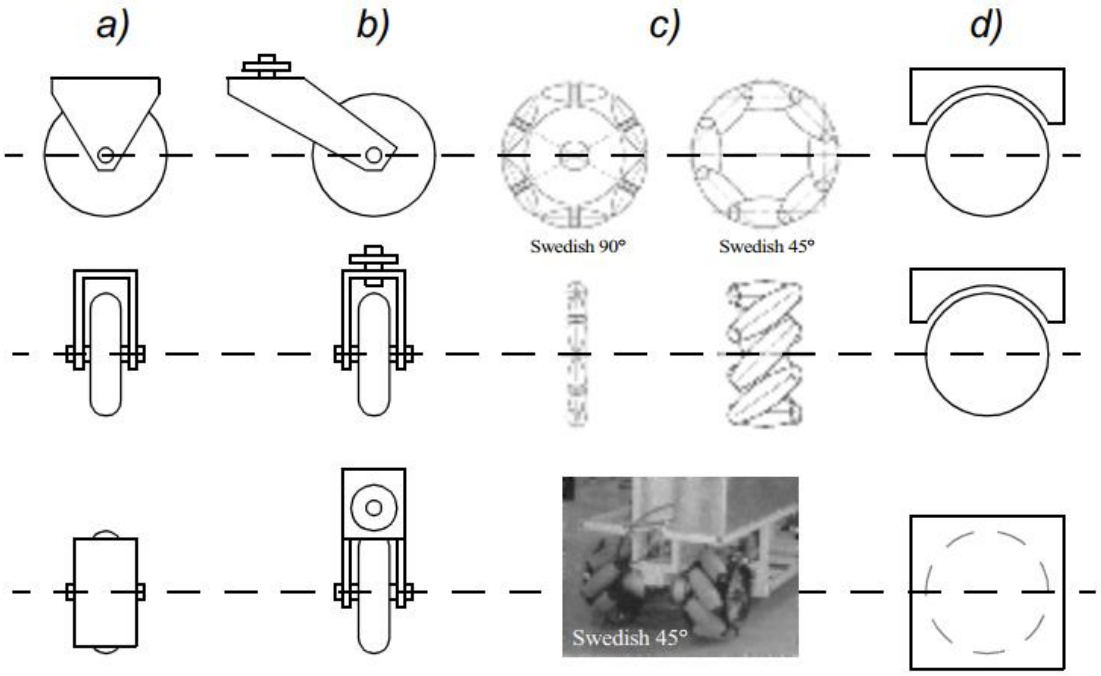
Proje kapsamında dikdörtgen bir robot gövdesi kullanılmıştır. Bu seçimin ana sebebi robotun iki katlı olmasıdır. Robotun kat aralarından elin kolaylıkla orta kısıma ulaşması herhangi bir soruna müdahale imkânını kolaylaştırmıştır. Robotun tekerleri ortada iki adet aktüatorlü tekerli ve robotun arkasında ve önünde iki adet bağımsız hareket eden teker bulunmaktadır. Bu tarz sistemler *nonholonomic* olarak adlandırılmaktadır. Bu robotlarda dikkat edilmesi gereken ağırlığın dengelenmesi ve ağırlığın robot merkezine verilmesidir. Bu şekilde yapılacak kinetik hesaplamalar kolaylaşacaktır. Robot şematiği Şekil 20'de gösterilmiştir.



Şekil 20. Tez Kapsamında Yapılan Mobil Robotun Teker Yerleşimi

Robotlarda kullanılacak gövdeye göre tekerlek tipleri ve tekerlek konumları da değişim gösterebilmektedir. Tekerlekler robotun köşelerinde dış kısımlarında olabileceği gibi robotun iç tarafına da yerleştirilebilmektedir. Tekerlerin konumu belirlendikten sonra dönüş için gereken hareket de belirlenebilecektir. Robot dönerken tekerler dönüş yapılacak yöne dönüp düz gidebilirler. Robotlarda tekerleri gidilecek yöne döndürecek mekanik sistemin bulunmaması durumunda farklı bir yöntem uygulanmaktadır. Bu yapıda birbirine karşılıklı bakan motorlar, tekerleri birbirlerine göre ters hareket ettirerek robotun yönü değiştirilebilir. Tekerin malzemesi ve boyutları önemlidir. Robotta kullanılan tekerler robotun gövdesini ve üzerinde bulunan tüm yükü sürekli taşımaktadır. Bu ağırlık altında kırılmadan, eğilmeden hareket etmesi gerekmektedir. Tekerler Şekil 23'de belirtildiği gibi dört ana tipe ayrılabilmektedir. Bu teker tipleri doğrudan robotun hareketine etki ettikleri için seçimlerine önem vermek gerekmektedir. Şekilde A ile ifade edilen teker standart tekerdir. Bu teker çoğu uygulamada kullanılır ve robotun doğru ilerlemesini sağlar. Bu tekeri kullanırken yönelme için farklı bir sistem geliştirilmesi gerekmektedir. B ile gösterilen teker tipi castor tekerdir. Castor tekerler robotun hareketine uyum sağlayarak kendi yönünü değiştirebilmektedir. C tipi tekerler sweedish olarak bilinmekte ve doğrudan robotun dönüş hareketini sağlayabilmektedir. Bu teker kendisi yönünü değiştirmeden robotu sağa, sola, ileri ya da geriye götürebilmektedir. D ile ifade edilen tekerler topludur. Robot nereye giderse gitsin o yöne doğrudan hareket

imkânı vermektedir. Bu tekerlerin hareketleri ve yönelmeleri için de farklı eylemci sistemlerin bulunması gerekmektedir.

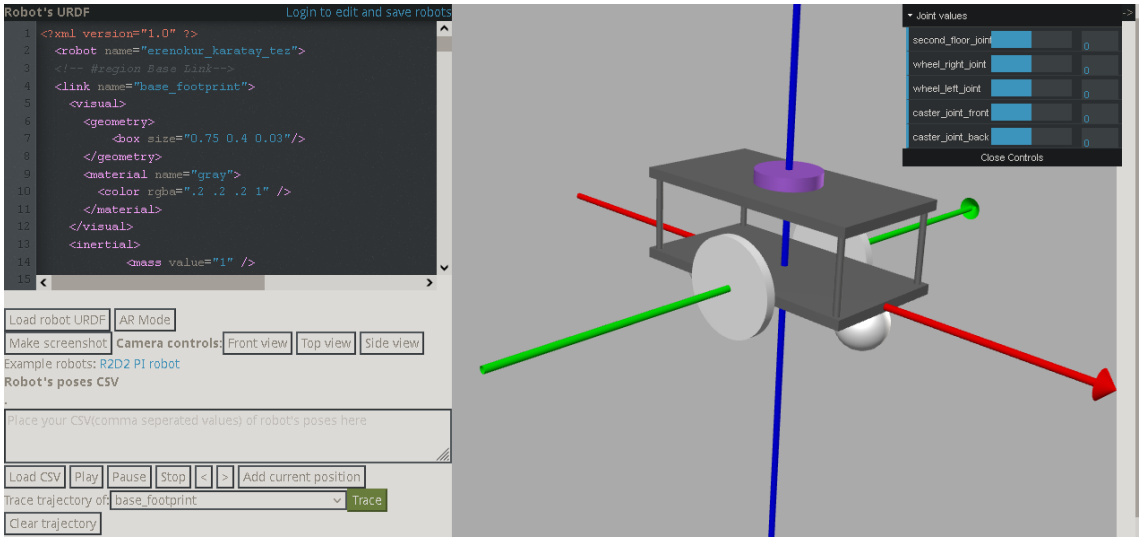


Şekil 21. Mobil Robot Teker Tipleri

(Kaynak: Siegward, 2014)

Projede kullanılan sarhoş tekerler Şekil 21'de belirtilen B tipi castor tekerlerdir. Bu tekerler robot hareketinde hemen en uygun konuma geçerek kendisini konumlandırmaktadır. Robotun üzerindeki yükü taşımakta ağırlık merkezinin dönmesini engellemektedirler. Yuvarlak tekere kıyasla daha büyük yüklere dayanabilmekte ve çoğu endüstriyel sistemde kullanılmaktadır.

Robotun tekerleri ve gövdesi seçildikten sonra hareketi için kinetik hesaplamaları yapılmalıdır. Bu sistemler ROS ile kullanılacaksa ROS için geliştirilen TF statik mesajları ayarlanmalıdır. ROS için kullanılan URDF dosyaları hazırlanmalıdır. Tez kapsamında yapılan robot için üretilen URDF dosyası Şekil 22'de gösterilmiştir.



Şekil 22. Tez Kapsamında Yapılan Robotun URDF çalışması

Bu işlemlerin yapılması ile robot hem yazılım sistemine kolaylıkla uyum sağlayabilecek hem de testlerde başarıya ulaşılacaktır. ROS'da kullanılan bazı araçlara geliştirilen TF ve URDF dosyaları doğrudan kullanılabilir. Bu işlemlerin sağlanması ile fiziksel robot ile yapılan testler hızlanacak ve kolaylaşacaktır. Bu sistemlerin olması SLAM ve navigasyon işlemleri için similasyonları da kolaylaştıracaktır. Gazebo ve RVIZ gibi GUI uygulamalarında yapılan URDF kullanılarak robotik işlemler kolaylıkla yapılabilecektir.

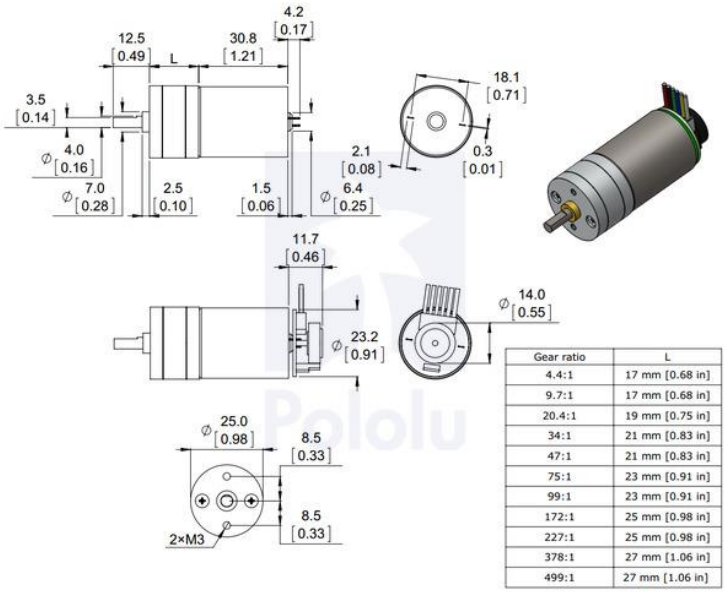
2.2.1. Robot Mekanik Parçaları

Robotun gövdesi bir çok mekanik parçadan oluşmaktadır. Bu parçalar gövdeyi oluşturan paneller, bu panelleri tutan destekler ve motor tekerlekleridir. Bununla birlikte sabitlemeyi sağlamak için çok sayıda sabitleyici ekipmanlar da bulunmaktadır. Robot ile ilgili en önemli kısım tüm yükü taşıyan alt gövdedir. Bu gövdede motorlar ve tekerlekler bulunmaktadır. Robot hareketleri bu bölgeden yönetilmektedir.

Tez projesi kapsamında yapılan robotta iki adet yüksek güçlü Pololu 2273 motor kullanılmıştır. Bu motor tipi 10.200 RPM hızına kadar çıkabilen motorlardır. Motorun sağlam bir şekilde sabitlenmesi robot hareketlerin doğru yapılabilmesi için önem taşımaktadır.

Tez çalışmasında kullanılan motorun çapı 25 mm dir. Ayrıca D şeklinde şaftı bulunmaktadır. Bu motorlara tip olarak 25 D x 64 L HP motor denilmektedir. Motorun

dişli takımı 34:1 verilmektedir. Motorun toplam uzunluğu 47,5 mm dir. Motorun teknik çizimi Şekil 23'te gösterilmiştir.



Gear ratio	L
4.4:1	17 mm [0.68 in]
9.7:1	17 mm [0.68 in]
20.4:1	19 mm [0.75 in]
34:1	21 mm [0.83 in]
47:1	21 mm [0.83 in]
75:1	23 mm [0.91 in]
99:1	23 mm [0.91 in]
172:1	25 mm [0.98 in]
227:1	25 mm [0.98 in]
378:1	27 mm [1.06 in]
499:1	27 mm [1.06 in]

Şekil 23. DC Motor Ölçüleri

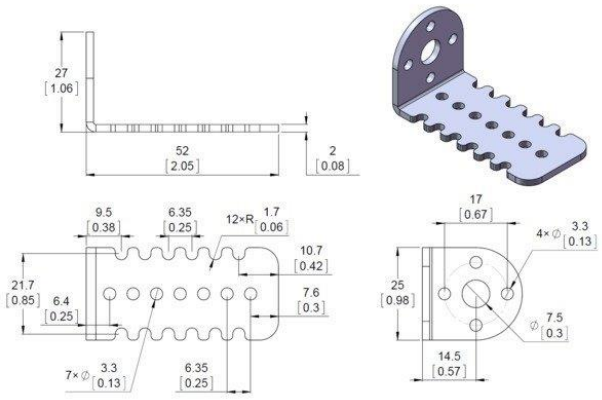
(Kaynak: Pololu, 2022)

Bu motora bağlı olan 48 CPR quadrature enkoder doğrudan motor şaftına bağlı ve kabloları takılı olarak gelmiştir. Motor ile arasında 11,7 mm mesafe olan enkoderin çapı 23,2 mm olarak ölçülmüştür. Enkoder motorun her tam tur dönüşünde 1632 sayım vermektedir.

Robot ilk yapıldığı zaman motor tutma aparatı kullanılmamıştır. Motor tutma aparatı yerine her motor için gövdede iki delik açılarak kablo sıkma aparatları ile sabitleme işlemi yapılmıştır. Bu şekilde bağlandığında motorlar ileri ve geriye doğru hareketlerde sabit durmuşlardır. Robot dönüş hareketi gerçekleştirirken motorlara robot içine ve dışına doğru basınç uygulanmaktadır. Bu basınç uygulandığı zaman motorlar içeriye doğru ya da dışarıya doğru hareket etmiştir. Ayrıca motorların yüksek güçlü ve yuvarlak bir yapıya sahip oması kablo bağları ne kadar sıkı bağlansa da motoru sabit tutmasına engel olamamıştır. Kablo bağı ile yapılan motor tutma denemesi işlemi Şekil 24'te gösterilmektedir.

Şekil 24. Motorların Kablo Bağı ile Bağlanma Deneyi

Motorlar için kullanılan kablo tutma aparatlarının yetersiz kalması nedeniyle robot motorlarını tutturmak için iki adet aparat kullanılmıştır. Bu aparatların alımı sırasında 25 D ve 47,5 boya uygun olamasına dikkat edilmiştir. Motor tutucu alınırken ayrıca motorun soğumasına yardımcı olması için hem hafif hem de kolay soğuyan alüminyum malzeme seçilmesine özen gösterilmiştir. Robot için Pololu markalı 2676 model 25 D motor tutucu seçilmiştir. Şekil 25'te motor tutucu aparat ile ilgili teknik çizim gösterilmiştir.



Şekil 25. Motor Tutucu Ekipman

(Kaynak: Pololu, 2022)

Motor tutucu ve motor 25 D olduğu ve motor vida delikleri ile tutucu delikleri 3,3 mm olduğu için motor tutucu motoru tam olarak tutabilmiştir. Motor hem vidalarla tutturulmuş, hem de motor tutucu ekipmana çift taraflı bant ile tutturulmuştur. Bu şekilde tüm yükün vidalara verilmesi engellenmiştir. Şekil 26'da görüldüğü gibi motor ekipman ile tam uyumlu şekilde yerleşmiştir.

Şekil 26. Motor Tutucu ve Motor

(Kaynak: Pololu, 2022)

Motor tutucu ve motor robot gövdesine hem üst taraftan hem de alt taraftan yapıştırılmıştır. Bu şekilde tüm yönlerden baskı uygulanmış ve motorun sabit kaldığı ve oluşacak elektrik kaçakları için gövdeden yalıtıldığı gözlenmenmiştir. Motorlar sabitlendikten sonra şaftlarının tipi D olduğu için uygun teker seçilmiştir. Teker içinde Pololu marka 63 model numaralı Tamilya 70145 dar teker modeli tercih edilmiştir. Tekerin boyutları 58 x 16 mm, şaftı yuvarlaktır. Teker bir çerçeve içinde satılmaktadır. Çerçevenin içerisinde bağlantı ekipmanları da bulunmaktadır. Tekerleğin çerçeve ve ekipmanları Şekil 27'de gösterilmiştir.

Şekil 27. Mobil Robot Tekeri

(Kaynak: Pololu, 2022)

Bu tekerler ile robot testleri yapılırken patinaj sorunları yaşanmıştır. Bu sorunların çözümü için öncelikle daha büyük yüz ölçümü olan silikon tekerler denenmiş fakat kullanılmak istenen silikon tekerler şafta uymamıştır. Geniş yüzey alanına sahip olan alternatif silikon teker Şekil 28'de gösterilmiştir.

Şekil 28. Denenen Yüzey Alanı Geniş Silikon Teker

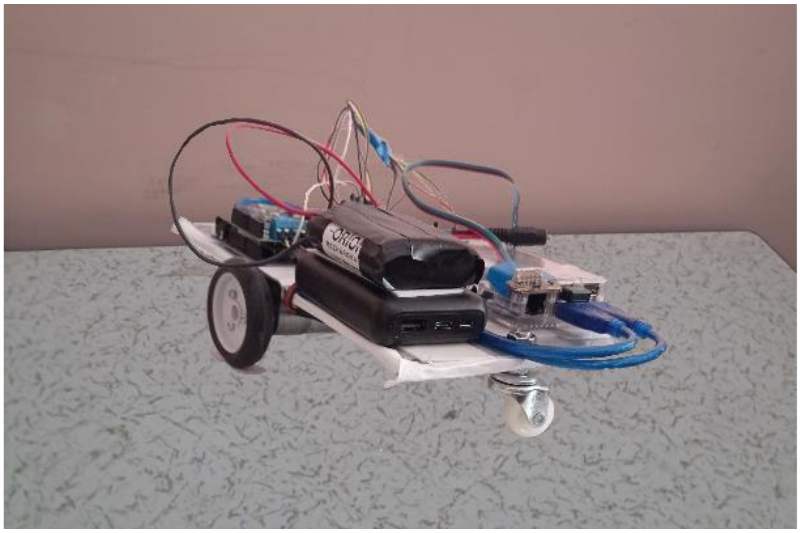
Türkiye pazarında 4 mm D şafta sahip başka bir teker bulunamadığı için bu motora D kesim 4 mm adaptör alınarak patinaj sorunu azaltılmıştır. Bu adaptörler üç adet M2 x 8 yıldız başlı vida ile tekere bağlanmıştır. Adaptör ile iglili resimler Şekil 29'da gösterilmiştir.

Şekil 29. Motor Şaft Adaptörü

Motorlara bağlı tekerlerin mekanik montajlaması tamamlandıktan sonra; robotun yük dengesini sağlamak amacıyla iki adet cıvatalı sarhoş teker kullanılmıştır. Sarhoş tekerler robotun önünde ve arkasında bulunmaktadır. Sarhoş tekerler robotun gittiği yöne kolayca kendiliğinden dönebildiğinden, robot hareketine olumlu katkı sağlamıştır. (Şekil 30)

Şekil 30. Mobil Robot Sarhoş Teker

Robot tabanı motorların ve sarhoş tekerlerin sabitlenmesi ile dengeye gelmiştir. Alt gövdesi üzerine Raspberry bilgisayar, bataryalar, ve mikroişlemciler yerleştirilmiştir. Robotun sensör ve mikroişlemcileri ile haberleşmesini sağlayan kablolar ve güç bağlantıları da bu bölümde toplanmıştır. Robotu taşıyacak kadar güçlü olan alüminyum kompozit malzemeye belirtilen parçaların montesi ile robotun alt gövdesi tamamlanmıştır. Şekil 31'de robot alt gövdesi gösterilmektedir.



Şekil 31. Geliştirilen Robotun Alt Gövdesi

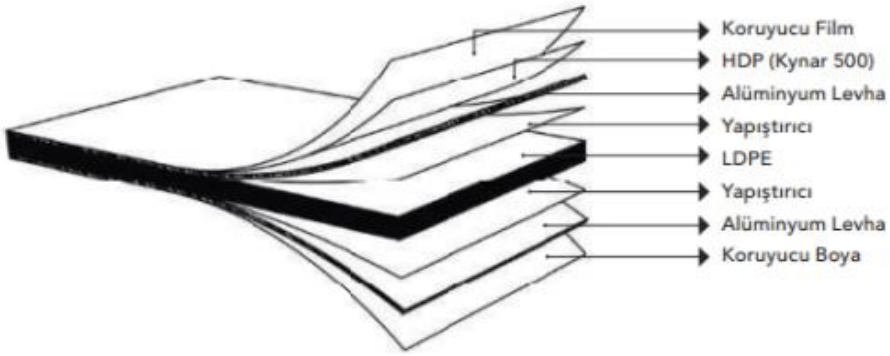
Robotun üzerindeki en büyük mekanik parça robotu taşıyan gövdedir. Bu gövde mümkün olduğunca hafif ama aynı zamanda ağırlığı taşıması için sağlam olmalıdır. Bu amaçla robot gövdesi için iki adet 27 x 15 boyutlarında aliminyum komposit malzeme kullanılmıştır. Bu materyalin içerisinde bir plastik türü olan polietilen kompozit malzeme bulunmaktadır. Bu komposit malzeme düşük yoğunluklu olduğu için Low-density polyethylene (LDPE) olarak adlandırılmaktadır. Komposit katmanın üst ve altında alüminyum levhalar bulunmaktadır. Malzeme kalınlıkları ;

Alüminyum üst levha kalınlığı : 0,3 mm

Alüminyum alt levha kalınlığı : 0,3 mm

Polietilen kompozit kalınlığı : 3,4 mm

şeklinde sıralanmaktadır. Panelde ayrıca koruyucu film, boya ve panelleri tutan yapıştırıcı da vardır. Malzemede kullanılan boya alüminyum için özel bir boya olup kolaylıkla çıkmamaktadır. Boya için Kynar 500 Bazlı HDP Boya kullanılmış olup kalınlığı 20 mikron olarak belirtilmektedir. (Şekil 32)



Şekil 32. Robotun Gövde Malzeme Yapısı

(Kaynak: Ronca, 2017)

Kompozitin üstünde iki adet alümiunyum levha materyalin çok daha sağlam olmasına imkân vermiştir. Ayrıca alümunyum plakalar boyama ile hem güzel bir doku hem de görünüm sağlamaktadır. Kompozit malzeme normalde çok dayanıklı olmayan plastikten yapılmaktadır. Plastik normalde kolay işlenebilir, hafif ve uzun ömürlüdür. Bu malzemenin farklı materyallerle karışması ile de çok daha dayanıklı olan kompozitler üretilmektedir. LDPE plastikler yüksek basınç altında sıkıştırılarak elde edilmektedir. Bu sıkışma sonucu iki karbon atomu ile dört hidrojen atomu kısa bağ ile bağlanmaktadır. Kompozitin dayanabileceği en yüksek sıcaklık 110 °C ve en düşük sıcaklık -75 °C olarak belirtilmektedir. (Ronca , 2017) Polimer özellikleri Tablo 5'de belirtilmiştir.

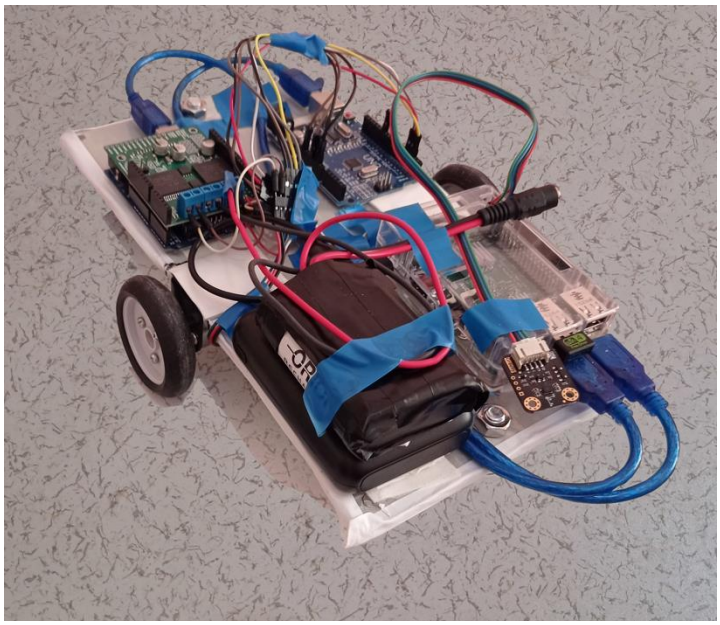
Tablo 5. Gövde Materyali (LDPE) Özellikleri Tablosu

Polimer	Kısaltması	Kimyasal yapısı	Minimum ısı	Maximum ısı
Düşük yoğunluklu polietilen	LDPE	$-CH_2CH_2-$	-75 °C	110 °C

(Kaynak: Ronca, 2017)

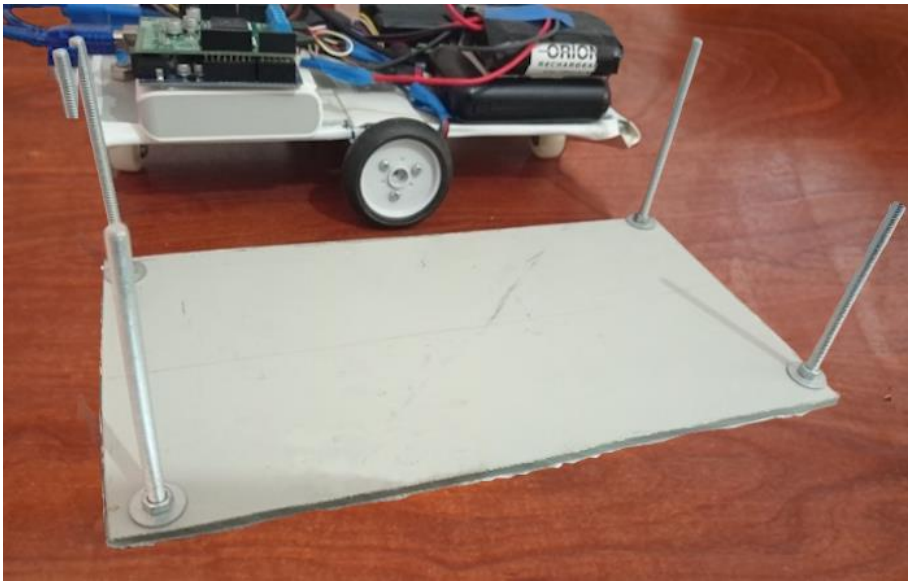
Alüminyum kompozit malzemenin hafif olması, kolay işlenmesi, dayanıklı olması ve kolayca soğuması sebebiyle bu malzemenin kullanılması tercih edilmiştir. Alüminyumlu levhalar spiral kesme taşları veya demir testeresi ile kolaylıkla kesilebilmektedir. Bu özelliği sayesinde ev ortamında bile istenilen boyutlarda kesilerek robot gövdesinde kullanılacak hale getirilebilmektedir. Ayrıca alüminyumlu levhalara M5 vidalar için gerekli delikler matkapla istenen genişlikte, kolayca ve hızla açılabilmektedir.

Alüminyum levhalar normalde elektriği iletebilmektedir. Bu iletkenlik robot üzerinde bulunan elektronik cihazlara temas etmesi hâlinde bu cihazlara zarar verebilecek ya da yanlış verilerin sisteme girmesine neden olabilecektir. Projede kullanılan alüminyum levhaların üzerinde koruyucu kılıf bulunmaktadır. Bu kılıf levhaya yapıştırılmıştır ve elektronik cihazlara zarar gelmemesi için gerekli izolasyonu sağlamaktadır. Alüminyum levhanın üzerindeki kılıf izolasyonu sağlamakta ancak kesilen kenarlarda bu kılıf bulunmamaktadır. Levha kenarlarının elektriksel olarak sorun çıkarmaması için izole bantlar ile robot gövdesi sarılmıştır. Kullanılan izole bantlar ayrıca gövdenin kesilmesi sırasında oluşan keskin yüzeylerden de robotun kablolarını korumaktadır. İzole bantlar çarpışmalarda robotun çevrede bulunan kapı, masa gibi diğer mobilyalara zarar vermesini, robotun kullanıcısının özellikle robotu kaldırırken ellerini de korumaktadır. Bu koruyucu bantlar Şekil 33'te paylaşılmıştır.



Şekil 33. Geliştirilen Robotun Kenar Kaplamaları

İki gövde parçasının birbirinin üzerinde olması ve sabitlenmesi için 4 adet M5 x 20 cıvata 4 adet sabitleyici somun kullanılmıştır. Robotun gövdesini oluşturan iki alüminyum levha birbirine paralel ve aralarında 20 cm somunlar bırakılarak yerleştirilmiştir. Robotun iki katlı tasarlanmasının sebebi robotun üzerine LIDAR sensörünü koyabilmek ve robotun iç kısımda elektronik devrelerin çarpışmalardan; elektronik ekipmanların sallanmadan, çarpmadan ve diğer fiziksel etkenlere karşı korunmasıdır. Alt gövdenin tüm elektronik ekipmanların ve kabloların rahatça sığacağı kadar geniş olması gerekmektedir. Robotun içindeki elektronik parçalara gerektiğinde robotun üst gövdesini sökmeden müdahale edebilmek için de iki levha arasında 20 cm lik bir boşluk bırakılmıştır. Bu mesafe ile robotun ortasına ve diğer noktalarına başka elektronik sistemlere zarar vermeden el girebilmekte, gerekli işlemler yapılabilmektedir. Robotun üst gövdesi Şekil 34'te gösterilmiştir.



Şekil 34. Geliştirilen Robotun Üst Gövdesi

Robotun iç kısımlarında bulunan enerji kablolarına oluşacak hataların çözümü ya da yeni özelliklerin eklenmesi için bazen bu bölgeye erişmek gerekmektedir. Bu nedenle kabloların bir yüzleri mümkün olduğunca robotun dış tarafına bakacak şekilde olması planlanmıştır. Robotta sürekli takıp çıkarılan bu kablolar LIDAR'ın güç kablosu, LIDAR'ın enerji kablosu, Raspberry Pi güç kablosu ve motor sürücü güç kablosudur. Bu kabloların dışarıdan çıkarılabilmesi için özellikle motor sürücüde tornavida kullanılması

gerekmektedir. Motor sürücüsünün üzerinde küçük bir tornavida girecek kadar yer bırakılarak işlem yapılması kolaylaştırılmıştır.

Robotun ikinci katını taşıyan cıvatalar robotun dört tarafında eşit mesafede ve sağlam şekilde sıkılması gerekmektedir. Bu işlem LIDAR çalışması için kritik önem taşımaktadır. Eğer LIDAR hareket sırasında titreşim ile karşılaşırsa ölçüm değerleri hatalı olacaktır. Cıvatalar eşit sıkılmazsa robotun üst kısmı eğik duracağı için yine doğru verinin elde edilmesi konusunda sorunlar yaşanacaktır. Bu nedenle tüm cıvatalar eşit sayıda sıkılmış, sağlamlıkları kontrol edilmiş ve su terazisi ile yatay ve dikey dengesi kontrol edilmiştir. Bu durum Şekil 35'te gös terilmiştir.

Şekil 35. Mobil Robotun Düz Şekilde Sabitlenmesi Kontrol Edilir.

Böylece bu işlemlerle birlikte sağlam ve dayanıklı bir robot gövdesi üretilmesi sağlanmıştır. Robot ile yapılan testlerde gövdenin sağlamlığı ya da titreşimi ile ilgili sorun yaşanmamıştır. Robot ilerleme ve dönüş işlemlerini başarıyla yapmakta, LIDAR üzerinden kullanılabilir veri alışverişi yapılmaktadır.

2.2.2. Robot Kinematiği ve ROS Statik Mesajları

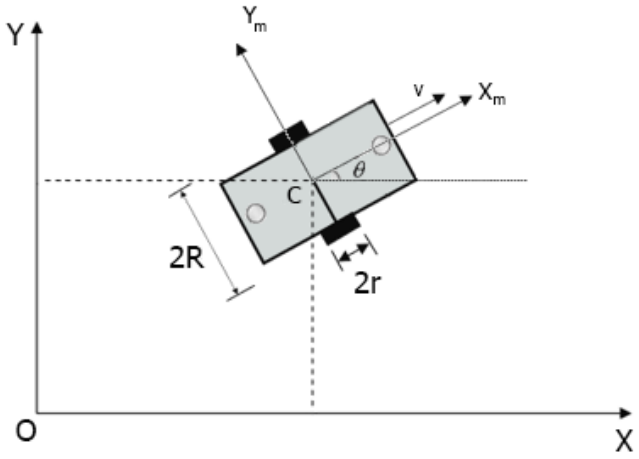
Kinematik, robotun hareketlerini kontrol ederek verilen emirlerin nasıl uygulanması gerektiğini belirlemektedir. Robota 100 metre ileri git emri verildiğinde robot mekanik olarak ileriye nasıl gideceğini bilmemektedir. Robotun nasıl ilerleyeceğinin hesaplanması için matematiksel olarak yapılan işlemlere kinematik işlemler denilmektedir. Robotun

ileri gidebilmesi için robota verilen değerler, robot mekanik aksamları ile kinematik hesaplarının yapılması gerekmektedir. Kinematik işlemleri hareketlerin yapılış şeklini etkilediği için çok önemli bir konudur. hesaplamalar robotun bulunduğu ortamda verilen emirleri gerçekleştirmesi için gereken açısal ve doğrusal hız bilgi modellerini oluşturarak işlem yapılmasını sağlamaktadır.

Kinematik, ileri ve ters olmak üzere iki ana kısma ayrılmaktadır. İleri kinematikte 3 eksenli bir düzlemde dönme hareketinin hesaplanması için kullanılmaktadır. Bu hesaplamada x ekseni etrafında dönmeye *yaw*, y ekseni etrafında dönmeye *pitch* ve z ekseni etrafında dönmeye *roll* denilmektedir. Uçan sistemlerde 3 eksenin 3'ü de bulunurken mobil robot gibi sistemlerde sadece 2 eksen üzerinden işlem yapılmaktadır. Ters kinematik ise eklem sayısının çok olduğu robot kol gibi sistemler için geliştirilmiştir. Test kinematik incelemeleri için sistem geometrik ve analitik olarak değerlendirilmektedir.

Tez kapsamında yapılan robot 2 eksen üzerinde hareket ettiği için ileri kinematik ile hesaplamaları yapılmaktadır. İleri kinetik hesaplamaları yapılırken tekerlekli robotlar holonomic ve nonholonomic olmak üzere ikiye ayrılmaktadır. Holonomic robotlar tüm eklemlerin kontrol edilebildiği robotlardır. Nonholonomic robotlarda sadece belirli eklem hareket ettirilirken tüm eklemler kontrol edilmez. Tüm eklemlerin doğrudan idare edilememesi nedeniyle nonholonomic robotlarda kontrol kısıtlı olmaktadır.

Tez çalışması için yapılan robotta iki adet kontrol edilebilir teker bulunmaktadır. Bu tekerler robotun sağ ve sol orta kısmında bulunup, idareleri bilgisayar üzerinden sağlanır. Robotun önünde ve arkasında ise iki adet doğrudan kontrol edilemeyen sarhoş teker bulunmaktadır. Sarhoş tekerler robotun hareketine göre kendisini konumlandırmaktadır. Robot ileri gittiğinde sarhoş teker öncelikle doğru pozisyona gelir ardından motora bağlı tekerlerin ilerlediği yönde ilerlemeye başlar. Robot dönerken sarhoş tekerler de robotun gittiği yöne uyumlu olarak dönüşlerini tamamlamaktadır. Robotun kinematik modellemesi Şekil 36'da gösterilmektedir.



Şekil 36. Geliştirilen Robotun Kinetik Modellemesi

Tez için geliştirilen robotta 2 adet Castor teker, iki adet de standart motora bağlı teker bulunmaktadır. Bu robotta motorlar kendi ekseni etrafında hareket edememektedir. Bununla beraber tekerleri de standart silikon teker olduğu için sadece ileri ve geri gitme özellikleri vardır. Robot sağa ve sola dönme işlemi yapabilmesi için iki teker arasında hız farkı oluşturmak zorundadır. Bu şekilde iki motorun hız farkı ile dönüş işlemini gerçekleştiren robotlara *unicycle mobil robot* denilmektedir.

Robotun tüm eklemleri kontrol edilemediği ve tekerleklerinden dolayı belirsizlikler oluştuğu için robot kinematik hesabını birinci dereceden lineer olmayan yöntemle yapılır. Robot kinematiği hesaplanırken zaman (t) değerinin her zaman sıfır değerinden büyük olduğu bilinmelidir. Robotun t anındaki lineer hızı v(t), açısal hızı w(t) ve düzlem orjinine göre konumu Q(t) olarak gösterilmektedir. Robotun orta noktasının düzlemdeki koordinatları x(t) ve y(t) olarak verilmiştir. Robotun kinematik hesapları yapabilmesi için giriş hızlarının robot hızlarına çevrilmesi gerekmektedir. Bu çevirme işleminin yapılması için Jacobian matrisleri S(p) kullanılmaktadır. Açısal ve doğrusal hız matrisi ise $v_k(t)$ olarak gösterilmektedir. Yapısal değişimler ve modellenmemiş dinamikler ΔS(p) ile gösterilmektedir. Oluşacak dalgalanma değerleri d(t) değeri olarak verilmektedir. Birinci derece kinematik vektörü ṗ(t) jakobian matrisi ile hızların çarpılması ve modellenmemiş dinamiklerle hızların çarpılması sonucu çıkan değerlerin dalgalanma değeri ile toplanarak bulunmaktadır. Bu durum Denklem 9'da gösterilmiştir.

$$\dot{\mathrm{p}}(\mathrm{t}) = \mathrm{S}(\mathrm{p}) * v_k(t) + \Delta\mathrm{S}(\mathrm{p}) * v_k(t) + d(t) \tag{9}$$

Formülde verilen Jakobian matrisi S(p) robot oryantasyon değeriyle oluşturulan bir matris ile belirtilmektedir. Bu matriste (1,1) konumunda cosθ(t), (2,1) konumunda sinθ(t) ve (3,3) verisinde 1 olarak verilmiş kalan veriler 0 olarak yerleştirilmiştir. ΔS(p) de ise matrisin (3,3) elemanı Δ olarak belirtilmiştir. Denklem 9 vektör bilgileri ile tekrar yazıldığında Denklem 10 elde edilmektedir.

$$\dot{\mathrm{p}}(\mathrm{t}) = \begin{vmatrix} \cos\theta(\mathrm{t}) & 0 \\ \sin\theta(\mathrm{t}) & 0 \\ 0 & 1 \end{vmatrix} * \begin{vmatrix} \mathrm{v}(\mathrm{t}) \\ \mathrm{w}(\mathrm{t}) \end{vmatrix} + \begin{vmatrix} \cos\theta(\mathrm{t}) & 0 \\ \sin\theta(\mathrm{t}) & 0 \\ 0 & 0 \end{vmatrix} * \begin{vmatrix} \mathrm{v}(\mathrm{t}) \\ \mathrm{w}(\mathrm{t}) \end{vmatrix} + d(t) \tag{10}$$

Robotun birim zamandaki konumunu ifade eden p(t) değeri robotun x, y ve açı değerlerinden oluşmaktadır. Bu durum Denklem 11'de ifade edilmiştir.

$$\mathrm{p}(\mathrm{t}) = [\mathrm{x}(\mathrm{t})\ \mathrm{y}(\mathrm{t})\ \theta(\mathrm{t})]^T \tag{11}$$

Kinematik hesaplamalarında genellikle belirsizlikler ve düzensizliklerin birbirine bağlı olduğu düşünülmektedir. Bu değerler sadece pozitif değerleri ifade etmektedir Bu bilgiler ile sistemin konum ve pozisyon hata verisine göre tekrar düzenlendiğinde $e_x(t)$ $e_y(t)$ $e_\theta(t)$ verileri elde edilmektedir. Bu verilerle de kinematik kontrol değeri olan $q_e(t)$ değeri elde edilmiştir. Kinematik kontrol değer Denklem 12 ile gösterilmektedir.

$$\mathrm{p}_e(\mathrm{t}) = \begin{vmatrix} \mathrm{e_x}(\mathrm{t}) \\ \mathrm{e_y}(\mathrm{t}) \\ \mathrm{e_\theta}(\mathrm{t}) \end{vmatrix} = \begin{vmatrix} \cos\theta(t) & \sin\theta(t) & 0 \\ -\sin\theta(t) & cos\theta(\,t) & 0 \\ 0 & 0 & 1 \end{vmatrix} * \begin{vmatrix} x_r(t) & -x(t) \\ y_r(t) & -y(t) \\ \theta_r(t) & -\theta(t) \end{vmatrix} \tag{12}$$

Kinematik kontrol denkleminden sistem için gereken hızları bulabiliriz. Hız bulma işlemi için öncelikle Denklem 9 üzerinde türev alınması gerekmektedir. Türev işlemi tamamlanınca Denklem 13'de belirlenen pozisyon hata değeri elde edilir.

$$\dot{\mathrm{p}}_e(\mathrm{t}) = \begin{vmatrix} \dot{\mathrm{e}}_{\mathrm{x}}(\mathrm{t}) \\ \dot{\mathrm{e}}_{\mathrm{y}}(\mathrm{t}) \\ \dot{\mathrm{e}}_{\theta}(\mathrm{t}) \end{vmatrix} = \begin{vmatrix} -v(t) + w(t)e_y(t) + v_r(t)cose_\theta(t) \\ -w(t)e_x(t) + v_r(t)sin\, e_\theta(t) \\ -w(t) + w_r(t) \end{vmatrix} \quad (13)$$

Doğrusal hız $v_r(t)$ ve açısal hız $w_r(t)$değerleri denklemden çekildiğinde Denklem 14 ve Denklem 15 elde edilmektedir.

$$v_r(t) = \sqrt{\dot{x}_r^2(t) + \dot{y}_r^2(t)} \quad (14)$$

$$w_r(t) = \frac{\dot{\mathrm{x}}_r(y) * \ddot{\mathrm{y}}_r(t) - \dot{\mathrm{y}}_r(t)\ddot{\mathrm{x}}_r(t)}{\dot{\mathrm{x}}_r^2(t) + \dot{\mathrm{y}}_r^2(t)} \quad (15)$$

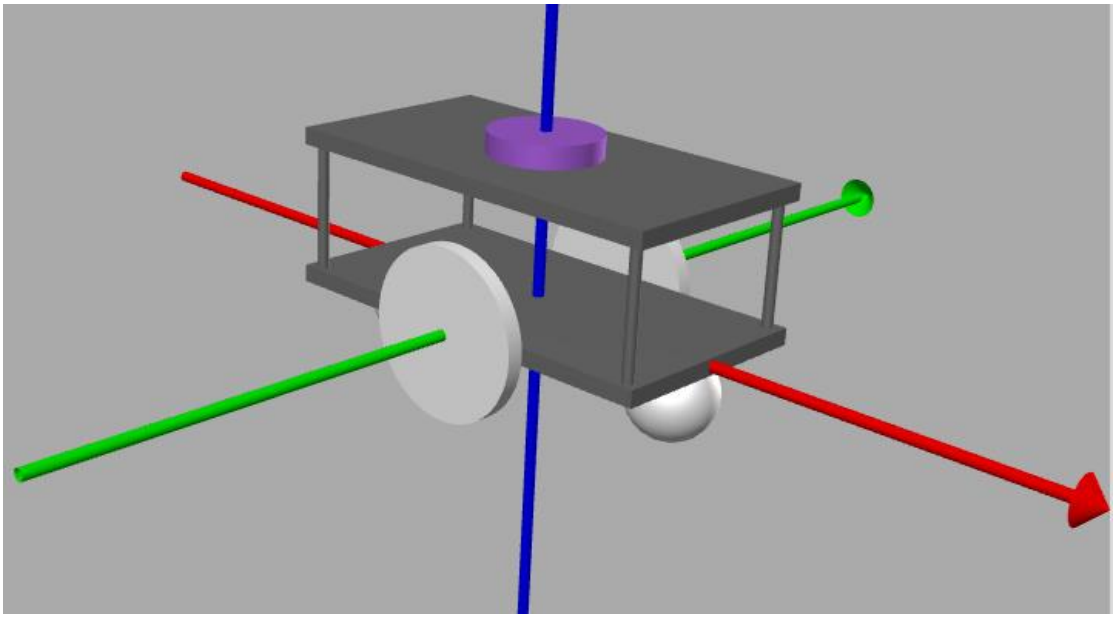
ROS sistemleri ile geliştirilen teknikler kinetik hesaplamaların yazılım tarafından yapılmasına olanak sağlamaktadır. ROS mobil robotlar için normalde olduğu gibi ileri kinematik hesabı yapmaktadır. ROS burada gerekli parametreleri alarak işlemleri kendi içinde otomatik yaptığı için işlemlerde kolaylık sağlamaktadır.

ROS ile kinematik hesaplamalarını yapmak oldukça kolaydır. Robotun proje dizini üzerine eklenecek konfigürasyon dosyaları proje launch dosyasında çağrılması ile otomatik hesaplamalar başlamaktadır. ROS bu konfigürasyon dosyalarının yanı sıra URDF dosyalarını da okuyup anlamlandırabilmelidir. Konfigürasyon parametreleri verilirken ROS paket adı olarak TF verilmektedir. Belirtilen argumanların neresi için gerektiği de type parametresi ile belirtilmiştir. Paremetreler ile verilen TF argumanları Şekil 37'de paylaşılmıştır.

```
<launch>

<!-- All sensors will be published on robot -->

  <!-- Map File -->
  <arg name="map_file" default="$(find erenokur_thesis_launcher)/maps/my_home_map.yaml"/>

  <!-- Map Server -->
  <!-- Publish: /map -->
  <node pkg="tf" type="static_transform_publisher" name="map_to_odom_broadcaster" args="0 0 0 0 0 0 map odom 20" />
  <node pkg="map_server" name="map_server" type="map_server" args="$(arg map_file)" />

  <!-- Costmap File -->
  <!-- Subscribe: /map -->
  <!-- Publish: /costmap_2d/costmap/costmap -->
  <node pkg="tf" type="static_transform_publisher" name="base_link_broadcaster" args="0 0 0 0 0 0 base_footprint base_link 20" />
  <node pkg="navigation_data_pub" type="tf_odom_to_base" name="tf_odom_to_base">
  </node>
```

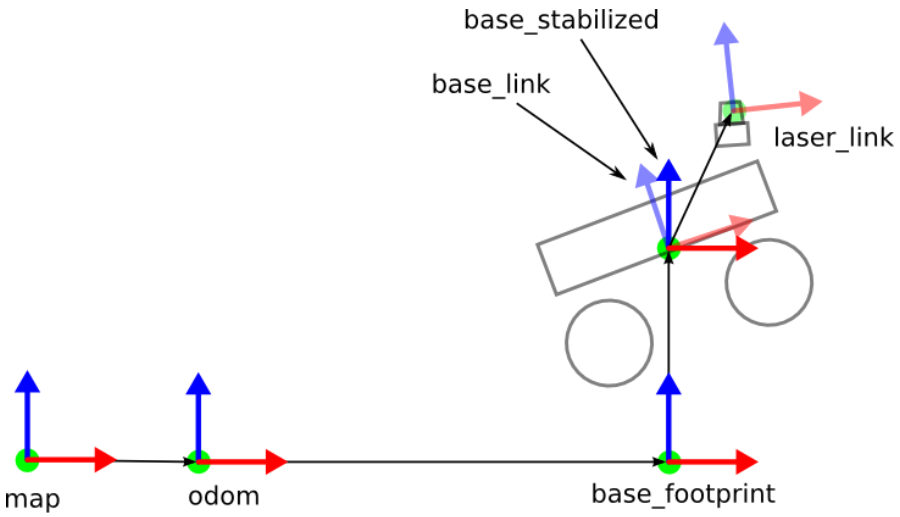
Şekil 37. Tez Robotunun TF Argümanları

ROS TF sistemi için parametreleri vermenin bir diğer yolu da URDF sistemleridir. URDF oluşturulurken tüm veriler doğru şekilde girilirse ve yine launch dosyasının içine URDF dosyasının yolu verilmelidir. URDF dosyaları endüstriyel uygulamalarda genellikle SOLID gibi mekanik çizimlerin yapıldığı programlarda oluşturulmaktadır. SOLID üzerinde oluşturulan çizimler hem CNC gibi makineler ile yapılabilmektedir. Bu çizimlerin URDF'leri çıkartılarak ROS üzerinde TF bilgileri yayınlanabilmekte ve simülasyon uygulamalarında görsel olarak robotun çalışmasına olanak verilebilmektedir. Tez robotunun URDF görseli Şekil 38'de paylaşılmıştır.



Şekil 38. Tez Robotu URDF Görseli

TF verileri robot_state_publisher paketi ile toplanarak bir ağaç şeklinde şekillendirebilmektedir. Bu ağaç içerisinde robotta kullanılan LIDAR, IMU, sağ ve sol enkoder sensör verileri yer almaktadır. Robotun hareket ettiği düzleme göre bulunduğu konuma base_footprint adı verilmektedir. Robotun tekerleri nedeniyle asıl bulunduğu konum base_footprintten biraz daha yukarıda olacağı için bu yeni noktaya base_stabilized denilmektedir. Base stabilized hesaplanırken robotun teker çapı bilgisine ihtiyaç duyulmaktadır. Robotun base_stabilized konumu robotun eğimi hakkında bilgi verememektedir. Robot eğer bir engelle karşılaşırsa gövdesi yukarı kalkabilecek ve robot eğik durabilecektir. Robotun eğimini ve asıl konumunu gösteren son değere de base_link adı verilmektedir. Map tanım noktası robotun hareket ettiği düzlemde bulunan herhangi bir noktadır. Odom verisi ise robotun ilk olarak tanımlandığı noktayı ifade etmektedir. Bu nokta robotun tüm hareket dünyası içinde her zaman tek olarak bulunmaktadır. Bu verilerin 2D bir robot üzerinde gösterilmesi ile Şekil 39'daki grafik elde edilmektedir. Grafikte mavi ok ile gösterilen eksen x ekseni olup kırmızı y eksenini ifade etmektedir. Mor ve turuncu renkler robotun eğiminden kaynaklanmakta olup x,y eksenlerine göre eğimi temsil etmektedir. (ROS, 2022)



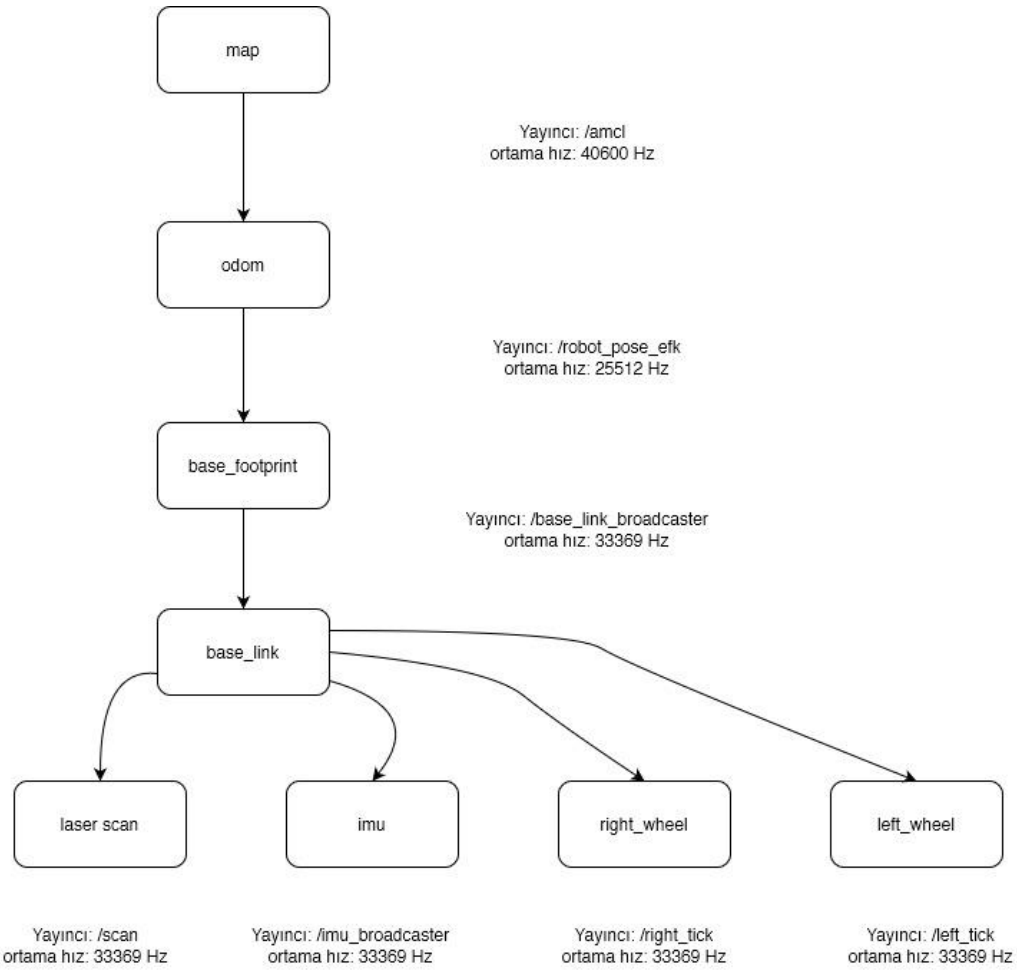
Şekil 39. Robotun 2D Düzlemdeki Linkleri

(Kaynak: ROS, 2022)

ROS üzerinde robot trajectory ağacı oluşturulurken en uçta yer alan sensör bilgileri en alttan başlatılır. Robot ilk önce üzerinde bulunan sensörleri okumakta ve bu verileri

base_link verisiyle birleştirmektedir. Bu veride robotun y ekseni ile kesişen base_footprint bağlantısı ardından sırası ile odom ve map bağlantıları ile devam ettirilir.

ROS bu hesaplamaları sürekli yaparak robotun odometri verisini ve kinematiğini hesaplamaktadır. ROS ile oluşturulan ağaç yapısında ROS'un bu işlemleri hangi paket ile yaptığı, ortalama hızının ne olduğunu, bufer uzunluğunu ve yaptığı transform işlem zamanını belirtmektedir. Robot için geliştirilen ağaç yapısı Şekil 40'da belirtilmiştir.



Şekil 40. Tez Robotu Link Ağaç Yapısı

Map verisi statik bir veridir. Bu veri istenirse robotun başlangıç konumunu belirten odom verisiyle eşleştirilebilmektedir. Odom ile base_footprint arasındaki robot hareket ettiği için sürekli değişen bir bağlantı bulunmaktadır. Base_footprint ile base_link ise robot aynı eksende hareket ettiği sürece statik bir bağlantıya sahiptir. Robotun sensörleri ile

olan ilişkisi de statik olarak sayılmaktadır. Bunun sebebi robot hareket ederken sensörlerin ve base_link değerinin de hareket ediyor olmasıdır.

Mobil robot için verilen TF'ler ile ilgili verilerin hesaplanması sonucu haritalama ve navigasyon işlemleri yapılabilmektedir. Robot statik mesajları sürekli olarak yayınlanmakta ve ROS'ta bulunan paketler sayesinde gerekli tüm işlemler hızlı şekilde yerine getirilebilmektedir.

2.3. Haritalama ve Konumlandırma Yazılım Sistemleri

Haritalandırma ve konumlandırma işlemleri yazılım mimarisi de iki ayrı sisteme ayrılmıştır. Mobil robotun üzerindeki tüm sensörlerin okunması ve motorlara emirlerin verilmesi işlemleri Raspberry Pi 4 mini bilgisayara verilmiştir. Çalışma esnasında gerekli olan algoritmaların asıl çalışacağı server bilgisayarı ise ASUS N550JK olarak seçilmiştir. Raspberry Pi küçük boyutları olan düşük donanım ve enerji gereksinimi duyan bir bilgisayar olduğu için robotun üzerinde yapılacak işlemler için çok daha etkin çalışabilecektir. Server bilgisayarı üzerinde ekran kartı, güçlü bir işlemci ve yüksek kapasiteli RAM olan bir bilgisayardır. Enerjisini doğrudan şebekeden alabilmektedir. Bu özellikler sayesinde server kendisine verilen karmaşık işlemleri kolaylıkla ve hızlı şekilde yapabilecektir. Ucuz, hafif ve az enerji tüketen bir bilgisayar seçilmesi sayesinde mobil robotun maliyeti ve etkin çalışması konularında avantaj sağlanmıştır.

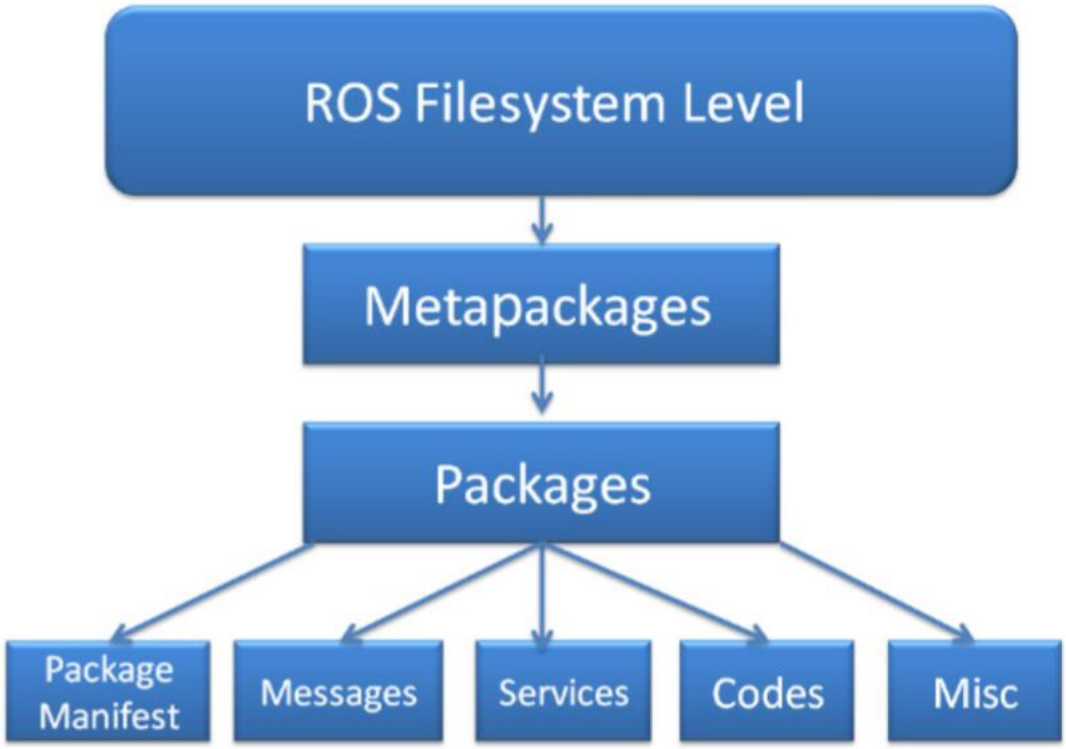
ROS robotik sistemleri, geliştirilen çeşitli uygulamaların bir araya getirilerek bir standart ile kullanıcıların geliştirilmesine olanak sağlayan yazılımlar bütünüdür. ROS 2007 yılında Morgan Quigley tarafından başlatılmış daha sonraki süreçte gelişimini Willow Garage laboratuvarında sürdürmüştür. Günümüzde ise ROS genel olarak Openrobotics Firması tarafından geliştirilmektedir. Openrobotik Firması ROS'u geliştirirken sadece kendi yazılımlarını dikkate almamış, genel olarak herkesin kullanabileceği bir sistem tasarlamıştır. Dünyanın her yerinde robotik sistemlerle ilgilenen şirketler ve yazılımcılar yazılımlarını ROS standartlarına uygun olarak geliştirmeleri durumunda kendi sistemlerini müşterilerine ve robotikle uğraşan diğer meslektaşları ile paylaşabilmektedir. Aynı şekilde kendileri de farklı geliştiricilerin ya da şirketlerin geliştirdikleri yazılımları kullanabilmektedirler. Bu şekilde robotik sistemlerin gelişmesi çok daha hızlanmış ve bir standarda bağlanmış olmaktadır. ROS ile kullanımı kolay frameworkler kullanılabilmektedir. Bu frameworkler ile normalde çok zor olan kinematik

hesaplamaları, haritalandırma, konumlandırma, hareket planlamaları ve navigasyon gibi konuların daha kolay yapılmasına olanak vermektedir. Bu frameworklerin günümüzde en çok kullanılanları SLAM, AMCL, MoveIt olmakla beraber çok sayıda alternatifleri de bulunmaktadır. ROS ile beraber yazılımcıların kullanımına çok sayıda hesaplama yapan ve görsel arayüze hitap eden yazılımlar da geliştirilmiştir. Bu yazılımlar kendi başlarına da çalışabilmekle beraber ROS ile çok daha etkin kullanım sağlamaktadır. ROS ile gelen görsel simülasyon araçlarına örnek olarka RVIZ ve Gazebo uygulamaları verilebilir. ROS ayrıca donanım geliştiricilerin ve yazılım topluluklarının desteği sayesinde çok sayıda sensörü ve eyleyiciyi kolaylıkla kontrol etmeye olanak vermektedir. Tüm bu avantajları modüler şekilde vermesi sayesinde yazılımcı istediği zaman istediği yazılım parçasını açabilmekte ve rahatlıkla kullanabilmektedir. Bu modülerlik çoklu thread yönetimlerinde de yazılımcıların önünü açmış müşteri kullanıcı ilişkisi ile kurulan threadlerin birbirleri ile çatışma olasılıkları farklı uygulamalarda oldukları için azalmıştır. Bir threadin çökmesi tüm sistemin yok olması ile sonuçlanmamakta sadece ilgili sistem etkilenmektedir. ROS'un bu modüler yapısına karşılık olarak haberleşme sistemlerinde de gelişme gösterilmiştir. ROS 1 ve ROS 2 de farklı çözümler olmasına rağmen ROS haberleşme sistemleri genel olarak modüler sistemlerin sorunsuz haberleşmesine olanak sağlar ve kullanımı kolaydır. Tez çalışmalarında genel olarak ROS 1 esas alınmış ve bu sistemin haberleşme modülleri ve sistemleri kullanılmıştır.

ROS'un sürekli gelişmesi ile sağlanan bu kolaylıklar sayesinde robotik topluluğu gün geçtikçe çoğalmakta ve robotların gelişim hızı artmaktadır. ROS topluluğu kendi ihtiyaçlarına göre ROS standartları ile yeni yazılım paketlerini çıkardıkları için aynı sorunları yaşayan farklı geliştiriciler için de uygulanabilmektedir. Topluluğun artması internetteki soruların ve cevapların artmasına bu da genel bilgi düzeyinin yükselmesine olanak sağlamaktadır. SLAM gibi karmaşık konularda eskiden yayınlanan karmaşık dokümanların yanına artık standart kullanıcıların robotları için sordukları her çeşit soru ve cevaplar eklenmiş, araştırma yapanların işleri kolaylaşmıştır. Ayrıca günümüzde oldukça popüler olan teknoloji formları ve bloglarında çok sayıda kılavuz yayınlanmakta ve örnek projeler paylaşılmaktadır. ROS'un gelişmesi büyük şirketlerin de dikkatini çekmektedir. Microsoft gibi dünyanın her yerinde pazar payı olan şirketler kendi işletim sistemlerinde ROS çalışmasına imkân vermek için çalışmalar yapmaya başlamıştır. Tüm bu gelişmeler ışığında ROS her geçen gün gelişimini çok daha hızlı sürdürmekte robotlar

için geliştirdiği yazılım ve çözümlerle robotik sistemlerin standartlarını oluşturmada öncülük etmektedir.

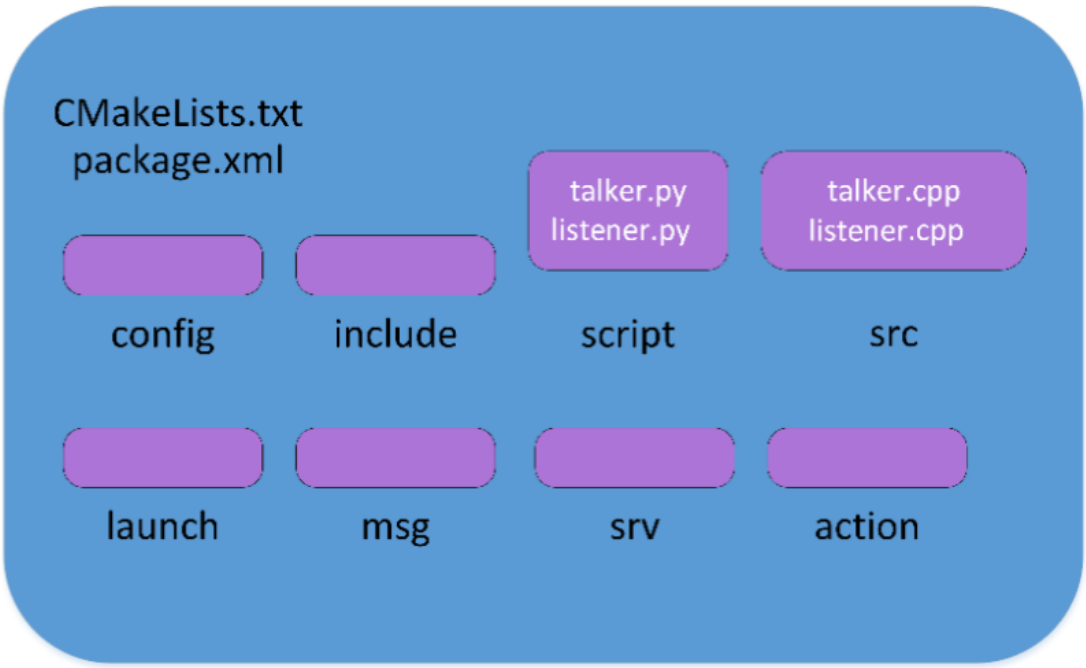
ROS bir yazılımdan çok bir işletim sistemi gibi davranmaktadır. ROS'un sahip olduğu dosya hiyerarşisi ve sisteminde paketler, servisler, mesajlar, kendi derleyicisinin bulunması, kendi sorun giderme araçlarının bulunması özellikleri sayesinde kullanıcılar istedikleri işleri kolaylıkla yapabilmektedir. Şekil 41'deki en alt kısımda ROS dosya sistem seviyesi vardır. Bu sistem içinde bir veri taşımayan sanal paketler taşıyan metapaketler barındırır. Bu metapaketlerden ROS'a bağlı yazılımlar, kütüphaneler ve konfigürasyonlar gibi tekil sistemler oluşmaktadır. Bu paketlerde ROS içinde bulunan paketler tanımlarına, mesajlarına, servislere, yazılımlara vb.işlere göre dağılmaktadır.



Şekil 41. ROS Paket Yapısı

(Kaynak: Joseph, 2021)

Paket yapıları ve sistemleri ROS'un temelinde olduğu için en önemli konulardan sayılmaktadır. En genel kullanılan paket sisteminde 10 adet yapı bulunmaktadır. Bunlardan ilki config dosyalarıdır. Config dosyaları kullanıcılar ve yazılımcılar tarafından doğrudan değiştirilebilmekte, bağlı olduğu yazılımın nasıl davranması gerektiğinin ayarlandığı yazılım ayar dosyasıdır. Include dizininde yer olan paketler özellikle C++ dili için header dosyalarını ve kütüphaneleri oluşturmaktadır. Script dizininde yer alan paketler doğrudan derlenmeden çalıştırılabilen Python gibi programlar için kullanılmaktadır. Src dizini C++ programlarının kaynak kodları için ayrılmıştır. Bu dizindeki kodlar cpp ile derlenerek doğrudan kullanılamamaktadır. Launch dizini yazılımların başlatıldığı launch dosyalarının olduğu dizindir. Bu dizindeki launch dosyaları doğrudan terminal üzerinden tek komutla çağrılır. Çağrıldığında içlerinde bulunan çeşitli yazılımları yine launch dosyasında belirtilen konfigürasyon dosyaları ya da parametreleri ile çalıştırır. Msg dosyasının içinde yazılımların etkin çalışması için geliştirilmiş çeşitli mesaj tanımları bulunmaktadır. ROS ön tanımlı paketleri kullanılacaksa bu dosyaya gereksinim duyulmamaktadır. Bu yazılımcıların uygulamaları, kendilerine gereken özel paketler için kullanılmaktadır. SVR dosyasında yazılımcılar haberleşme sistemilerini servis olarak tanımlamışlarsa onların oluşturdukları tanım dosyaları bulunmaktadır. Action dosyasında yazılım sisteminin aksiyonlardan oluşturulması gereken yazılımlar için kullanılmaktadır. ROS proje dizininde bulunan package.xml dosyası içinde op projenin ne için olduğu, kimin yaptığı gibi tanıtım bilgileri bulunmaktadır. Paketler dosyasında proje için gerekli derleme ve çalışma gereksinimleri ile araçları da belirtilmektedir. CMakeList.txt yine ROS proje dizinin de bulunduran oldukça önemli bir pakettir. Bu paket sayesinde hangi yazılımın nasıl derleneceği bilgisi catkin_make'e verilmektedir. Bu dosyanın konfigürasyonunda proje ya da bu dosyanın çalışma şekli hakkında tam olarak bilgi sahibi olmadan değişiklik yapıldığı takdirde projede derleme hataları, mesajların tanınmaması, servislerin ya da aksiyonların çalışmaması gibi sorunlar oluşacağı için dikkatli kullanılması gereken bir dosyadır. Bu dosya yapısı Şekil 42'de gösterilmektedir.



Şekil 42. ROS Derleme Paketi (CMakeList) Yapısı

(Kaynak: Joseph, 2021)

ROS kendi üzerinden çalışan sistemlerde C++ ve Python yazılım dillerini doğrudan desteklemektedir. Bu yazılım dillerini kullanarak ROS catkin paketi oluşturulabilir. ROS'un sağlamış olduğu catkin_make derleyicisi ile programlar derlenebilir. Derlemek için gerekli konfigürasyonlar catkinmake dosyasında tutulmaktadır. ROS bu işlemlerin yapılması için gerekli standartları ve dokümantasyonu hazırladığı için geliştiriciler kolaylıkla kendi paketlerini oluşturup derleyebilmektedir. ROS ile farklı yazılım dillerinde geliştirme yapmak isteyen kullanıcılar için de ROS bridge yazılımları geliştirilmiştir. Bu yazılımlara gerekli paremetreler verildiğinde farklı dillerden gelen topik açma ve takip etme istekleri karşılanabilmektedir. ROS ile geliştirilen modüller istenirse ROS dışındaki programlarla da haberleşme sağlayabilmektedir. Bu konuda ROS herhangi bir engel getirmemiştir.

ROS haberleşmek için üç ana unsur kullanmaktadır. Bu unsurlar mesaj yapıları, servisler ve olay yapılarıdır. ROS topikleri ROS'un sağladığı en büyük katkılardan biridir. Bir çok yazılım sistemi bu topikler üzerinden işlemlerini kolaylıkla yapabilmektedir. ROS topiklerine erişimin kolay olması, izlenebilir olması, basitliği, fazla hata yapmaması sayesinde bir standart hâline gelmiştir. ROS topikleri yazılımlarda kullanmak için sadece

takip edilecek topik adı takipci olarak programa tanıtılır; sonrasında gelecek paketler için bir olay oluşturularak topikteki paketlere erişilir. Bu işlemi bilgisayar terminalinden de yapılabilmesi son kullanıcıların da bu paketlere kolaylıkla erişmesini sağlamaktadır. Terminal üzerinden "rostopic list" komutu yazıldığı zaman ROS üzerinde çalışan tüm topikler listelenmektedir. ROS topikleri bulunduktan sonra tekrar terminal üzerinden rostopic echo topik adı yazıldığında veriler canlı olarak izlenebilmektedir.

ROS topikleri paket sistemi kendi mesaj sistemini kullanmaktadır. ROS Mesajlarının oluşturulması ve kullanılması olayları ROS üzerinden kolaylıkla yapılabilmektedir. Topiklerin üzerinden yapılan işlemler için ROS'un belirlediği standartlar kullanılmalıdır. Bu standartlar endüstri standartlarına uyduğu için hem farklı yazılımların bu mesajları kullanmasını kolaylaştırır hem de mesajı üreten yazılım ekibi için kolaylık sağlar. Ros paket tipleri Tablo 6'da gösterilmiştir.

Tablo 6. ROS Paket Tipleri

Tipi	Değeri	C++	Python
bool	Unsigned 8-bit int	uint 8_t	bool
int8	Signed 8-bit int	int8_t	int
uint8	Unsigned 8-bit int	uint8_t	int
int16	Signed 16-bit int	int16_t	int
uint16	Unsigned 16-bit int	uint16_t	int
int32	Signed 32-bit int	int32_t	int
uint32	Unsigned 32-bit int	uint32_t	int
int64	Signed 64-bit int	int64_t	int
uint64	Unsigned 64-bit int	uint64_t	int

float32	32-bit IEEE float	float	float
float64	64-bit IEEE float	double	float
string	Ascii string	std::string	string
time	Sec/nsec unsigned 32 bit ints	ros::Time	rospy.Time
duration	Sec/nsec signed 32 bit ints	ros::Duration	rospy.Duration

(Kaynak: Joseph, 2021)

ROS un belirlediği mesaj tipleri int8, int16, int32, int64, uint, float32, float64, string, time, duration gibi bilgilerdir. Bu bilgilerin sınıflandırılması için istenildiği gibi veri dizisi işlemleri de yapılabilir. İstenirse kullanılan paket tipine göre mesaja header da eklenebilmektedir. Bu header ile gönderilen paket tipi bilgisi paylaşılabilmektedir.

Example .msg file

Header header

string child_frame_id

geometry_msgs/PoseWithCovariance pose

geometry_msgs/TwistWithCovariance twist

Bu mesaj tipleri rosmsg show paket adı boşluk mesaj tipi şeklinde kullanılabilmektedir. Bu bilgiye topik ile de erişilebilmektedir. Topik üzerinden erişim için rostopic type topik ismi boşluk | boşluk rosmsg show ile izlenebilmektedir.

Eğer yazılımcılar ROS sistem mesajları ile işlem yapamayacaklarını düşünürlerse ROS burada herhangi bir kısıtlama getirmediği için kendi paketlerini de oluşturarak kullanmaya başlayabilirler.

Herhangi bir yazılım sistemi için geliştirilen paketleri kullanmak isteyen diğer yazılımcılar bu topiğe takipçi olabilirler. Hatta eğer kullanıcılar da topikten gelen paketleri izlemek isterse terminal üzerinden de mesaj tiplerini elde edebilirler. Bu işlem

için .msg sisteminde kendi mesaj tipi yazıldıktan sonra bu mesajın sisteme eklenmesi için CMakeLists dosyasında find_package kısmına message_generation komutu eklenir ve add_message_files komutu ile istenilen msg dosyası ROS'a eklenebilmektedir. Burada dikkat edilmesi gereken unsur eğer paketin ROS gereksinimi varsa bu da listeye eklenmelidir.

2.3.1. Gömülü Yazılımlar

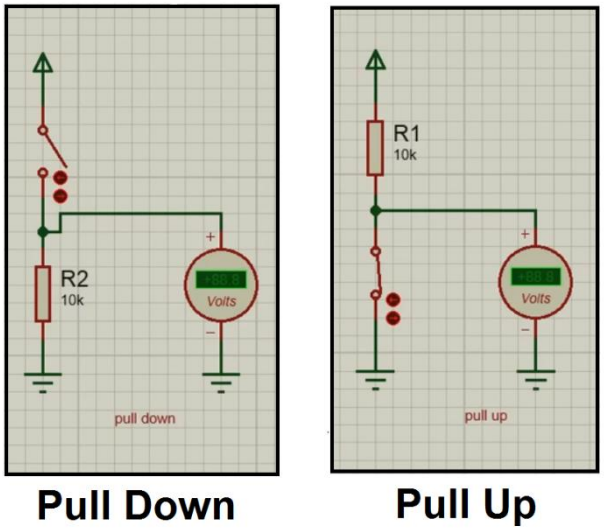
Robot üzerindeki elektronik ve elektromekanik parçaların kontrolleri gömülü yazılımlar ile yapılmaktadır. Robot üzerinde bulunan enkoder gibi sensörlerin okunması ve motorların kontrolleri bu sistemleri yöneten elektronik devrelerde programlanmaktadır. Bu programlar bilgisayardan ayrı çalışmaktadır. Elektronik devreler gömülü yazılım sistemleri ile programlandıkları zaman bağlı bulundukları bilgisayarda bulunan programlara bilgi gönderebilir ya da onlardan emir alabilmektedirler.

Gömülü yazılımlarda önemli olan kullanılan mikroişlemciye uygun işlemlerin yapılmasıdır. Her işlemcinin farklı özellikleri olduğu için işlemcinin hafızası ve işlem kapasitesi dâhilinde işlem yapılması gerekmektedir. Gömülü sistemlerde olan bu kısıtlı kaynaktan dolayı, yapılacak işlemleri mümkün olduğunca etkin yapmak ya da daha fazla işlemciye bölmek gerekmektedir. Gömülü sistemlerde yazılım geliştirilirken kullanılan her değişkenin boyutu önemsenmelidir. Eğer gerekli değilse yeni bir değişken tanımlanmadan işlemler yapılması mikroişlemcinin çok daha etkin kullanılmasına olanak sağlayacaktır. Eğer yapılan işlemler işlemcinin kapasitesinin çok üzerinde çıkarsa bu durumda ya bölünebilecek işlemleri bölerek; farklı bir işlemci kullanmak ya da daha yüksek bir mikroişlemci ile işlemleri yapmak gerekmektedir. Günümüzde kullanılan bir çok entegre sistemde mikroişlemciler kullanıldığından gömülü yazılımların önemi çok büyüktür. Projelerde kullanılan sensörlerin bir kısmında gömülü yazılım bulunmaktadır. Bazı sensörler ise sadece dijital ya da analog sinyal verdikleri için dışarıdan programlanan bir mikroişlemciye bağlanmaları gerekmektedir.

Tez çalışmaları kapsamında kullanılan enkoder verisi okunması ve robotun motorlarının kontrolünde mikroişlemcilerin programlanması gerekmektedir. Yine projede kullanılan LIDAR ve IMU sensörleri ise doğrudan bilgisayara GPIO veya USB protokolleri ile bağlanabilmekte ve gerekli bilgileri kendilerini okuyan yazılımlara gönderebilmektedir.

Sensörler bilgilerini doğrudan bilgisayara GPIO ile iletebilirler. USB üzerinden haberleşen sistemlerde gömülü yazılımlarının yazılması gerekmektedir. Her iki haberleşme sisteminde verilerin okunması için bilgisayar üzerinde gerekli yazılımların bulunması gerekmektedir. Bilgisayar ve mikroişlemciler arasında kullanılan haberleşme protokolüne göre okuma ve yazma hızlarının belirlenmesi gerekmektedir. Eğer yüksek frekansta haberleşme isteniyorsa bu hızı destekleyecek bir sensör ve haberleşme protokolü seçilmesi gerekmektedir.

LIDAR ve IMU gömülü sistemleri kendi içlerinde gömülü yazılım içerdikleri için bu sensörler ile ilgili sadece gerekli haberleşme protokolleri ve haberleşme hızlarının ayarlanması yeterlidir. IMU için bilgisayar üzerinde bulunan GPIO pinleri kullanılmış ve I2C protokolü üzerinden haberleşme sağlanmıştır. I2C protokolü 0 ile 1 MHz arasında frekansta veri transferi yapabilmektedir. Projede kullanılan adafruid 9 DOF BOSCH BNO055 IMU 100 Hz veri gönderebilmektedir. Projede kullanılan Raspberry Pi BNO055 den daha yüksek hızda okuma yaptığı için okuma problemleri olabilmektedir. Raspberry kendi içinde okuma hızını ayarlayamadığı için okuma hızı ayarlamaları yazılımsal olarak yapılmıştır. Ayrıca, Raspberry'ye clock strectching olarak da bilinen BNO055 işlemcisinin okuma frekans özellikleri belirlenmiştir. I2C veri okumada sorun yaşanması durumunda sensöre göre akımın yükseltilmesi pull up veya azaltılması pull down işlemleri gerekebilir. Bu işlemler Şekil 43'te gösterilmektedir.



Şekil 43. Pull Down ve Pull Up Devreleri

Tez çalışmasında sensör pull up işlemi ile desteklendiğinde sistem sorunsuz çalışmıştır. Bu pull up işleminden kurtulmak için BNO055 serisinin Adafruid tarafından geliştirilen devre kartı kullanılmıştır. Bu kart ile sistem sadece tek bir kart ile çalışmakta gerekli yükseltme işlemleri kart içinde olmaktadır.

Raspberry üzerinde bulunan I2C haberleşme sistemi üzerinden haberleşme yapılmıştır. Bu haberleşme türünde 4 adet kablo, güç bağlantıları için Vin ve GND kabloları, haberleşme için de SDA ve SCL kabloları kullanılmıştır. Sensörde bulunan SDA'ya karşılık Raspberry GPIO pinlerinde bulunan SDA, SCL pinine karşılık yine Raspberry'de bulunan SCL pini kullanılmıştır. Bu şekilde bağlantı yapılması ile haberleşme sağlanmıştır.

BMI160 IMU verilerinin okunması için Raspberry üzerinde python kodları hazırlanmıştır. Bu kodlar genel olarak GPIO pinlerinin tanımlanması, gerekli izinlerin verilmesi ve IMU'dan gelen verilerin yorumlanarak programda kullanılabilir hâle gelmesi sağlanmıştır. Raspberry'de GPIO verilerinin okunması için yapılan çalışmada RPi.GPIO ve smbus kütüphaneleri; Bu kütüphanelere Linux üzerinden izin vermek için de python sys ve os kütüphaneleri; Veri işlemek ve analiz işlemleri için time ve numpy kütüphaneleri kullanılmıştır.

Veriler okunmaya başlamadan önce yazılımın cihaza erişimine izin verilmesi gerekmektedir. Bu amaçla dev dizini altındaki /dev/i2c-1 cihazına gerekli okuma izinleri verilmelidir. Bu işlem için Linux işletim sistemi üzerinden cihaza chmod izinleri verilir.

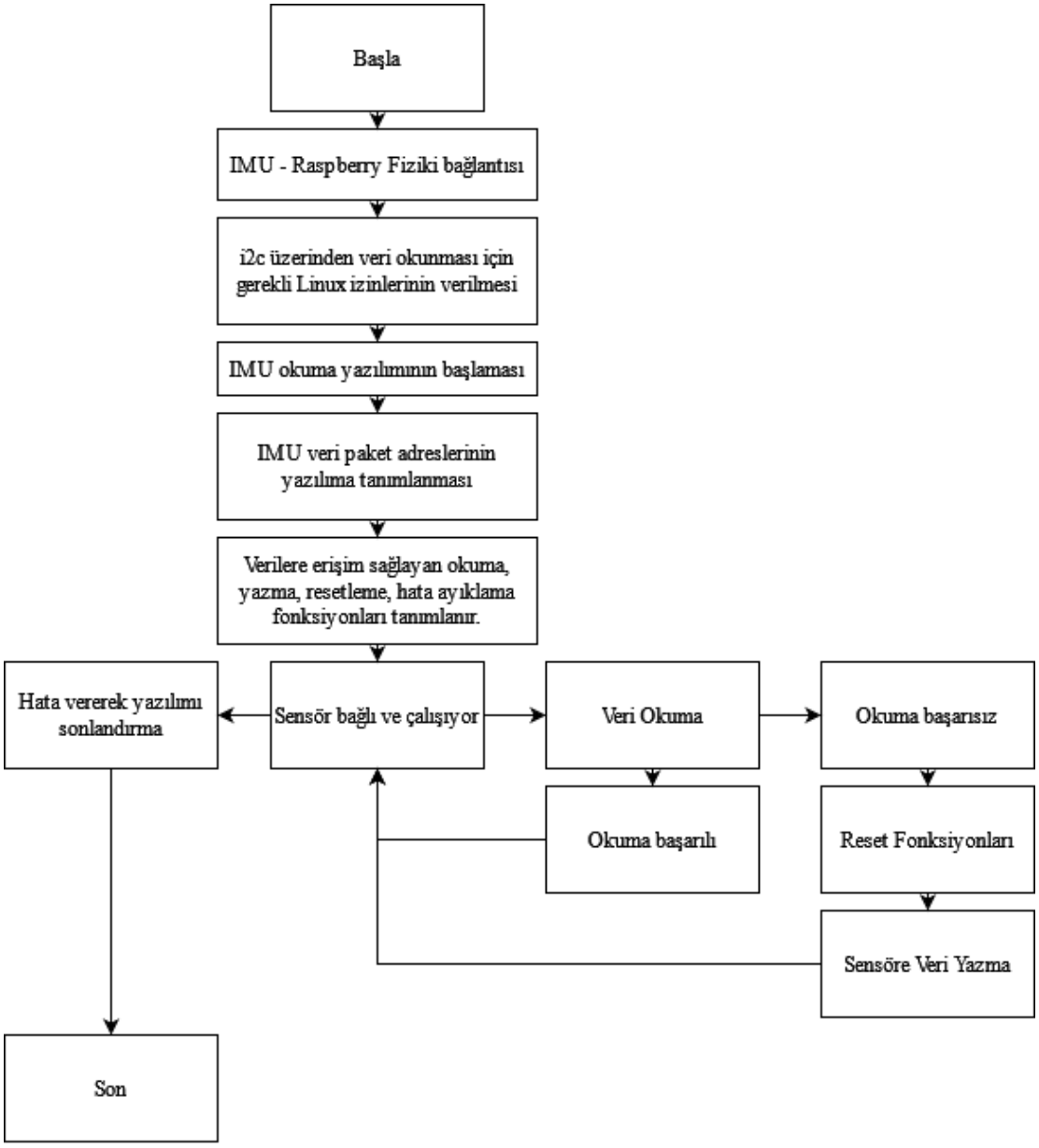
İşlemin yapılması için "os.system('sudo chmod a+rw /dev/i2c-1')" komutu verildikten sonra IMU dan gelebilecek tüm verilerin hex karşılıkları tanımlanarak IMU'dan gelen verilerin anlamlandırılması için ortam hazırlanır. Bu işlemden sonra gerekli değişkenler ve fonksiyonlar tanımlanarak sensörün güç modu, statüsü, reset durumları, veri okuma durumları tanımlanır. Bu fonksiyonlar BMI160 sistemi ile belirlenmiş ve dokümantasyonunda belirtilmiştir. Bu fonksiyonlara erişmek için gerekli kodlar yazılarak veriler bu şekilde kullanılabilir hâle gelmiştir.

6 eksende verebilen BMI160 sensörü, IMU'yu 9 eksende bilgi verebilen ve ROS entegrasyonuna daha kolay uyum sağlayan BNO055 ile değiştirilmiştir. Bu IMU verisini okumak için Dheera'nın geliştirdiği ROS kütüphanesi kullanılmıştır. Raspberry'den veri çekmek bu sensör için diğer IMU'ya göre daha zor olmuştur. IMU verilerini yazılımsız

okumak ve sensör bilgilerine erişmek için i2cdetect programı kullanılmıştır. BNO055 sensörü ile i2cdetect üzerinden ilk aşamada veri elde edilememiştir. Sorunu gidermek için I2C ile ilgili Raspbery dokümanları araştırılmıştır. Bu konuda yazılan dokümanlarda boot/config.txt üzerinde dtoverlay=i2c-gpio,bus=3 işleminin eklenmesinin gerekli olduğu görülmüştür. Bu komut sanal bir GPIO cihazı oluşturarak bu cihazı i2c-3 olarak tanımlanmıştır. Bu işlemin sebebi ise işletim sisteminin doğrudan IMU üzerinden veri okumayarak bunu sanal bir cihaz üzerinden yapılmasına izin vermesini sağlamaktır. Ayrıca, projede kullanılan Raspberry de Raspian işletim sistemi yerine Ubuntu kullanılmıştır. Bu işletim sistemi değişikliği nedeniyle bu dosya /boot/firmware/syscfg.txt aktarılmıştır. Bu dosyada da gerekli değişiklikler yapılmasına rağmen yine de veriler okunamamıştır. İkinci aşamada Ubuntu'da veri okumak için farklı bir sistemin daha değişmesi gerekliliği ortaya çıkmıştır. Bu diğer sistem dosyası da /etc/modules dosyasıdır. Bu dosyaya i2c-bcm 2708 ve i2c-dev komutları girilmiştir. Bu komutlar I2C için gerekli izinleri vermiş ve I2C protokolünü geliştirme modunda erişime açmıştır. Bu işlemlerden sonra i2c-tool kütüphanesi apt ile yüklenmiş son olarak smbus kütüphanesi python ile yüklenmiştir. Tüm bu işlemlerin sonunda i2cdetect ile tüm sensör bilgilerine erişim sağlanmıştır.

IMU bilgilerinin okunarak ROS üzerinde paylaşılmasını sağlayan kütüphanede 3 önemli sınıf bulunmaktadır. Bunlardan biri, cihazda olan değişimleri ve sorunları algılamak, gerektiğinde işlemleri sonlandırmak için kullanılan watchdog kütüphanesidir. Bu kütüphane ayrı bir thread de çalışarak yazılımın çalışmaya devam edip etmediğini fonksiyonel olup olmadığını ölçmektedir. Gerektiğinde resetlemekte, gerektiğinde de tüm işlemleri durdurmaktadır. Watchdog kütüphanesindeki header sınıfı tanımları tuttuğu için bilgiler paylaşılmamaktadır. Burada sadece işlemlerin yapıldığı sınıf tanımlanmıştır.

Bir diğer kütüphanede daha önce BMI160 sensörde kullanıldığı gibi adres tanımlarının yapıldığı ve verilerin okunduğu bno055_i2c_driver sınıfıdır. Bu sınıfın header dosyasında tüm adres tanımları yapılmış ve sınıfın içinde başlatma, uyuma, okuma gibi fonksiyonlar tanımlanmıştır. IMU çalışması akış diyagramı Şekil 44'te gösterilmiştir.



Şekil 44. UMU Çalışma Akış Diyagramı

LIDAR sensörü bilgisayara doğrudan bağlanabilmektedir. Bu sensör HID cihazı olarak işlem görmektedir. Verilerini kendisine bağlanan cihaza istenilen frekansta veri kümesi olarak gönderir. LIDAR'da önem verilmesi gereken konu enerjinin yeterli olmasıdır. LIDAR enerjisini bilgisayar üzerinden çekmesi durumunda, bu enerjinin mümkün olduğunca kararlı olması gerekmektedir. Enerjinin kararlı olması hem okunan verinin doğru okunması hem de LIDAR'ın sağlıklı çalışması için gereklidir.

Enkoder verilerinin okunması için proje kapsamında mikroişlemci kullanılması gerekmektedir. Enkoderin motor dönüşlerini tespit etmek için enkoder üzerinde çentikler

bulunaktadır. Motor şaftı her çentiğin üzerinden geçerken enkoderin üzerinden geçen akım değişmektedir. Bu akım değişikliklerine tik denilmektedir. Enkoder bu değişimlere göre sinyal vermektedir. Enkoder tikler ile sadece 0 ve 1 olarak veri gönderdiği için bu verilerin anlamlandırılarak motor dönüş sayısının belirlenmesi gerekmektedir. Mikroişlemci enkoderden alınan sinyalin değerine göre dönüşü hesaplamaktadır. Bu işlem için mikroişlemci sürekli enkoderi izlemektedir. Motorun dönüşü sırasında oluşan 0 ve 1 değerleri sayılarak motorun kaç derece döndüğü tespit edilmektedir. Burada enkoderin hangi aralıklarla veri gönderdiği bilgisinin mikroişlemci tarafından bilinmesi gereklidir. Bu aralıklar ile toplamda kaç tur dönüş olduğu tespit edilebilecektir. Elde edilen tur miktarları mikroişlemciden ana bilgisayara doğrudan sayı olarak gönderilmektedir. Gönderilen bu veriler de bilgisayardaki programlar aracılığıyla kolaylıkla işlenebilmektedir. IMU verileri bilgisayardan okunurken gelen tur sayısı bir sayısal değere bağlı bulunmaktadır. Bu değerin mikroişlemci üzerinde bir hafızası bulunmaktadır. Bu değer belli bir miktarı geçtiğinde sıfırlanması gerekmektedir. Bilgisayar üzerindeki yazılımın bu miktara göre sıfırlanması işlemlerin sürdüğünü göstermektedir.

IMU gömülü kodlarında sadece gelen veriler değerlendirilip bilgisayara aktarılacağı için Arduino'da özellikle farklı bir kütüphane kullanılmasına gerek yoktur. ROS ile iletişim kütüphanerinin kullanılması yeterli oluğu haberleşme paketleri için std_msgs kütüphanesinde int16 ile string kütüphanelerinin bulunması gerekmektedir.

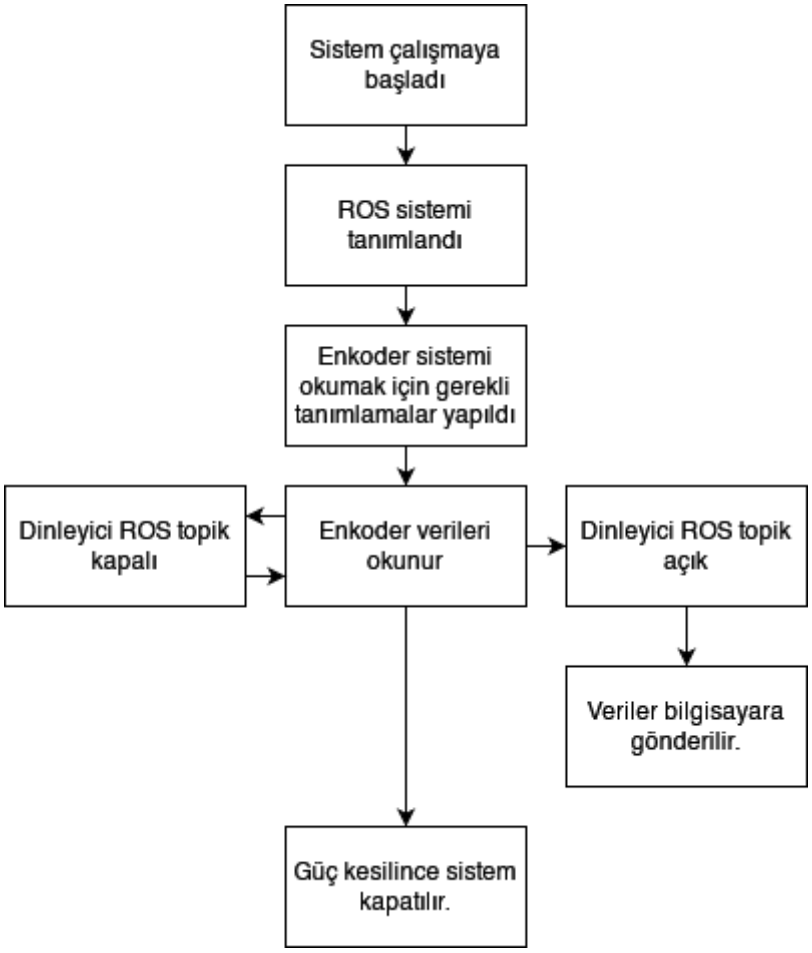
ROS ile ilgili başlama ve bitme düğüm işlemlerinin ve ROS takipçi, yayıncı işlemlerinin yapılabilmesi için gerekli kodlar yazılmalıdır. ROS tanımlama işlemleri yapıldıktan sonra enkoder için dönüş yönü ve dönüş tik sayısı özelliklerinin belirtilmesi gerekmektedir. Proje kapsamında yapılan robotta iki adet motor bulunması nedeniyle mikroişlemci üzerinden enkoder verilerini okumak için dört adet port belirlenmiştir. Motorları sağ ve sol olarak ayırdığımız, tik sayılarına A ve dönüş yönünü B olarak ifade ettiğimizde tanımlama kodu yapılmış olmaktadır. Yönü önceki değerlerle karşılaştırmak için ayrıca bir değişken belirlenmesi gerekmektedir. Enkoderin dönüşü için hafızada belirli bir limit bulunmaktadır. Bu limitin aşılmaması için bir kontrol tanımlanmalıdır. Hesaplamaların yapılması için periyot, önceki zaman ve şimdiki zaman değerlerinin belirlenmesi gerekmektedir. Mikroişlemcilerin kurulum işlemleri için daha önce belirlediğimiz pin değerleri; pinlerdeki dalgalanmalar için de event'ler tanımlanır. ROS ile ilgili bilgisayara

hangi hızla haberleşme sağlanacağı belirlenir. ROS için gerekli olan yayın adları ROS düğümünde oluşturulur.

Enkoder pinlerindeki dalgalanmaları yakalamak için gereken tik olayları otomatik olarak tanımlanmaktadır. Olay oluştuktan sonra enkoderin dijital değeri okunarak motorun ileri ya da geriye doğru gittiği tespit edilir. Yön belirlendikten sonra eğer motor sağa doğru hareket ediyorsa tik sayacı arttırılırken sola doğru giderken tik sayacı değeri düşürülmektedir. Eğer sayaçlar daha önce belirlenen limit değerlere ulaşırsa tam ters yöndeki en yüksek değere geçirilir. Değer düşme işlemine buradan devam edilir.

Tik hesaplamaları yapıldıktan sonra bu işlemlerin zamana göre belirlenmesi ve ROS üzerinde sürekli yayınlanması için döngü fonksiyonu kullanılmaktadır. Döngü fonksiyonu işlemcinin dönme hızında çalışmakta, ROS'a da aynı hızda yayın yapmaktadır. Yayınlanan veri daha önce tanımlanan sağ ve sol için 16 bitlik sayısal değerlerdir.

Tüm bu işlemler yapıldıktan sonra enkoder verileri bilgisayardaki yazılıma sürekli gönderilmeye başlanmış olmaktadır. Bu kütüphanenin ROS üzerinden çalışabilmesi için bilgisayar üzerindeki ROS serial kütüphanesine yüklenerek çalıştırılmalıdır. Enkoderin çalışma akış diyagramı Şekil 45'te paylaşılmıştır.



Şekil 45. Enkoder Çalışma Akış Diyagramı

Motorların iletişim işlemleri için ROS haberleşme yapısından yararlanılması planlanmıştır. Haberleşme için Raspberry üzerinde rosserial kütüphanesi kurulmuş; Bu kütüphanenin çalışması için gerekli haberleşme hızları, port bilgileri ROS launch dosyasında belirtilmiştir. Gömülü sistemlerde ise Arduino ile uyumlu ROS kütüphanesi kurularak sisteme girilmiştir. Gömülü sistemde ROS'tan yayın alınacağı için ilgili topikten okumak için gerekli tanımlamalar yapılmıştır. ROS launch dosyasının çalışması ile topiklere takipçi olarak tüm hareket emirleri dinlenebilmektedir. Burada tekerlerin çalışması için kullanılan topic /cmd_vel olarak ayarlanarak sistemde teker hareketini isteyenler bu topiğe emirlerini yazmaktadır.

Motorların mikroişlemciler tarafından kullanılabilir olması ve güç ihtiyaçlarının dengelenmesi için harici kartlardan motor sürücü kartlarına ihtiyaç duyulmaktadır. Motor sürücüler içinde bulunan sistemlerle farklı gerilimlerde akıma daha dayanıklı olarak motorları çalıştırabilmektedir. Ayrıca, çoğu sürücü yazılımla motorları PWM ile

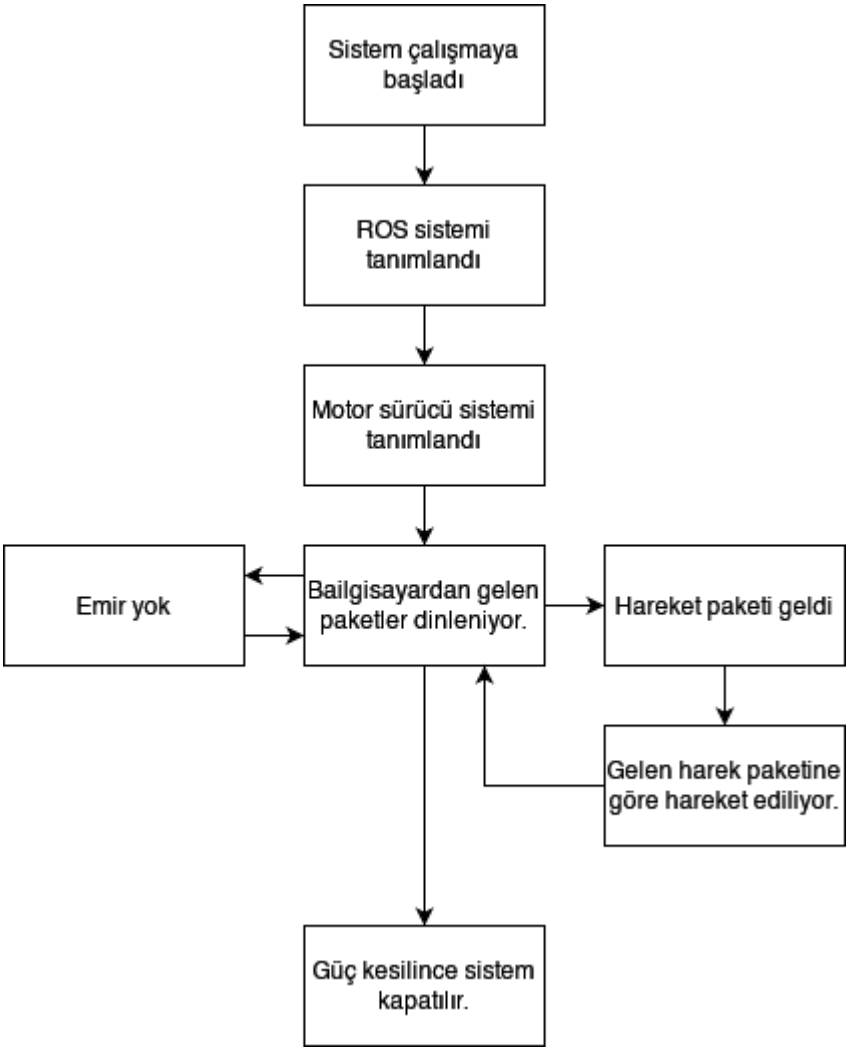
yöneterek çok daha yumuşak ve etkin hareketler verebilmektedir. Motor sürücünün çalışması için çok fazla parametre gerekmektedir. Bu parametrelerin fazla olması nedeniyle motor sürücülerdeki bağlantı pinleri de artmaktadır. Motor sürücünün pin sayısı mikro işlemcinin pin sayısına eşit olduğu zamanlarda mikro işlemci motor dışında farklı bir donanıma bağlanamamaktadır. Projede kullanılan motor sürücünün de pin sayısı kullanılan mikro işlemci pinlerine eşit olduğu için bu mikro işlemci sadece motor sürmek için kullanılmıştır. Yazılım olarak motorların hareketini motor sürücü üzerinden sağlamak için motor sürücüyü geliştiren firmalar, desteklenen kartlara özgü sürücüler ya da kütüphaneler yazmışlardır. Projede kullanılan motor sürücü kütüphaneleri de ROS kütüphaneleri tanımlandıktan sonra sisteme çağrılır. Motor sürücü kütüphaneleri için yapılması gereken en yüksek hız, en düşük hız, hata ayıklama sistemleri gibi ayarlamalar da yapılmalıdır.

ROS üzerinden veri okunurken yazma işleminden farklı olarak bir de callback sistemi tanımı yapılması gerekmektedir. Bu callback ROS sistemine yazılan her emirde tetiklenerek Raspberry'deki ROS topiğinden gelen verilerin kullanımına imkân vermektedir. ROS topiğindeki veride motorların nasıl hareket etmesi gerektiği konusunda açısal ve doğrusal hız değerleri kullanılmaktadır. Bu değerler sayesinde sadece ileri git, geri git gibi basit komutlar değil ayrıca yavaş yavaş ilerle, hızlı geri git gibi işlemlerde kolaylıkla yapılabilmektedir. Motorun hızı gelen mesaja, kullanılan tekerle formülüze edilerek bulunmaktadır. Topikten gelen liner hıza speed_lin, açısal hıza speed_ang ile ifade etmektedir. Tekerleğin sahip olduğu tekerlerin uzaklığı wheel_sep, tekerlerin çevre uzunluğu da wheel_rad ile ifade edilmektedir. Motorların PWM değerleri w_m (numara) değeri 13 ve 14 numaralı Denkelemler elde edilmektedir.

```
w_m1 = (speed_lin/wheel_rad) + ((speed_ang*wheel_sep)/(2.0*wheel_rad));   (13)
```

```
w_m2 = (speed_lin/wheel_rad) - ((speed_ang*wheel_sep)/(2.0*wheel_rad));   (14)
```

Motor buradan gelecek değerin negatif olması durumunda geriye doğru, pozitif olması durumunda ise ileri doğru hareket etmektedir. Motorun hızını da doğrudan bu değeri PWM değeri olarak ayarlanamsı durumunda erişilmektedir. Motorun çalışma diagramı da Şekil 46'da verilmiştir.



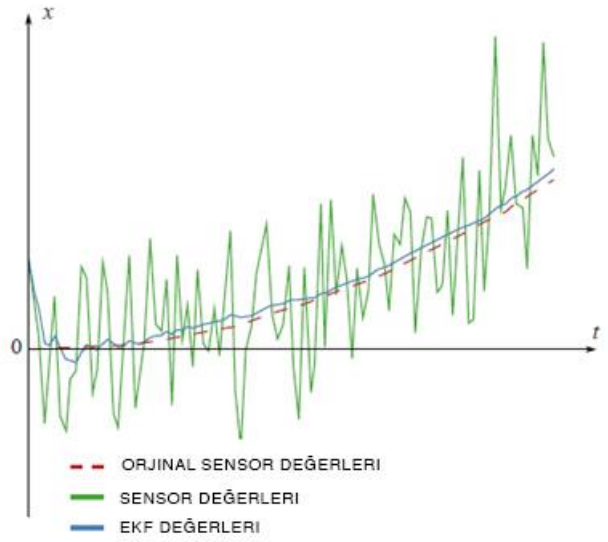
Şekil 46. ROS Üzerinden Motor Kontrolü Akış Diyagramı

Gömülü sistemler robota güç verildiği anda robot üzerindeki Raspberry Pi üzerinden güç alarak çalışmaya başlamaktadır. ROS sistemi aktif değilken bu sistemler çalışmakta ama herhangi bir yere veri göndermemektedir. ROS açıldığında kendiliğinden sıfır değerlerine geri dönmektedirler. Özellikle enkoderde sayım değerleri gömülü sistemin üzerinde tutulduğu için bu konu verilerin hesaplanması konusunda önem taşımaktadır.

2.3.2. Kalman Filtreleri

Sensörler günümüzde bir çok alanda yaygın olarak kullanılmaktadır. Ancak kullanılan sensörlerin toleransları bulunmaktadır. Sensör toleransları bazı uygulamalar için kritik öneme sahiptir ve çok daha yüksek doğruluğa ihtiyaç duyulmaktadır. Bu doğruluğun sağlanması için ucuz, uygulanması kolay ve başka sistemlere entegre edilebilen bir sisteme gereksinim duyulmuştur. 1960 yılında Rudolf E. Kalman tarafından geliştirilen Kalman filtreleri temel olarak uçak sistemleri için tasarlanmıştır. Kalman filtreleri lineer ve lineer olmayan uygulamalar için oldukça başarılı sonuçlar göstermiştir. Kalman filtreleri temel olarak bir çok sensör verilerinin birleştirilmesi ile doğru verinin alınmasını; çok sensör olmadığı durumlarda da oldukça başarılı sonuçlar elde edilmesini sağlamaktadır.

Yıllar içerisinde Kalman filtreleri geliştirilerek bir çok versiyonu oluşturulmuştur. Bunlardan bazıları lineer Kalman filtresi, Extended Kalman filtresi (EKF), continuous Kalman Filtresi, continiuous-discrete Kalman filtresi, centralized Kalman filtresi, vb. Kalman filtreleri ile gelen verilerin incelenerek sensörden gelen verilerin gerçek değerlere yaklaşmasını sağlar. Şekil 49'da gösterildiği gibi verinin lineere yakın bir şekilde değişimi gösterilmiştir.



Şekil 47. Sensör Verilerinin EKF ile değerlendirilmesi

Kalman filtrelerinin kullanılabilmesi için, kullanılacak sistemin fiziksel modellemesinin olması hız ve yerleşimin zamana göre değişiminin belirlenmesi gerekmektedir. Kalman filtreleri hesaplanırken Kalman kazancı (KG) olarak belirtilirken, tahmin edilen hata için E_{est} ve hesaplama hatası E_{mea} olarak; zamanlara göre ifadeler ise t, t+1, t-1 olarak EST_t olarak verilmektedir. Bu veriler değerlendirildiğinde Kalman kazancı Denklem 15, t +1 zamandaki hesaplama hatasına Denklem 16 ve t+1 zamanda tahmin edilen hata Denklem 17 ile hesaplanabilmektedir.

$$KG = E_{est} / (E_{est} + E_{mea}) \quad (15)$$

$$E_{t+1} = E_t + (KG * (mea - EST_t)) \quad (16)$$

$$E_{est\ for\ t+1} = (1 - KG) * (E_{est\ for\ t}) \quad (17)$$

Kalman filtresi ilk çıktığı zamandan beri geliştirilerek daha önce kullanılmadığı birçok alanda kullanılması sağlanmıştır. Farklı uygulamalar için geliştirilen çeşitli Kalman filtreleri sayesinde lineer ya da lineer olmayan sistemlerin ve sensörlerin de incelenerek anlamlandırılması sağlanmıştır. Kalman filtreleri temel olarak lineer sistemler için geliştirilmiş ve temelleri Bayesian filtreleri teorilerine dayanmaktadır. Kalman ağırlık hesabı için geliştirilen kazanç formülü ile hesaplanır. Formül 18' de belirtilen $\overline{p_k}$ tahmin edilen kovaryans matris durum vektörü, H_k^T tasarım matrisi, R_K gözlem için kullanılan kovaryans matrisi olarak gösterilir.

$$KG = \overline{p_k} H_k^T (H_K \overline{p_k} H_k^T + R_K)^{-1} \quad (18)$$

Kalman sistemlerinin en önemli özelliklerinden biri de istenilen uygulamaya göre uyarlanabilmesi ve çok çeşitli alanlarda farklılaştırılarak kullanılabilmesi olmuştur. Zamanla değişen uygulamalar, farklı tiplerde oluşan gürültüler, hesaplamalarda oluşan zorluklar nedeniyle Kalman Filtreleri zamanla farklı uygulamalara göre değişerek özelleşmeye başlamıştır.

Kalman filtreleri ile ilgili genel olarak iki farklı temel esas alınır. Bunlar, Kalman filtrelerinin her zaman Gaus Dağılımı kullanmaktadır. Bu gereklilik nedeniyle Kalman filtreleri her zaman lineer fonksiyonlar ile çalıştırıldığıdır. Gaus Dağılımı klasik olarak değerlendirilir ve daha önceki verilere bakılarak gelecekteki durumların tahmini ile ilgili esasları içerir.

Lineer fonksiyonlar için durum biraz daha değişmektedir. Lineer olan bir fonksiyon Gaussian sisteme sokulduğunda ortaya çıkan sonuç yine Gaussian olur. Lineer olmayan bir sistem Gaussian sistemine sokulduğunda ortaya çıkacak sistem Gausssian dağılımına uymayacaktır. Bu durumda ise bir Gaussian dağılımını kullanımı mümkün olmayacaktır. Bu şekilde lineer olmayan veriler için onları lineer hâle getirmeye yönelik yeni bir sistem kullanılması gerekecektir. Bu sistem lineer yakınsama olarak isimlendirilmektedir. Bu yöntemle Kalman sistemlerinin lineer olmayan yapılara uygulanmasına da Extended Kalman Filtrelemesi EFK denilmektedir.

Lineer olmayan sistemlere lineer yakınsama yapabilmek için Taylor serilerinin araç olarak kullanılması gerekmektedir. Bu araçtan alınan çıktılar artık Kalman filtresinde kullanılmak için yeterli olacaktır. Taylor serilerinin kullanılabilmesi için $f(a)$ verilerine diferansiyel formüller uygulanarak lineer olmayan formüle göre Gaussian dağılımını kullanılabilir yapılmaktadır. Taylor serisi hesabı yapılırken Denklem 19 kullanılmaktadır.

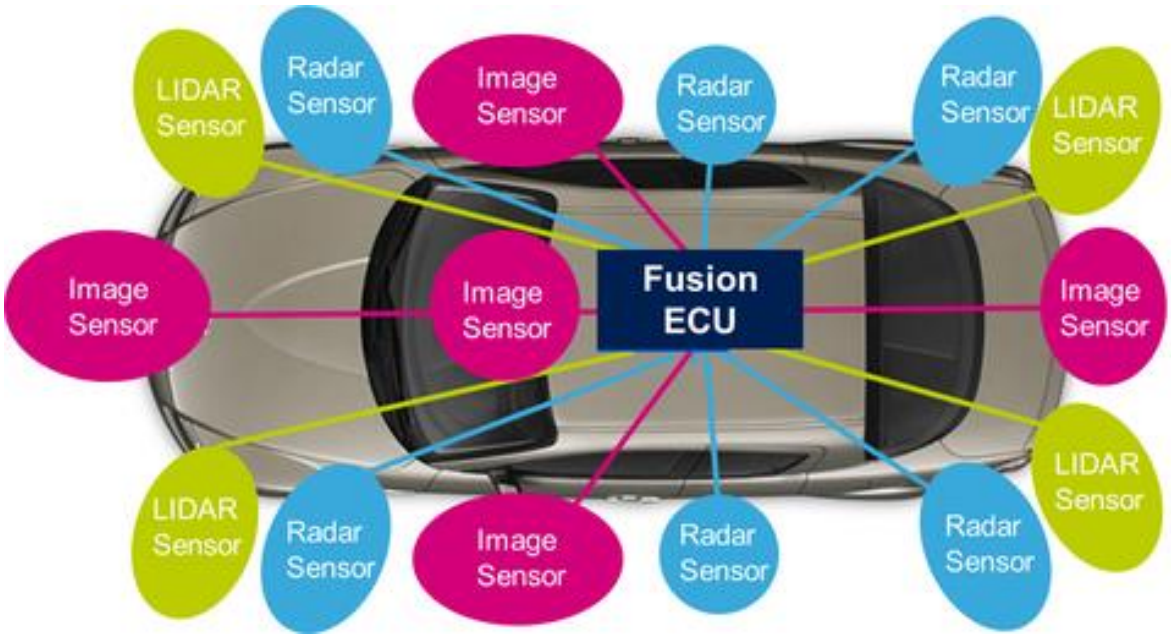
$$f(x) \approx f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots \qquad (19)$$

Taylor serileri ile lineer olmayan girdilere de Kalman filtrelerinin başarılı şekilde uygulanabilmesi bu filtrenin önemini son yıllarda oldukça yükseltmiştir.

Yeni geliştirilen otonom cihazlarda da konum belirleme ciddi bir önem teşkil etmektedir. Kullanıcıların araçlarını aktif şekilde kullanmalarına ihtiyaç duymaması amaçlanmıştır. Otonom cihazlar kendi konumlarını tam anlamıyla bildikleri durumlarda, çevresindeki olayların da farkında olurlarsa kendilerini usta bir şoför gibi geliştirerek araçları hasarsız ve kazasız biçimde kullanabileceklerdir.

Kalman filtreleri temel olarak konum belirlerken tek bir sensörden gelen verilere dayanmamakta, araçta konum belirleme konusunda olabilecek her türlü veriyi

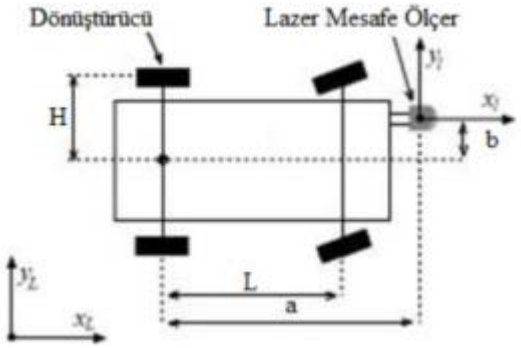
değerlendirmekte ve gerçeğe en uygun verinin araç tarafından kullanılmasını amaçlamaktadır. Otonom araçlarda kullanılan kameralar onlar için kritik öneme sahip sensörlerden biridir. Kameralardan gelen veriler incelenip GPS, IMU, Barometre gibi başka sensör verileri ile birleştirilerek daha doğru ve gerçek zamanlı konum bilgisi oluşturulabilir. Ayrıca, cihazlarda bulunan kameralar yardımı ile bu sistemler coğrafyaya kayıtlı noktaların bulut sistemine kaydolması ve neural networkler tarafından işlenmesi de sağlanabilecektir. Geografik 3D bulut verilerinin 2D dijital haritalara kamera verileri kullanılarak uygulanmaktadır. Şekil 48'de gösterilen grafikte aracın kontrol kartı olan ECU sistemlerine bağlı olan sensör sistemleri gösterilmiştir. Şekilde de görüleceği gibi araç sensörlerle donatılmıştır. İhtiyaca göre çok sayıda aynı tip sensörden birkaç adet bulunabilir.



Şekil 48. Araç İçinde Kontrol Tarafından Kullanılan Sensörler

(Kaynak: Liu, 2018)

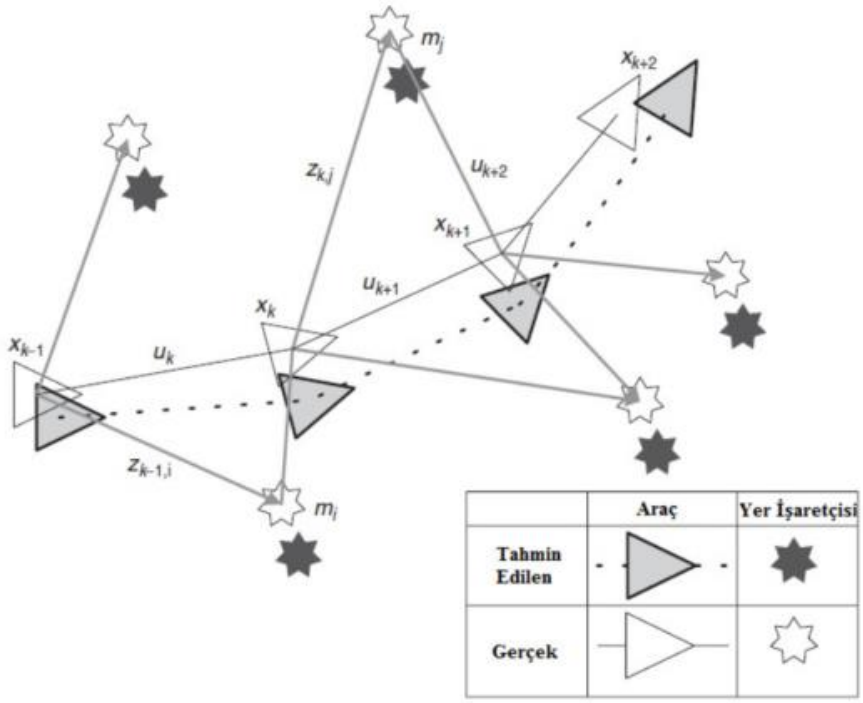
Kalman filtrelerinin kullanımı ile sensör verilerinin güvenilirliği arttırıldığında karşılaşılan engellerin sadece birer gürültü veya hata eseri olmayıp gerçek engeller anlaşılarak dinamik olarak korunma tedbirleri de alınabilecek bir duruma gelmiştir. Otonom bir aracın grafiği Şekil 49'da verilmiştir.



Şekil 49. Otonom Bir Aracın Kinematiği

Şekil 50'de belirtilen L aracın iki tekeri arasındaki uzaklık olarak ifade edilmektedir. Otonom araçta kullanılan lazer ile aracın merkezi arasındaki mesafede b olarak nitelendirilmektedir. Grafikteki H aracın tekeri ile aracın merkezi arasındaki mesafeyi ifade etmektedir.

Kalman ile işlenen veriler aracın kinematik sistemine sokularak doğru şekilde aracın konumlandırılmasında etkili olmaktadır. Araçlara konulan lazer, radar, enkoder gibi sistemlerin bilgileri kullanılarak aracın kinematik bilgileri Kalman filtrelerinde kullanılarak etkin konum tespiti yapılmasına olanak sağlanmaktadır. Bu tarz sistemler lineer sensör verileri göndermedikleri için EKF kullanılması gerekmektedir. Bu sistemde Şekil 50'deki Kalman filtresi kullanıldığında sistemin verimliliği artmakta ve konum belirlemedeki başarı yükselmektedir. Araç konum açısının konum vektörü x_k olarak ifade edilmektedir.



Şekil 50. Aracın gerçek konumu ile Kalman filtrelerinin kullanımı

(Kaynak: Souliman, 2017)

Şekil 50'deki aracın gerçek konumu ve tahmin edilen konumu arasındaki fark gösterilmektedir. Görüleceği gibi her Kalman Filtrelerinden çıkan sonuç gerçeğe yaklaşmış bir biçimde paylaşılmaktadır. Kalman filtresi daha önce de anlatıldığı üzere bir tahmin algoritması, gerçeğe uygun tahminler yapabilmesi için sensörlerden gelen verilere ihtiyaç duymaktadır. Gelen veriler Kalman filtreleri ile birleştirilerek ölçümlerdeki farkların bulunması ve gerçek konum bilgisine ulaşılmasını sağlaması amaçlanmıştır. Kalman filtreleri kullanılırken her sensörün kendi içinde kullanılması ile gelen verilerin doğrulukları da artmaktadır. Otonom araçlarda kullanılan bu konum bilgisi sayesinde oldukça doğru kararlar alınmasının önü açılmış olmaktadır.

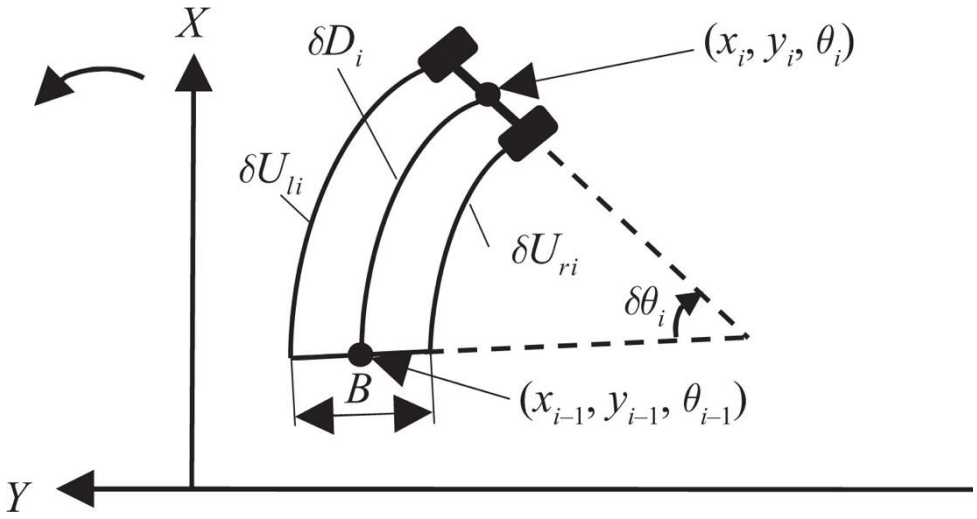
Otonom araçlarda verinin güvenliği ve doğruluğu önemlidir. Bu nedenle verilerin sürekli takip edilmesi doğruluğunun kontrolü ve güvenilirliği sağlanmalıdır. Kalman filtresinin araçlarda kullanılmasının bir diğer avantajı da bu filtrenin kullanımının çok kolay olması ve hızlı şekilde her sensöre uygulanmasının çok fazla kaynak gerektirmemesidir. Kalman filtresi konum belirleme dışında çarpmaya karşı koruma, araç dışı ışıkların açılıp

açılmayacağının belirlenmesi, kaza durumunun tespiti, araç içi acil durumların fark edilmesi gibi pek çok farklı alanda da kullanılabilecektir.

Tez kapsamında geliştirilen mobil robotta veriler lineer olmadığı için EFK filtrelemesi yapılmıştır. Robotun verilerinin doğruluğu ve güvenliği konusuna önem verilmiştir. Kullanılan Kalman filtrelerinin önemini ve yaygın kullanım alanlarından biri olan otonom araçlar otonom araçlar üzerinde konu detaylandırılmıştır. Otonom araçlardaki sistem de mobil robotlardaki sisteme uygun çalışmaktadır.

2.3.3. Odometri

Mobil robotların temel işlevleri bulundukları alan içerisinde hareket edebiliyor olmalarıdır. Bu özelliğin haritalandırma ve navigasyon gibi sistemler ile birleştirilmesiyle robot hareket alanlarının belirlenmesinin önemi artmıştır. Hareket alanlarının tespiti için robotun teker dönüş sayısına göre hesap yapma tekniği olan odometri geliştirilmiştir. Odometri robotun hareketine ilk başladığı konum ile robotun anlık konum değişiminin hesaplanması işlemidir. Robot bu işlemi üzerinde bulunan enkoderlerin sayım miktarları ile yapmaktadır. Enkoderden gelen veriler odometri sayesinde işlenerek robotun nasıl hareket ettiği, nereye doğru hareket ettiği, ne kadar hareket ettiği gibi bilgilere ulaşabilmektedir. Şekil 51'deki grafikte robotun i zamanındaki konumu (x_i, y_i, θ_i) olarak verilmiştir.



Şekil 51. Robotun i Zamandaki Konum Grafiği

(Kaynak: Chen, 2017)

Burada verilen x_i ve y_i değerleri robotun iki boyutlu düzlemdeki konumunu ifade ederken θ_i değeri de robotun düzleme göre oryantasyonunu göstermektedir. Grafikte ifade edilen B ise robotun ilk konumunu belirtmektedir. Robotun i zamanında oryantasyonundaki değişim $\delta\theta_i$ ve gittiği ortama mesafe ise δD_i olarak gösterilmektedir. Robotun oryantasyonundaki değişim ve ortama mesafe değişimi bulmak için sağ ve sol tekerlerin i zamanında aldıkları yol sağ δU_{ri} , sol δU_{li} miktarlarının bilinmesi gerekmektedir. Oryantasyon hesaplanması için sağ tekerin ortalama mesafesinden sol tekerin ortalama mesafesinden çıkarılıp ilk değere bölünerek Denklem 20'ye göre bulunmaktadır. Ortalama ilerleme mesafesi bulunurken sağ ve sol tekerlerin ilerleme sayısı toplanacak ve yarısı alındığında 21 numaralı Denkleme erişilecektir.

$$\delta\theta_i = \frac{\delta U_{ri} - \delta U_{li}}{B} \tag{20}$$

$$\delta D_i = \frac{\delta U_{ri} - \delta U_{li}}{2} \tag{21}$$

Formül 20 ve 21'deki sağ ve sol teker mesafelerinin bulunabilmesi için sağ δU_{ri} ve sol δU_{li} değerlerinin; bu değerlerin belirlenmesi için sağ δN_{ri} ve sol δN_{li}enkoder sayım değerlerinin bilinmesi gerekmektedir. Enkoder verisi teker boyutları ile anlamlandırılmaktadır. Piyasada bulunan her enkoderin farklı özellikleri bulunmaktadır. Bu farklılıklara enkoder çözünürlük katsayısı n denilmektedir. Teker boyutları sağ teker için D_r^ave sol teker için D_l^a olarak verilmektedir. Bu değerler mobil robotlar için genellikle aynı olmakta ve değerleri milimetre olarak hesaplanmalıdır. Enkoder sayısı ve teker çapı ile çarpılıp bu değer enkoder çözünürlüğüne bölündüğü zaman tekerlerin ortalama aldıkları mesafe Denklem 22 ve 23'te belirtilmektedir.

$$\delta U_{ri} = \frac{\delta N_{ri} D_r^a \pi}{n} \tag{22}$$

$$\delta U_{li} = \frac{\delta N_{li} D_l^a \pi}{n} \tag{23}$$

Odemetri için gerekli olan sağ δU_{ri} ve sol δU_{li} değerleri tespit edildikten sonra δD_i değeri elde edilmektedir. Bu değer x ekseni için hareketin ilk başladığı nokta ile hareket miktarı toplamı şeklinde ortaya çıkacaktır. Bu durumda B noktasının konumu Denklem 23'teki gibi olacaktır.

Tez kapsamında geliştirilen robotta iki adet teker bulunmaktadır. Bu tekerlerin çapları 58mm'dir. Çap bilgisi D_l^a ve D_r^a bu bilgilere göre 58mm olarak çıkmaktadır. Mobil robotun 100 cm mesafe $\delta U_{(l-r)i}$, kat ettiğinde yazılımın okuması gereken enkoder dönüş hesabı için n 1632 ve her iki teker için çap bilgisi D_{l-r}^a olarak verilmiştir. Kullanılan enkoderde darbe sayısı hesaplaması yapıldığı zaman darbe sayısı Denklem 24'te gösterilmiştir.

$$U_{(l-r)i} = \frac{D_{l-r}^a \pi x}{n} => 100 = \frac{3.14*58*x}{1632} => x \cong 901\ enkoder\ darbesi \qquad (24)$$

Robot üzerinde enkoder sayım testleri yapıldığında bir metrelik mesafe için 1014 darbe sayısı ölçülmüştür. Referans olarak 1000 darbe sayısına göre işlemler yapılmıştır. Robotta bir derecelik dönüş için gerekli hesaplamaların yapılması için motor aks uzunluğu dikkate alınması gerekmektedir. Bir tur için 360 derece gerektiği için 25, 26, 27 ve 28 numaralı Denklemlere göre elde edilen bilgilerin 360 dereceye bölünmesi gerekmektedir.

$$2\pi \frac{aks\ uzunluğu}{2} = 2 * 3.14 * \frac{12.5}{2} = 39{,}25\ cm \qquad (25)$$

Bulunan 39,25 cm teker çapına bölünmelidir.

$$\frac{39.25}{58} = 0{,}67m \qquad (26)$$

Referans olarak alınana değer ile 0,67 değeri çarpıldığında 360 derece dönüş için gerekli enkoder darbesine ulaşılmaktadır.

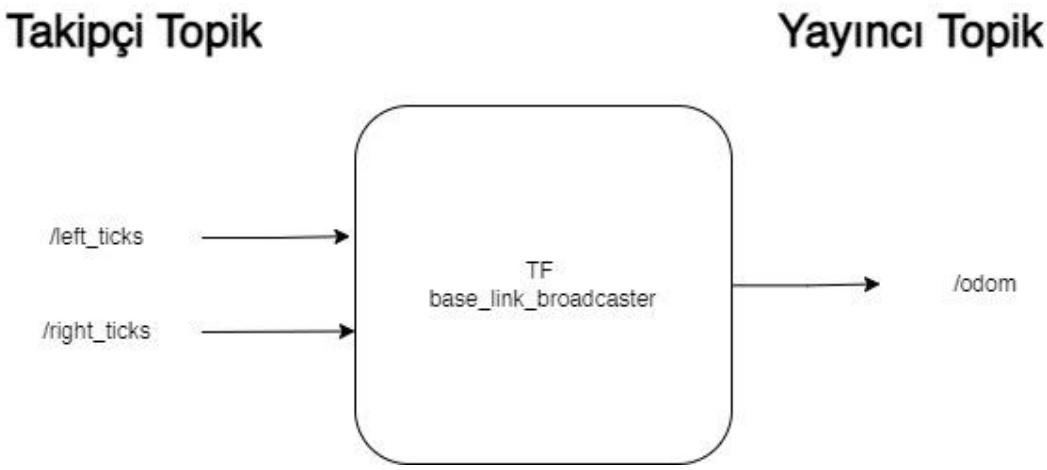
$$0{,}67 * 1000 = 670\ enkoder\ darbesi \qquad (27)$$

Bir derecelik hareket için bulunan 670 derecenin 360 a bölünmesi gerekmektedir.

$$\frac{670}{360} = 1{,}87\ enkoder\ darbesi \tag{28}$$

ROS yazılım paketlerinin gelişmesi ile odometri hesaplamaları çok daha kolay bir yapıya ulaşmıştır. ROS üzerinde enkoder verileri ve statik mesajlar yayınlanmaktadır. Bu mesajlar ile odometri kolaylıkla hesaplanabilmektedir.

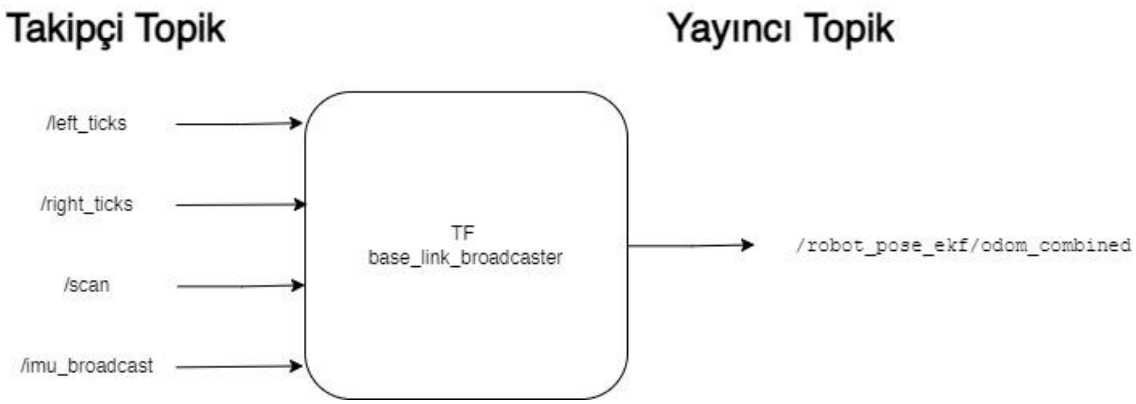
ROS ile odometri hesaplanırken robot üzerinde yayıncı olan left_tick, right_tick, imu_broadcaster, scan verileri alınarak base_link_broadcaster paketine ardından robot_pose_efk topiğine gönderilir. Çıkan sonuçlar ile odometry verisi elde edilmektedir. Bu sistem girdileri ve çıktıları olan bir makinedeki gibi düşünülmektedir. ROS içerisinde bulunan algoritmalar robotun TF bilgilerini ve sensör bilgilerini kullanarak odometri oluşturmaktadır. (Şekil 52)



Şekil 52. ROS Odometri Yayıncı Takipçi Diyagramı

Sensör verileri genellikle lineer veri olmamaktadır. Bu nedenle Kalman filtresi odometri sisteminde kullanılması için önce linerizasyon işlemi yapılması gerekmektedir. Linerizasyon işleminin yapılması gerekliliği nedeniyle odom verisine Kalman filtresinin kullanılması için EFK'ya ihtiyaç duyulmaktadır. ROS ile EFK kullanılarak odometry hesaplanması sonuçlara erişim için çok daha etkin sonuçlar oluşturmaktadır. ROS, EFK kullanılarak sadece enkoder verilerine değil aynı zamanda robot üzerinde bulunan LIDAR, IMU gibi sensör bilgilerinden de faydalanılabilmektedir. EFK tüm sensör verilerini sürekli kontrol ederek tek bir sensörde oluşacak hatalı ölçüm ve

değerlendirmeleri engellemektedir. Geliştirilen robot için EFK odometri hesabının yapılma grafiği Şekil 53'de verilmiştir.



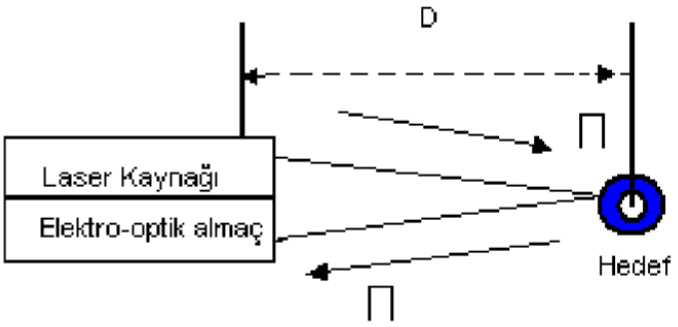
Şekil 53. ROS Odom EKF Yayıncı Takipçi Diyagramı

Odom verisi navigasyon işlemlerinin yapılması için önemlidir. Bu veri olmadan robot tam olarak ne kadar hareket etmiş olduğunu bilemeyecektir. Hareket mesafesini hesaplamak için elimizde olan en önemli sensör enkoderdir. Enkoder ile dönüş sayısı hesaplanması sayesinde teker uzunluklarına göre bir derecelik dönüş için ilerleme miktarını göstermektedir. Odometri ile kullanılacak EFK ile enkoder dışındaki sensörlerin bilgileri de robotun hareket miktarının hesaplanmasına katkı sağlamaktadır. Bu sensörler yardımı ile oluşturulan odometri verisi oluşturulduktan sonra navigasyon ve haritalama işlemleri yapılabilir hâle gelmiştir.

2.3.4. İç Alan Konumlandırma ve Haritalandırma (SLAM)

Mobil robot ilk olarak uzaydaki bir konumda açıldığı zaman çevresi hakkında herhangi bir bilgi sahibi olmadan açılmaktadır. İlk anda sadece bulunduğu noktayı görmekte ve sensörleri ile ilk konum bilgisi edinmeye çalışmaktadır. Bu bilgiyi sağlamak için robotun sensörlerinden faydalanması gerekmektedir. Mobil robotlar için geliştirilen en önemli harita çıkarma sensörüleri LIDAR ya da derinlik algısına sahip kameralardır. Tez kapsamında üretilen mobil robot üzerinde LIDAR olduğu için bu sensörün harita çıkarması üzerinde durulacaktır. Ancak kameralar içinde benzer bir yapı kullanılmaktadır. LIDAR içerisinde bir lazer ve bir alıcı bulundurmaktadır. Lazerden

gönderilen sinyal alıcı tarafına gönderilerek uzaklık ölçülmeye çalışılmaktadır. Lazerin gönderdiği sinyal ve yol miktarı D Şekil 54'de gösterilmektedir.



Şekil 54. Lazer Çalışma Diyagramı

(Kaynak: Şengül, 2006)

Lazer ışığın gidip gelmesi arasındaki süreyi kullanarak mesafe ölçümü yapmaktadır. Bu durumda ışık hızı olarak C kullanılmaktadır. T ile ışığın gidip gelme süresini ifade etmektedir. T gidip gelme süresi olduğu için ikiye bölünmesi gerekmektedir. C ise 300 m/µsn olarak ifade edildiğinde Denklem 29 elde edilmektedir.

$$D = C * {}^{T}/_{2} \tag{29}$$

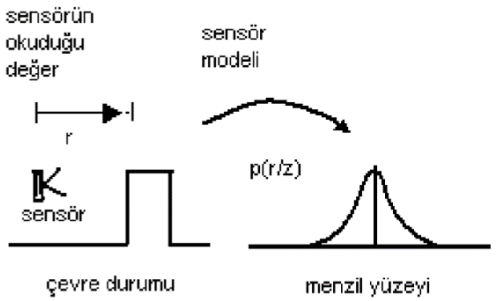
LIDAR bu tarama işlemini 360 derece için yapmakta ve her bir açı için bir engel mesafesi çıkarmaktadır. Eğer engel bulamazsa sonucu hesaplayamadığını bildirmekte ve sonucu nan olarak yazmaktadır. Tez kapsamında kullanılan LIDAR 16 metre mesafeye kadar ölçüm yapabilmekte ve 8 kHz örnek olabilmektedir. LIDAR'ın gönderdiği bu veriler harita oluşumu için önemlidir.

LIDAR ile harita oluşturulurken robotun bulunduğu alan grid adı verilen bölgelere ayrılmaktadır. Bu gridler iki boyutlu haritalar için x_r ve y_r ekseninden oluşurlar fakat robotun hareketine göre açısal olarak θ_r üçüncü boyut eklenmektedir. Robot haritayı çıkarırken bu giridleri işaretlemektedir. Gridler beyaz olduğu zaman o noktaya gidebilirken, siyah olduğu zaman bu gridin bir engel olduğu anlaşılmaktadır. Grid yöntemi kullanılırken bulunan noktanın gidilebilir olduğunun anlaşılması için Bayes tahmin algoritmaları kullanılmaktadır. Bayes daha önceki alınan örneklem verilerini

kullanrak ortam hakkında yeni varsayımların yapılması prensibine dayanmaktadır. Bayes kuralları sensörden okunan r değerinin uzaydaki z noktası ile ilişkilenmesine göre hesaplanmaktadır. Bayes kuralları Denklem 30'da paylaşılmıştır.

$$p(r|z) = \frac{p(Z|r)\,p(r)}{p(z)} \tag{30}$$

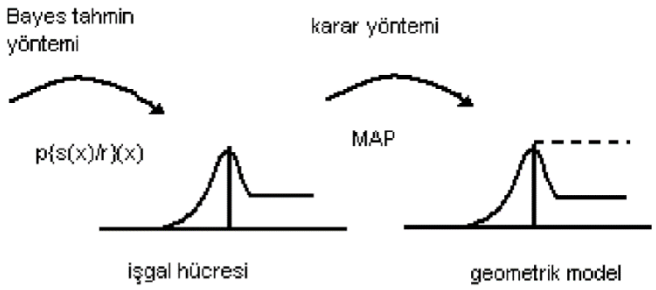
Bayesian kuralları ile sensörden r kadar uzaktaki bir değer okunacak ve bu değer modellenecektir. Bu modelleme işlemi Şekil 55'de gösterilmektedir.



Şekil 55. Sensör Modelleme Grafiği

(Kaynak: Şengül, 2006)

Sensör modeli oluştuktan sonra Bayes tahminleri ile karar yöntemleri kullanılarak harita elde edilmektedir. Bu tahmin ve haritalama işlemleri Şekil 56'da belirtilmiştir.



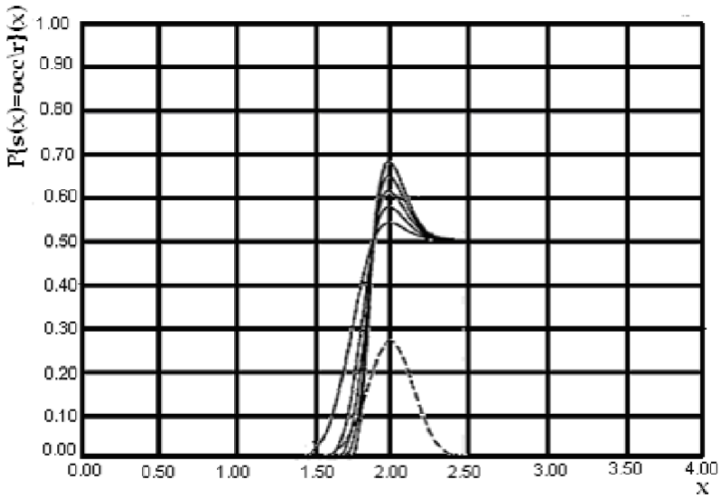
Şekil 56. Sensör Geometrik Model Grafiği

(Kaynak: Şengül, 2006)

Son aşamada harita çıkarılırken, Markov rastgele alan MRF ile birbirinden bağımsız olan noktalarında birleştirilmesi sağlanmaktadır. Robot aldığı sensör bilgilerinden birbirlerine bağlı olanları Bayesian ile birleştirirken bağımsız olanları Markov modellemesi ile kararlaştırmaktadır. Markov modellemesinde hücrenin tahmin durumu $C_i$olarak verildiğinde ve hücreye ait başlangıçtan t anına kadar yaptığı gözlemler $\{r\}_t = \{r_1, r_2, \ldots., r_t\}$ olarak verilmektedir. Mobil robotta yeni gözlem yapıldığında r_{t+1} anı için Markov modellemesi $P[s(C_i) = OCC|\{r\}_t)]$ hesaplaması Denklem 31'de belirtilmiştir.

$$P[s(C_i) = OCC|\{r\}_t)] = \frac{P[r_{t+1}\,|s(C_i)=OCC|\{r\}_t)]\;P[s(C_i)=OCC|\{r\}_t)]}{\Sigma_{s(C)_i}\,p\left[r_{t+1}\middle|s(C_i)\right]P[s(C_i)|\{r\}_t]} \qquad (31)$$

Markov rastgele alan formülünü noktaya tüm zamanlar için uygulanması sonucunda bir işgal olasılık formülü oluşmaktadır. Bu formul sensörün ölçtüğü mesafeye göre olası engellerin konumlarını ve olasılıklarını vermektedir. Bu işgal olasılık profili Şekil 57'de gösterilmektedir.



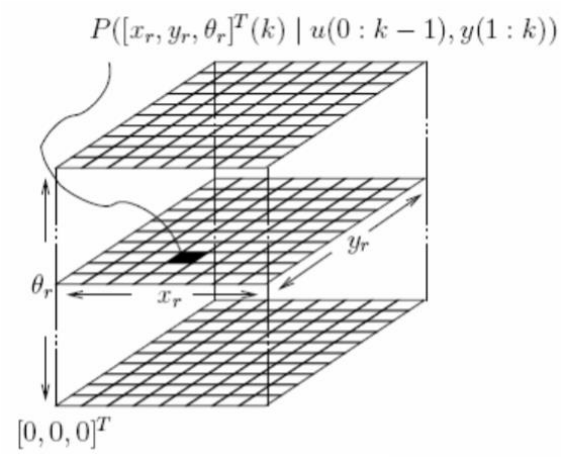
Şekil 57. Lazer İşgal Olasılık Profili

(Kaynak: Şengül, 2006)

Robot tüm haritadaki noktaları oluştururken [0,0,0] noktasına göre işlem yapmakta ve giridleri olasılıksal olarak hesaplamaktadır. Robotun konumu x_r , y_r ve θ_r olarak belirlendiğinde olasılıksal Denklem 32'deki gibi olmaktadır.

$$P(|x_r, y_r, \theta_r|^T(k)|u(0:k-1), y(1:k)) \quad (32)$$

Formul uygulandığında robotun girid şeması Şekil 58'deki gibi gösterilmektedir.



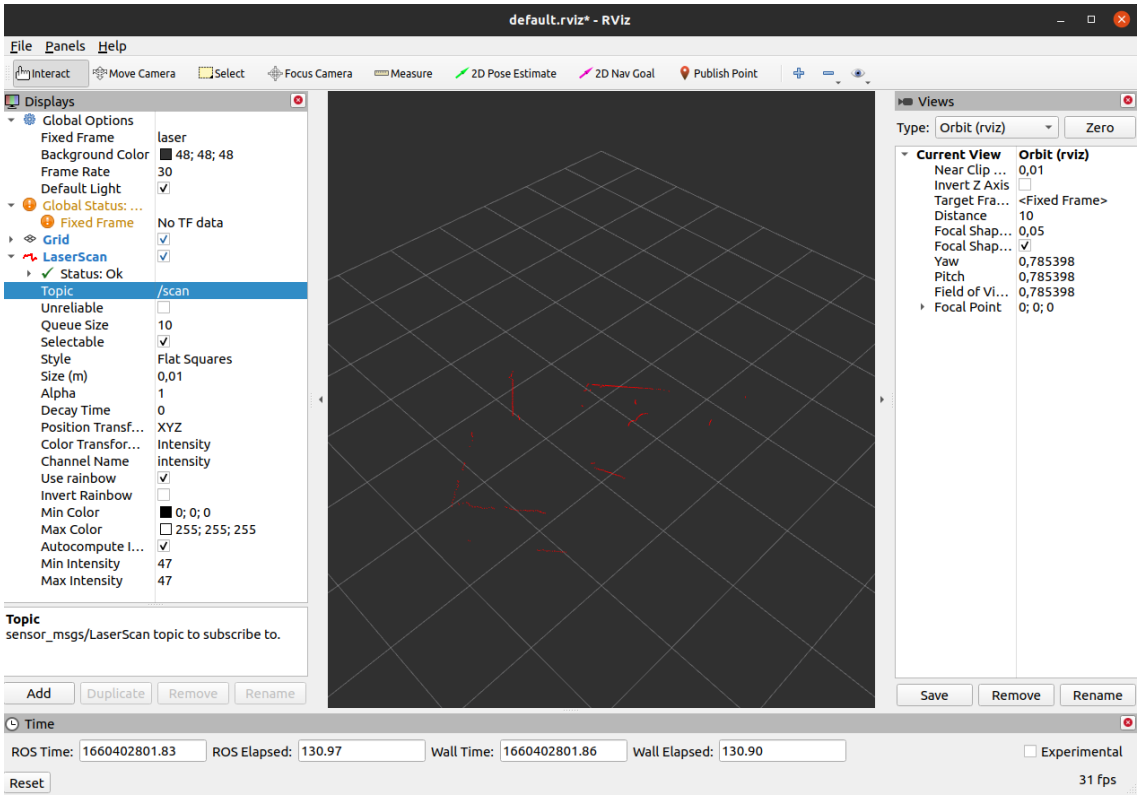
Şekil 58. Robot SLAM Grid Şeması

(Kaynak: Şengül, 2006)

Mobil robotta girid ile harita çıkarma işlemi ROS ile otomatikleşmiş, bu işlemler ROS üzerinde yüklenen paketler sayesinde standart şekilde yapılabilir hâle gelmiştir. ROS sensör bilgileri haritalama paketlerine verildikten sonra gerekli robot ayarlarının yapılması ve kullanılacak algoritmaların düzenlenmesi iyi bir harita çıkarmak için yeterli olmasını sağlamıştır.

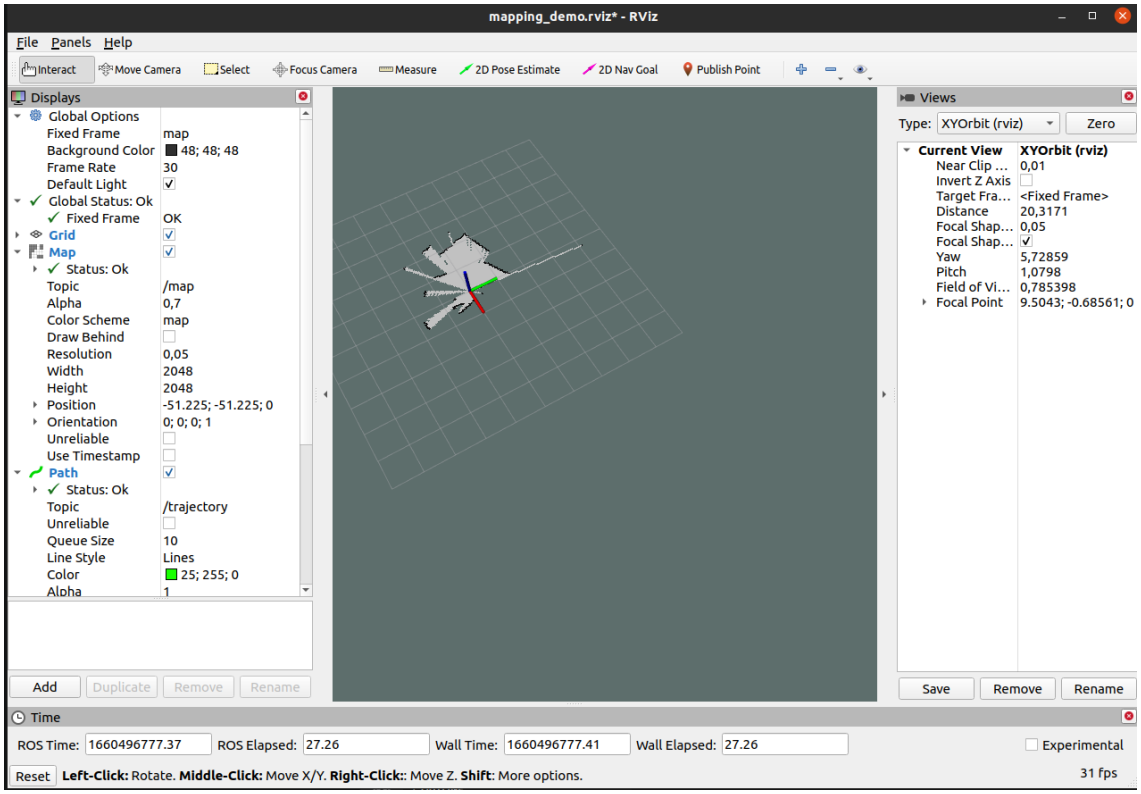
Tezde kullanılan LIDAR'ın fiyatının uygun olması ve 360 derece görüş kabiliyetine sahip olması bu cihazın seçiminde etkili olmuştur. LIDAR robotun ikinci katına yerleştirilmiş ve robot hareket ederken tüm alanı görebilmesi için etrafı boş bırakılmıştır. Bu şekilde lazer ışıkları 360 derece dağılabilmekte ve hızlı şekilde harita oluşabilmektedir. Lazer ışıkları ile harita çıkarılırken ilk olarak okunan LIDAR verisi ile ilk engele kadar ki kısımlar tam sayılarak bir harita çıkarılmaktadır. Bu haritada sadece kırmızı noktalar

verilmekte ayrıntılı bir bilgiye erişilememektedir. Bu LIDAR verisi Şekil 59'da gösterilmiştir.



Şekil 59. LIDAR Verisinin RVIZ Üzerinde Gösterilmesi

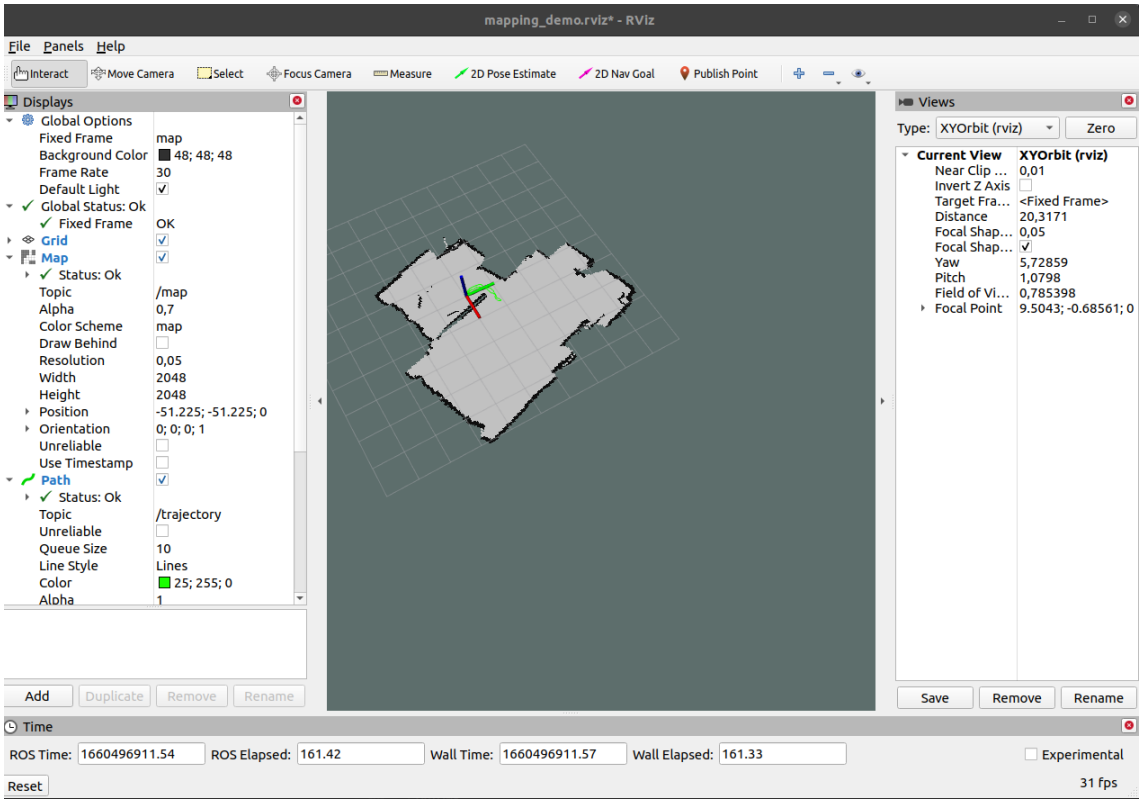
SLAM sisteminin LIDAR ile entegre çalışması sonucunda robot haritalama işlemine başlayacaktır. Robot ilk aşamada iki farklı işlem yapmaktadır. Bu işlemlerden ilki engelleri bulmak, ikincisi de lazerin uzunluğunu tespit edemediği noktaları bulmaktır. SLAM lazer ışıklarında engelle karşılaştığında beyaz noktalarla ilk engele kadarki alanı beyaz işaretler ve ilk gördüğü engeli duvar olarak siyah işaretler. Eğer LIDAR mesafesi yeten kısmına kadar herhangi bir engel bulamazsa bu durumda lazer uzunluğu kadar beyaz alan işaretlemektedir. Bu durumu anlatan görsel Şekil 60'da verilmiştir.



Şekil 60. LIDAR verisinin SLAM Algoritması ile Durağan Çalıştırılması

Robot ilk hareketi eklediği zaman bazı duvarları gittiği mesafeye ve yöne doğru kontrol ederek silmektedir. Bu sırada yeni duvarlar da eklemektedir. Robot hareket ettikçe harita da büyümekte genişlemektedir. Grid olarak ifade edilen bölmelerde bu genişleme bir tür işgale benzemektedir. Robot ilerledikçe LIDAR açısına giren yerlerdeki duvarlar silinmekte, yenileri oluşmakta ve harita genişlemektedir. Bu giridlerin oluşumları için Bayes formülleri kullanılmaktadır. Bayes formülleri LIDAR'ın ölçtüğü 360 derece ölçümleri anlık olarak gelen ve daha önce ki ölçümlere göre değerlendirilmektedir. Bu ölçümleri kullanarak anlık olarak yeni ortam tahminleri yapar. Mobil robot bu şekilde ortamın şeklini gerçeğine yakın şekilde çıkarmaktadır.

Tez kapsamında çıkarılan ilk harita Şekil 61'de verilmiştir.



Şekil 61. Tez Kapsamında Mobil Robotun İlk Haritası

Bu harita Kalman filtresi kullanılmadan çıkarılmış ve robot kısıtlı hareket etmiştir. Mobil robot ilk haritayı çıkardığında bulunduğu nokta için mavi, kırmızı ve yeşil renkleri kullanarak nokta belirtmiştir. Sonrasında hareket ederek ince yeşil çizgi ile yolu belirtmiştir. Robotun ilk odanın haritasını çıkartmasından sonra ikinci odaya geçmesi ile çok daha belirgin ikinci oda haritası hızlıca ortaya çıkmıştır.

Mobil robot ile hazırlanan haritayı kaydetmek için ROS içinde bulunan MAP server paketi kullanılmıştır. Bu işlemin yapılması için terminal üzerinde “rosrun map_server map_saver -f harita ismi” komutu çalıştırılmalıdır. Bu komut ile kullanılan çalışma alanı içinde maps dizininde bir harita dosyası oluşacaktır. Bu standart bir harita olduğu için herhangi bir zamanda farklı ROS paketleri ve programları tarafından açılarak kullanabilir duruma getirilmektedir.

2.3.5. Noktalar Arası Otonom Hareket (Navigasyon)

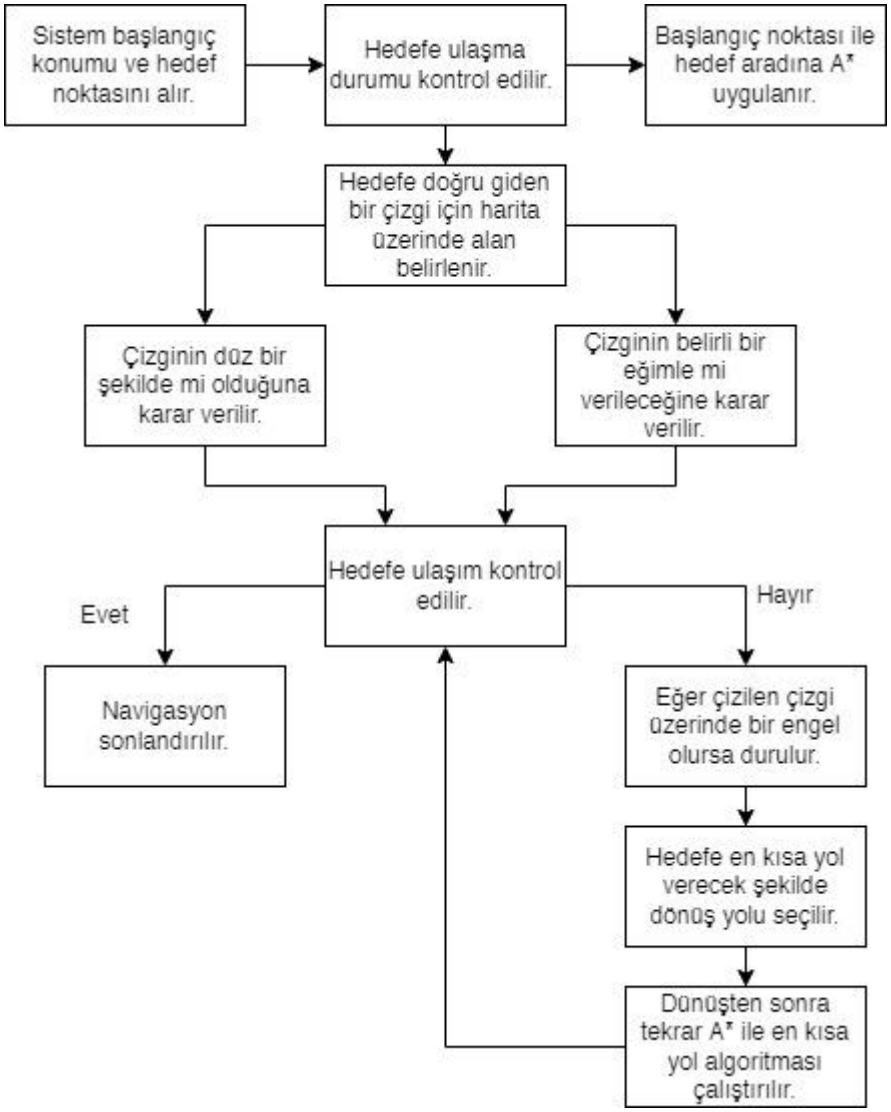
Navigasyon işlemleri için öncelikle bulunulan ortamın haritasının çıkarılması gerekmektedir. SLAM işlemi tamamlanıp harita serverda oluştuktan sonra navigasyonda kullanmak için map_server paketinde haritanın yayınlanması gerekmektedir. ROS

haritalarının yayınlanması işlemi launch dosyaları ile otomatik hale gelebilmektedir. Burada yapılan ayarlar ile harita RVIZ üzerinde açılabilmekte ve ilk konum gibi bilgiler üzerinde çalışabilmektedir. ROS aynı zamanda harita üzerinde olmayan duvarlar, bölgeler vb. çeşitli ayarlar yapılmasına da imkan vermektedir.

Mobil robotlarda navigasyon sistemi için Dijikstra, A* ve Bug gibi bazı algoritmalar geliştirilmiştir. Bu uygulamalar harita çıkarma işleminde olduğu gibi grid sistemi ile çalışmaktadır. Algoritmalar genel olarak ulaşılabilecek bir yol olduğunda gidilecek yolu bulabilmekte, fakat ulaşma hızları konusunda farklı sonuçlar verebilmektedir.

Dijikstra algoritması her zaman en optimal yolu bulmaya çalışmaktadır. Her zaman en optimal yola baktığı için genellikle işlem süreleri çok uzun sürmekte, çok fazla kaynak tüketilmesine sebep olmaktadır. Bug algoritması düz şekilde hedefe yönelmekte ve ilerlemeyi hedeflemektedir. Bu algoritmada bir engelle karşılaşıldığı zaman her zaman belirli bir yöne doğru yönelerek sonuca ulaşmak amaçlanmaktadır. Bu algoritmada her zaman en kısa sonuca ulaşılacağı garanti edilmemektedir. Ayrıca harita karmaşıklaştığında yolu bulmak çok zaman ve enerji kaybına sebebiyet vermektedir. A* harita üzerinde kendisi ile gideceği yol arasında bir çizgi çizer. Bu çizgi eğer bir engel ile karşılaşırsa o zaman engelin etrafından en kısa yol ile dolanmaya çalışmaktadır. A* algoritması da en kısa yolu bulacağını garanti etmemektedir. A* en kısa yolu bulmayı garanti etmese de yaptığı işlemleri hızlı şekilde yaparak optimal kaynak harcamayı amaçlamaktadır.

Tez kapsamında yapılan mobil robotta hız ve etkinlik açısından optimal olan A* yol bulma algoritması kullanılmıştır. Algoritmada amaçlanan hedefe en hızlı şekilde ve en az kaynak tüketerek ulaşmaktır. Bu amaçla harita üzerinde robotun konumu ve hedef konum seçilir. Bu iki konum arasında bir çizgi çizilerek yol belirlenir. Eğer çizgi üzerinde bir engel ile karşılaşılırsa engel taranarak en kısa yola doğru yönelme yapılmaktadır. Bu şekilde tüm harita engellerle karşılaşana kadar taranıp uygun yol tespiti yapılmaktadır. A* algoritması için yapılan işlemler Şekil 62'de gösterilmiştir.



Şekil 62. Robot Navigasyon Çalışması Akış Diyagramı

A* algoritması ile mobil robot ilk engele kadar düz gitmektedir. Bu şekilde ilerledikten sonra ilk engelle karşılaşılması sonucu engelin en kısa yoldan dolanarak hedefe ulaşmayı amaçlamaktadır.

ROS ile çalışırken Navigasyon çalışması için öncelikle tüm sensörlerin sisteme bağlanması ve odom verisinin oluşturulması gerekmektedir. Odom verisi ile oluşturulan bu veriler A* algoritmasına ve navigation_data_pub paktine gönderilir. Bu iki paket üzerinden üretilen cmd_vel verisi robota gönderilerek robotun kontrolü sağlanmaktadır. Tez kapsamında server ile robot birbirinden farklı sisemler olduğu için navigasyon için gerekli A* uygulamasını server kullanmakta ve robota verileri göndermektedir. Robot aldığı verileri gömülü sistemlerini kullanrak harekete dönüştürmektedir. Robot üzerinde

bulunan sensörler verileri tekrar toplayarak servera göndermekte, robot bu şekilde navigasyon işlemini yapmaktadır.

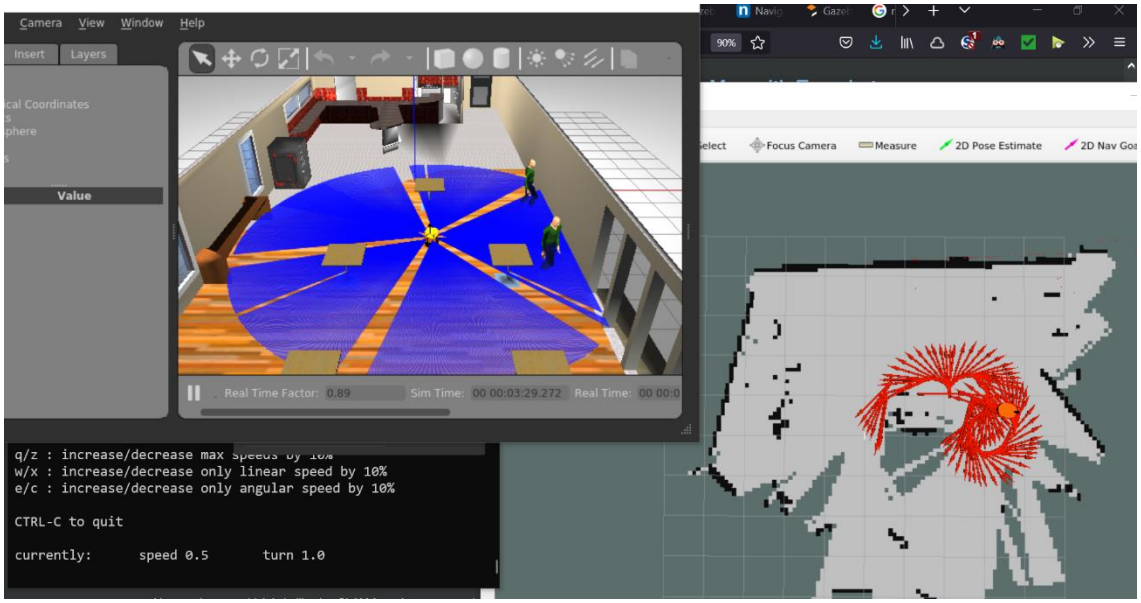
2.3.6. Simülasyon Çalışması

Simülasyon çalışmaları sistemlerin çalışmasını görmek ve test yapabilmek için önemlidir. Tez kapsamında ROS ile LIDAR, IMU ve enkoderler yardımı ile çalışan bir simülasyon çalışması yapılmıştır. Simülasyon çalışmasında robot olarak Şekil 63'de verilen EVO robot kullanılmıştır. Bu robot ROS sitesinde paylaşılan bir robottur ve ROS ile tam entegre yapıya sahiptir.

Şekil 63. EVO Robot

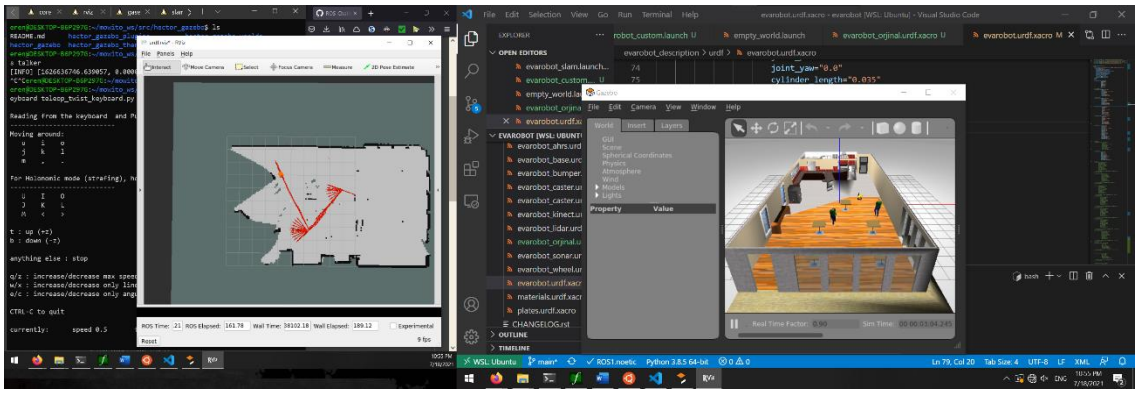
(Kaynak: ros.org)

Robot çalışma ortamı, tezdeki yaklaşımdan farklı olarak her şeyin robot üzerinde olmasına göre hazırlanmıştır. Bu sistemde haritalama ve navigasyon doğrudan robot üzerinden ve GAZEBO simülasyon programı ile yapılmaktadır. Oluşturulan haritalar ve konum verme işlemleri ise RVIZ üzerinden yapılabilmektedir. Şekil 64'de simülasyon çalışmasında GAZEBO üzerinde hareket ettirilen robotun harita çıkarması işlemleri gösterilmektedir.



Şekil 64. Simülasyon Haritalama Çalışması

Robot haritalandırma işlemleri tamamlandıktan sonra, Navigasyon sistemi için haritada nokta belirleme işlemleri yapılmaktadır. Robotun gidebildiği alanlar içinde seçilecek bir noktaya doğru robot öncelikle bir yol çizer. Yol çiziminin tamamlanmasının ardından robot noktaya yönelim sağlayarak ilerlemektedir. Navigasyon çalışması ile ilgili ekran görüntüleri Şekil 65'te paylaşılmaktadır.



Şekil 65. Simülasyon Navigasyon Çalışması

Simülasyon çalışması sonucunda EFK kullanılmadan haritalama ve navigasyon sistemlerinin çalışmaları test edilmiştir. ROS paketleri simülasyon üzerinde çalıştırılarak geliştirme ortamının doğru olduğu ve server bilgisayarın haritalama ve navigasyon

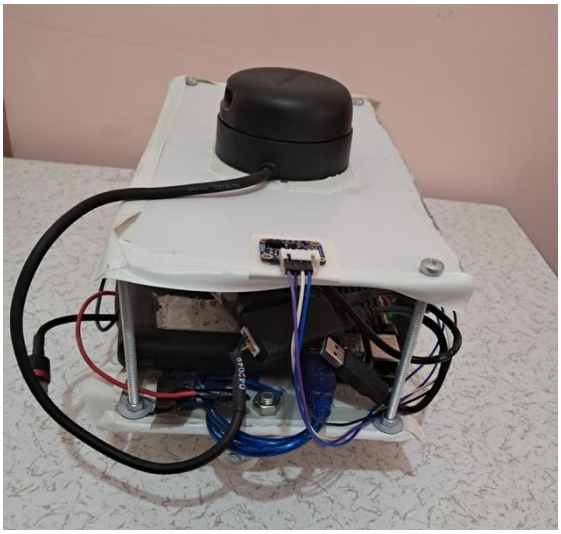
yapabilmek için gerekli yazılım paketlerine ve bilgisayar donanımına sahip olduğu görülmüştür. Simülasyon ortamında ROS ile tam uyumlu bir robot kullanıldığı için tezde yapılması gereken TF ve odom ayarlamaları yapılmamıştır. Bu ayarlar sistemde bulunduğundan çalıştırıldığında sisteme doğrudan entegre olmuştur.

Simülasyon çalışmasının sonucu tez için yapılan prototip çalışmasının önünü açmış ve ROS ile ilgili ilk yol haritasının oluşturulmasına olanak sağlamıştır.

2.3.7. Robot ile SLAM ve Navigasyon Çalışması

Navigasyon işlemleri mobil robotlar için son yıllarda en önemli konulardan biri hâline gelmiştir. Bu işlemlerin yapılması sayesinde robotlar bilinmeyen bir ortamı tanıyabilmektedir. Ortam tanındıktan sonra robot bu alan içerisinde verilen noktalara otonom olarak giderek, gerekli görevleri yapabilmektedir. Örnek uygulama olarak depolarda kullanılan AMR tipi robotlar verilebilir.

Otonom gezinme işlemlerinin yapılabilmesi için alt sistemlerin entegrasyonuna ihtiyaç vardır. Mekanik sistemlerin navigasyona uygun şekilde tasarlanması, elektronik için gerekli pahalı sensörlerin alınması, güç sistemlerinin planlanması ve yazılımsal olarak sistemin çalışır hâle gelmesi gerekmektedir. Yazılım konusunda tez çalışmasında robotun üzerindeki bilgisayardan verilerin alınması ve uzak sunucu üzerine gönderilmesi planlanmıştır. Robotun haritalama ve navigasyon işlemleri bu uzak server (sunumcu) bilgisayarında yapılacaktır. Uzak serverın işlemci ve bellek olarak yüksek kapasitede olması yapılacak işlemlerin hızını arttıracaktır. Tez kapsamında yapılan haritalama, navigasyon ve haberleşme işlemleri için ROS altyapısı ve paketleri kullanılmıştır. Tez için simülasyon çalışmasının yanı sıra bir de gerçek bir robot ile prototip çalışması yapılmıştır. Mobil robot üzerine LIDAR, IMU ve enkoder gibi sensörler yerleştirilmiş. Verilerin alınabilmesi için gerekli bilgisayar ağ bağlantı ayarlamaları yapılmıştır. Geliştirlen mobil robot Şekil 66'da gösterilmiştir.



Şekil 66. Tez Kapsamında Geliştirilen Robot

ROS un geliştirilmesi ve kullanımının son yıllarda artması ile mobil robotlarda haritalandırma ve navigasyon işlemlerinde standart yapılar oluşmuştur. ROS navigasyon paketi çok kolay ayarlanabilen bir sistem olarak geliştirilmiştir. Bu paket mobil robotun özelliklerinin config dosyaları ve TF verileri ile belirlenmesi ile kolay şekilde işlemlerin yapılmasına olanak sağlamaktadır. ROS paketleri standart bir yapıya sahip olduğu için kullanılan yazılımlar ve yazılım paketleri mobil robotun kullandığı sensör verilerine kolayca ulaşabilmektedir. ROS içinde gelen launch paket yapısı sayesinde tüm veriler ve paketler gerekli sıra ve parametrelerle çalıştırılırlar. ROS'un çıkardığı sonuçlar yine ROS ile entegre olabilen RVIZ gibi GUI programları ile anlık olarak izlenebilmektedir. ROS launch çalıştırıldığı zaman tüm yazılımlar, sensörler otomatik olarak çalıştırılır. Otomatik olarak açılan harita üzerinden robot ilk anda bulunduğu konumun haritasını çıkarmaya başlamaktadır. Robot teleop_twist_keyboard isimli bir ROS paketi yardımı ile klavye üzerinden gezdirilebilmektedir. Bu gezdirme işlemi sonucunda ilk haritalama gerçekleştirilmiştir.

Navigasyon ile ilgili ROS launch çalıştırıldığında ise ekranda yine RVIZ görünmekte, robot daha önce çıkarılmış harita üzerinde konumlandırılmaktadır. Bu aşamada kullanıcı RVIZ üzerinde hedef ve pozisyon bilgisini fare yardımı ile verir. Robot yolunu çıkardıktan sonra ilgili konum ve oryantasyona doğru otonom hareketine başlamaktadır.

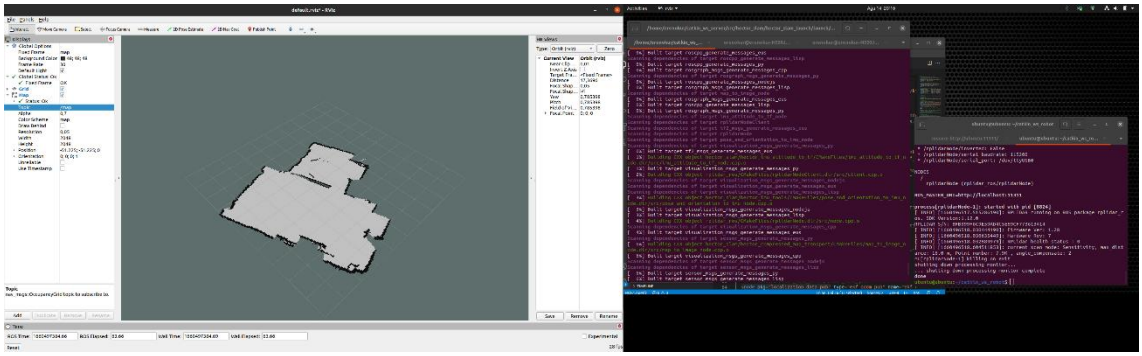
Mobil robot ile SLAM ve navigasyon Kalman filtreleri ile yapıldığı zaman sensörlerden gelen veriler birleştirilip çok daha belirgin bu verilerle işlemlerin doğruluk oranları

artmaktadır. Robot Kalman filtresini kullanarak başlatılmak istendiğinde launch dosyalarını değiştirmek yeterlidir. Robot verileri doğrusal olmadığı için genişletilmiş Kalman filtresi EKF sistemi kullanılmaktadır. Launch dosyasına EFK paketinin çıkarılması ve bu pakete sensör topik bilgilerinin verilmesi sistemin çalışması için yeterli olmaktadır. Kalman ile ilgili gerekli tüm işlemler yeni bir topikte yayınlanmaya başlamaktadır. Bu paketleri kullanmak için de yazılımlara ve yazılım paketlerine bu çıktıların iletilmesi gereklidir. Robot çalışma ortamına göre hazır bekleyen durumu Şekil 67'de paylaşılmıştır.

Şekil 67. Robot Çalışma Ortamı

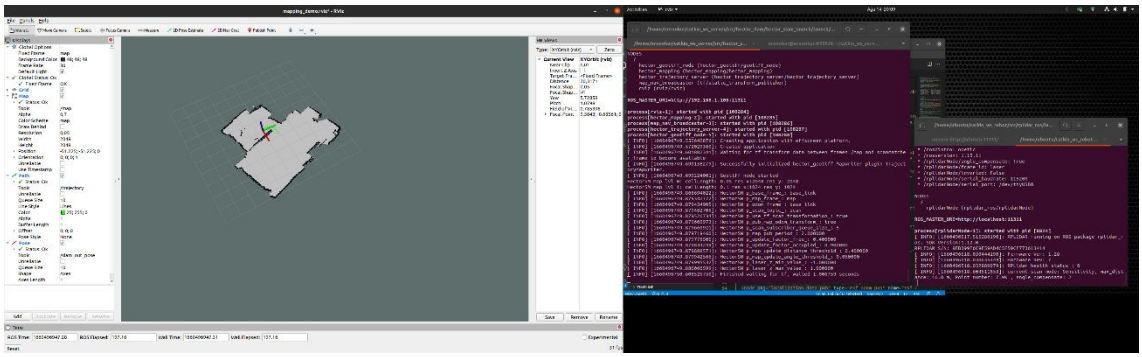
ROS üzerinde SLAM ve Navigasyon işlemlerine başlanırken robot kontrol edilmelidir. Robotun herhangi bir noktasında sorun olmadığı belirlendikten sonra robota güç verilmesi gerekmektedir. Robot işleme başlatıldıktan sonra terminal üzeriden sensör erişim izinleri verilir ve paketlerin haberleşmesini sağlamak için launch dosyası çalıştırılır. Bu noktadan sonra robot ile ilgili işlemler server üzeriden devam edilmektedir. Robot ilk aşamada herhangi bir bilgiye sahip olmayıp sadece sensör bilgilerini göndermekte ve serverdan gelen bilgilerle hareketini sürdürmektedir. Serverda ise dört farklı launch dosyası bulunmaktadır. Bu launch dosyaları EKF li ve EKF siz haritalama ve navigasyon işlemlerini yapmak için oluşturulmuştur.

Geliştirlen robot açıldıktan sonra sevrer tarafında EKF'sız haritalama işleminin yapılması için ilk aşamada haritalama launch dosyası çalıştırılır. Bu launch çalıştığı anda RVIZ otomatik olarak açılarak LIDAR'ın ölçümleri ile oluşturulan bir haritayı kullanıcının erişimine sunar. Bu aşamadan sonra farklı bir terminal üzerinde twisted_teleop_keyboard modülü çalıştırılır. Bu modül doğrudan tekerleri yönetmek için kullanıldığından robot bu paket yardımı ile klavye ile haritası çıkarılacak alanda gezdirilir. Gezinme tamamlandıktan sonra, server ekranında RVIZ üzeride çıkarılan harita ile gerekli kontroller yapıldıktan sonra map_server ile harita kaydedilir. Şekil 68'de bu durum belirtilmektedir.



Şekil 68. Haritanın Oluşturulması

Bu işlem EKF ile yapıldığı zaman kullanılan launch dosyasında EKF modülü çağırılır ve ilgili parametreler bu modüle verilir. Bu işlemle hazırlanan EKF harita launch dosyası çalıştırılır. Sistem çalışınca RVIZ otomatik açılarak oluşan haritayı kullanıcıya göstermeye başlar. Kullanıcı klavye ile robotunu kontrol ederek etrafta gezdirir ve harita oluşturulur. İşlem tamamlandıktan sonra haritayı map_server paketi ile kaydeder. (Şekil 69)



Şekil 69. Map Server ile Haritanın Kaydedilmesi

Haritalama işlemlerinde robotun haritasını çıkardığı referans alınan test ortamı üç ayrı bölmeden oluşmuştur. Bu alanlar sırasıyla 30 m^2, 45 m^2 ve 75 m^2 alanlarına sahiptir. Test yapılan ortamlarda mobilyalar bulunduğu için bu mobilyaların boyutları toplam test ortamlarından çıkarılarak işlem yapılmıştır. Robotun etrafta gördüğü dolap, koltuk vb. eşyaların kapladığı alanlarda sırasıyla $1m^2$, 3 m^2 ve 4 m^2 mobilya olduğu yapılan ölçümlerde tespit edilmiştir. Ölçümlere göre mobilyaların alanları test ortamının alanından çıktığında ölçüm sonucu olarak bulunması gereken boyutlar 29 m^2, 42 m^2 ve 71 m^2 olmalıdır.

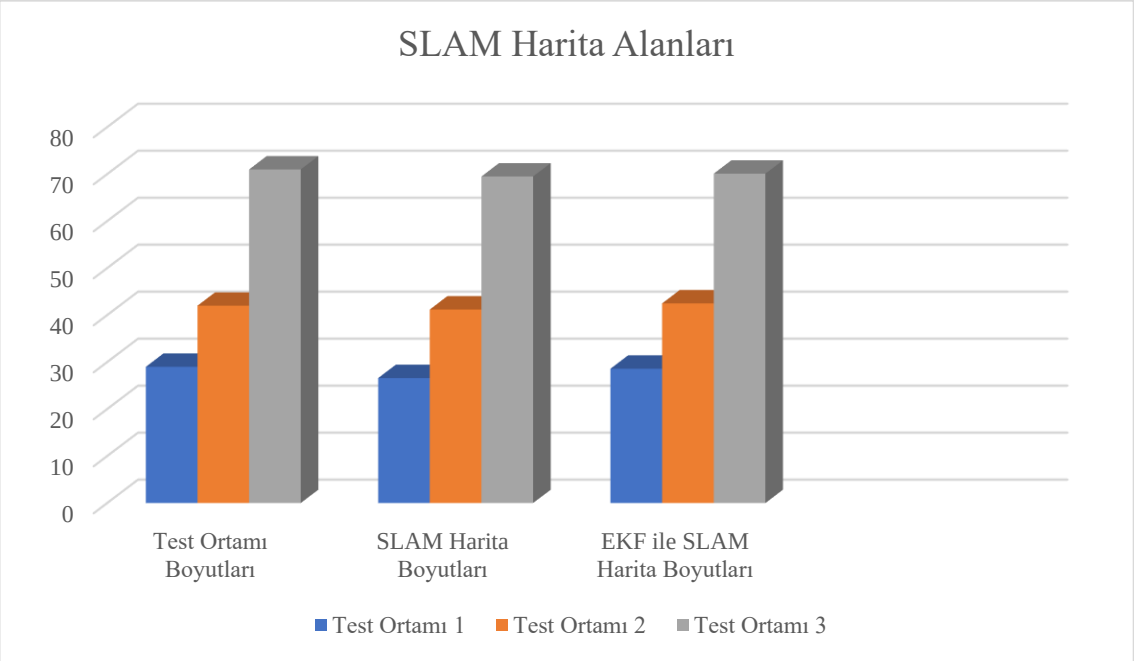
Test ortamında mobil robot hem EKF ile hem de EKF filtresi olmadan haritalama işlemi yapmıştır. Testler sırasında robotun mümkün olduğunca aynı yoldan ilerlemesi sağlanmıştır. Her test için 5 dakikalık süre belirlenmiş, robotun mümkün olduğunca benzer noktada durması sağlanmıştır.

Test sonucunda EKF ile EKF'sız robot haritalarında gözle fark edilebilen değişim olmamıştır. İki sistem arasında farklılıklar alan hesapları ile bulunmuştur. Her iki sistemde aynı robot, aynı alanı, aynı zamanda haritalamış; Fakat EKF ile çalışan robot çok daha doğru sonuçlar bulmuştur.

Robot belirtilen 29 m^2, 42 m^2 ve $71m^2$ test alanında EKF ile çalıştırıldığında alanları 28,6 m^2, 42,5 m^2 ve 70,1 m^2 olarak bulmuştur. Buna karşılık olarak EKF'siz yapılan testlerde 29,6 m^2, 41,2 m^2 ve 69,5 m^2 alan ölçümü yapılmıştır. Bu testler EKF ile yapılan testlerin gerçek alanlara göre 0,4 m^2 eksik, 0,5 m^2 fazla ve 0,9 m^2 fazla sonuç bulduğu gözlemlenmiştir. EKF'siz yapılan testlerde ise aynı ortamda 0,6 m^2 fazla, 0,8 m^2 eksik ve 1,5 m^2 eksik alan bulunmuştur. Bu durum da EKF kullanılan testlerde hata oranlarını % 1,3, % 1,2 ve % 1,4 olarak göstermiş ortalama hata % 1,3 olmuştur. EKF'siz

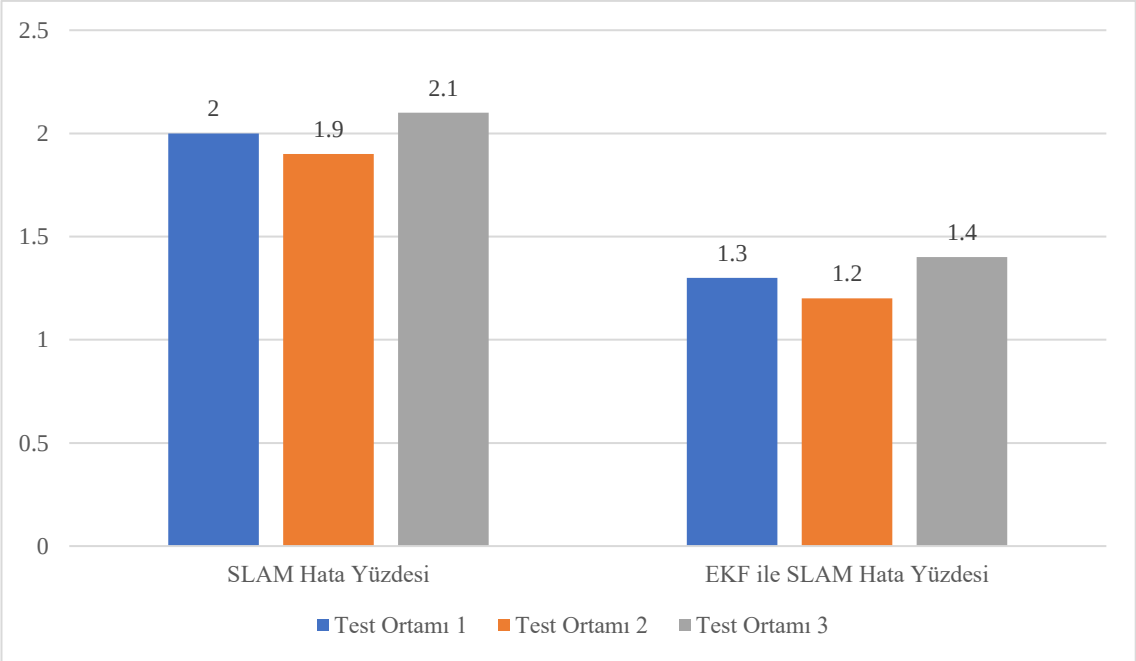
testlerde ise hata oranları % 2, % 1,9 ve % 2,1 olarak tespit edilmiştir. EKF'siz sistemde ortalama %2 hata oranı bulunmuştur. EKF ile yapılan testlerde EKF'siz testlere göre % 0,7 daha az hata oranı tespit edilmiştir.

Şekil 70'de farklı test alanları için mobil robotun oluşturduğu haritaların boyutları gösterilmiştir. EKF kullanan SLAM algortiması aynı şartlarda EKF'sız algoritmaya göre daha doğru sonuca ulaşmıştır. Hata miktarları bazı testler için fazlalık olarak ortaya çıkarken bazı durumlarda eksik olarak ortaya çıkmıştır.



Şekil 70. Test Ortamlarında Yapılan Harita Alan Farkları

Hata oranlarına bakıldığı zaman EKF ile çalışan haritalama sistemi ortalama % 1,3 hata payı ile çalışırken, EKF'sız çalışan haritalamada hata payı ortalama % 2 olmuştur. Hata yüzdeleri test yapılan 29 m^2, 42 m^2 ve 71 m^2 alanlara göre hesaplanmıştır. Bu çalışma ile harita çıkarma işleminde Kalman filtrelerinin kullanılmasının faydalı etkileri gözlemlenmiştir. Yapılan haritalama işlemlerinin sonucu oluşan hatalar Şekil 71'de gösterilmiştir.



Şekil 71. Haritalama Hata Yüzdeleri Karşılaştırması

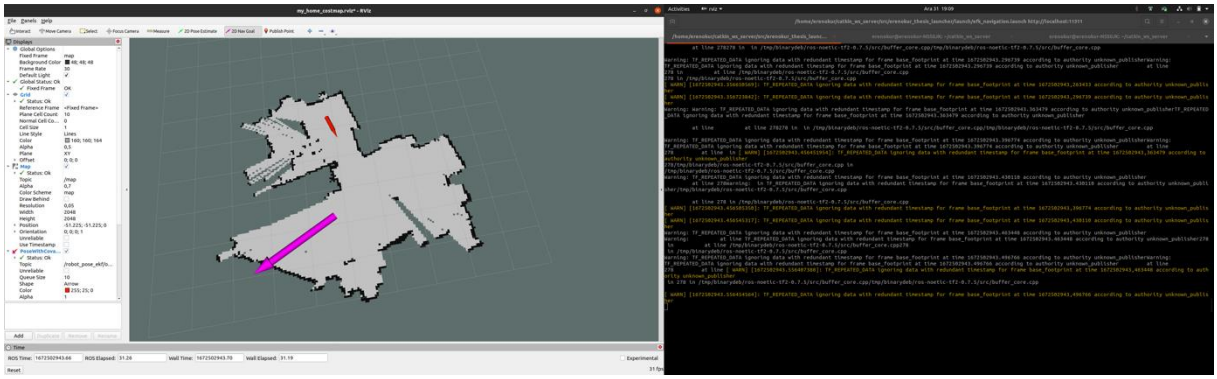
Mobil robotlarda harita çıkarma işlemi, navigasyon yapılacak ortamın doğru bir haritası olmaması durumunda oldukça önemli bir işlemdir. Navigasyon işlemi çalışması için bir haritaya ihtiyaç duyulmakta ve bu haritanın mümkün olduğunca doğru olması gerekmektedir. EKF ve EKF olmadan harita hazırlanması mobil robotla navigasyonun çalışması için yeterli kriterleri sağlamaktadır. Gerekli haritalama işlemi sağlandıktan sonra navigasyon işlemine başlanmaktadır.

Tez kapsamında otonom navigasyon işlemi için iki adet launch dosyası geliştirilmiştir. Bu dosyalarda A* algoritması ile EKF ile ve EKF olmadan navigason işlemi yapılmış sonuçları karşılaştırılmıştır. Navigasyon işleminde robotun hedef noktaya ulaşması esas alınmış ve o noktaya tam oturması göz ardı edilmiştir. Normalde robot hedef noktaya ulaştıktan sonra kendisini verilen pozisyona göre konumlandırmakta ve noktaya vardığını teyit etmektedir. Bu çalışma kapsamında robotun sadece noktaya gitmesi kontrol edilmiş konumlama ve teyit etme aşamaları hesaplamalarda dikkate alınmamıştır.

Navigasyon için önemli konulardan biri de konumlandırma işlemleridir. Mobil robot navigasyon işlemine ilk başladığında nerede olduğu kendisine kullanıcı ya da tanımlanmış farklı bir sistem tarafından bildirilmektedir. Robot bu ilk tanımdan sonra tam olarak nerede olduğuna kendisi karar vermek zorundadır. Bu işlem sırasında robotun en önemli verisi odom tarafından verilmektedir. Odom robotun bir noktadan başka bir

noktaya yaptığı hareketi verdiği için ortamdaki diğer değişkenler hakkında bilgi vermez. Robotun kendisini konumlandırması ve etraftaki farklı nesnelere çarpmasının önlenmesi için odom verisinden bağımsız olarak LIDAR, IMU verisinin izlenmesi en yakın uzaklıkların tespiti gereklidir. Bu bilgi robotun LIDAR ve IMU odom verisinde kullanılmasa da harita üzerinde engellerden kaçınmada son derece etkindir.

Navigasyon işlemi için robot çalışır hâlde olduğu ve haberleşme sistemleri kontrol edildikten sonra server üzerinde navigasyon için hazırlanmış launch dosyaları çalıştırılarak yapılmaktadır. Server üzerindeki launch dosyası çalıştırıldığında sistem otomatik olarak robota bağlanmakta ve tüm sensör paketlerine erişebilmektedir. Sistemin çalıştırılmasından sonra otomatik olarak açılan RVIZ üzerindeki haritada robotun konumu ve hedef noktası seçilmelidir. Seçim işlemi yapıldığında robot ilk aşamada gerekli hesaplamaları yapar ve otonom hareketine başlar. RVIZ üzerinde robot konumu ve hedef belirleme işlemleri Şekil 72'de paylaşılmıştır.



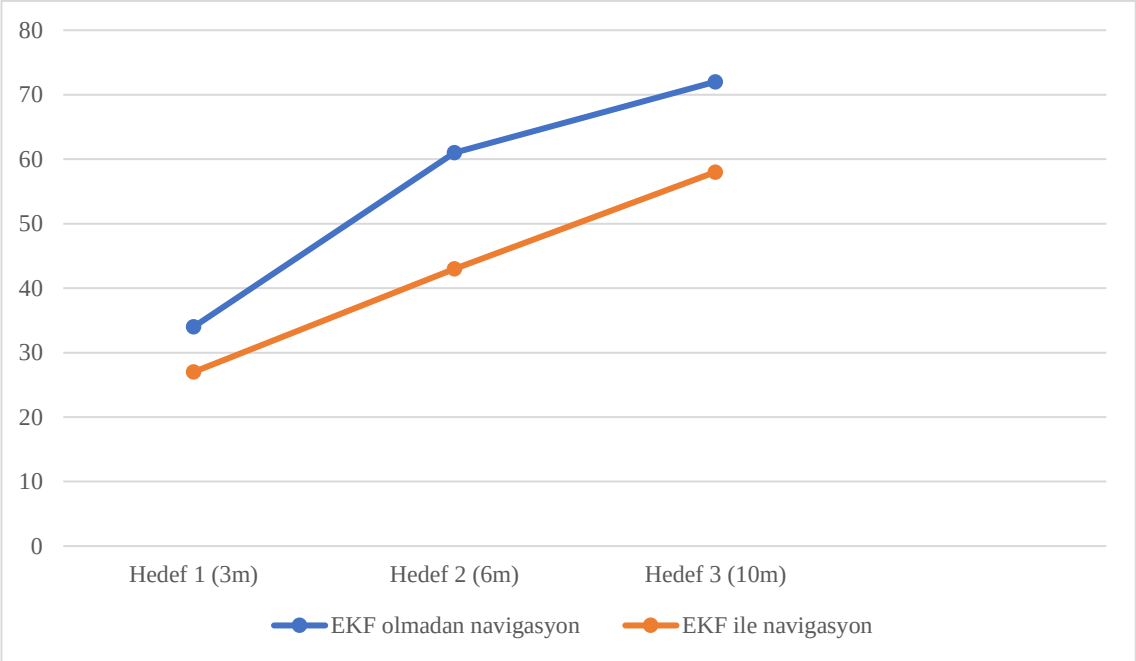
Şekil 72. Robota Navigasyon ile Hedef Konum Verme

Navigasyon için geliştirilen EKFsız navigasyon işleminde launch dosyası içinde odometri hesaplarında Kalman filtreleri çağrılmaz. Odom bu sistemde sadece enkoderlere bakarak konum belirlemeye çalışmaktadır. Sistem odomertri hesabı yapılırken LIDAR ve IMU verileri hesaba katılmaz. Odom verisi düzenlendikten sonra sistem yine A* ve locatization_data_pub paketleri çalıştırılarak gerekli sensör bilgileri bu yazılım paketlerine aktarılır. LIDAR, IMU bilgilerine engelden korunmak için anlık olarak bakılmakta ve robotun çarpışmadan korunulması amaçlanmaktadır. Robot konumuna geldiğinde kendi pozisyonunu ve konumunu doğrulamakta işlemi sonlandırmaktadır.

EKF sisteminde ise odometri hesaplarında tüm sensör bilgileri dikkate alınır. Bu sistemde Kalman filtreleri sistemdeki tüm sensörleri kontrol ederek anlık olarak en doğru konumlandırma işleminin yapılmasını sağlamaktadır. Robot açıldıktan sonra odometri hesabını A* ve localization_data_pub yazılım paketleri ile paylaşarak robotun hareketini sağlar. Robot otonom hareketi sırasında çarpmaya karşı korunmak için LIDAR ve IMU bilgilerini ayrıca kontrol etmeyi sürdürmektedir. Tüm işlemler sonlandığında robot pozisyon ve konumunu kontrol ederek işlemlerini sonlandırmaktadır.

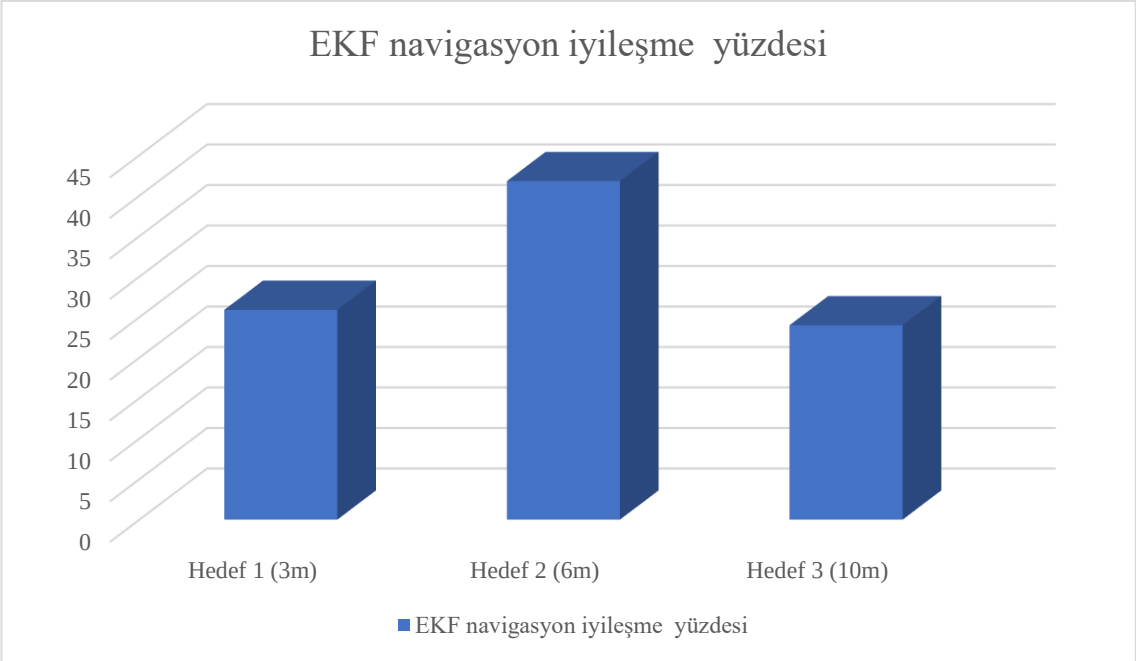
Tez kapsamında geliştirilen mobil robotta navigasyon işlemi için bir harita üzerinden üç farklı nokta seçilmiş ve o noktaya ulaşma süreleri hesaplanmıştır. Tüm harita üzerinde belirlenen üç noktadan ilki robot konumu ile aynı odada robottan 3 metre uzak mesafede; ikinci verilen nokta haritanın ortasına yakın bir konumda robota 6 metre mesafede; hedef olarak verilen son nokta da haritanın uzak noktasında robottan 10 metre mesafede verilmiştir.

EKF ile yapılan işlemde noktalara ulaşma süresi yakından uzağa doğru 27 sn, 43 sn ve 58 sn olarak belirlenmiştir. EKF olmadan yapılan ölçümlerde ise süreler 34 sn, 61 sn ve 72 sn olarak ölçülmüştür. Ölçümler sırasında robotun konuma ulaşma sürelerine bakılmış konumlanma işlemleri dikkate alınmamıştır. Robotun sürelerindeki farklılıklar robotun dönüş hereketi yapması ve yolunu kaybedip tekrar bulma süreleri dâhil edilerek hesaplanmıştır. Robotun otonom navigasyon hareket süreleri Şekil 73'te gösterilmektedir.



Şekil 73. Navigasyon ile Hedefe Ulaşma Hata Yüzdesi

EKF ile robot navigasyon işlemi sırasında konumlara göre iyileşme sağlanmıştır. Bu iyileşmeler 3 m olarak verilen konum için 34 sn'den 27 sn'ye düştüğü için iyileşme % 25,9 olarak bulunmuştur. Robottan 6 m uzakta verilen hedef için 43 sn'den 61 sn'ye düştüğü için % 41,8 iyileşme olmuştur. Son olarak 10 metre mesafe için 58 sn'lik ulaşma süresi 72 sn süreye indirilerek % 24 iyileşme sağlanmıştır. Robotun otonom navigasyon hareketinden kaynaklanan yüzde iyileşme Şekil 74'te gösterilmektedir.



Şekil 74. EKF ile Navigasyonun İyileşme Yüzdesi

Robot ile çalışma sırasında gerekli kontroller yapılırken robot sensörlerinin doğru çalıştığından emin olunması, haritalama ve navigasyon işlemleri için kritik öneme sahiptir. Tez kapsamında yapılan robotta tüm işlemler server üzerinden yapıldığı için robota erişim olduğuna bakılmalı; haberleşmede herhangi bir sorun olmadığının belirlenmesi gerekmektedir. Haberleşmede ve sensörlerde sorun olmadığı gözlemlendiğinde server tarafında gerekli işlemlerin başlaması için launch dosyalarının çalıştırılması yeterlidir. ROS ile navigasyon ve haritalama işlemleri hazırlanan launch dosyaları ile kolaylıkla ayarlanabildiği için işlem süreçleri kolay bir şekilde tamamlanmaktadır.

3. SONUÇ

Bu tez çalışmasında mobil robotlarda Kalman filtreleri kullanılarak daha etkin iç alan haritalandırma ve navigasyon işlemleri yapılabilmesi için çalışmalar yapılmıştır. Kalman filtrelerinin lineer sistemlere uygun olmaması nedeniyle verilerin lineerleştirilmesine göre işlem yapan Genişletilmiş Kalman Filtresi EKF kullanılmıştır.

Testlerin yapılması için bilgisayar üzerinde bir simülasyon çalışması ile bir adet fiziki mobil robot geliştirilmiştir. Yapılan simülasyon çalışması ile haritalama ve navigasyon işlemleri tez için geliştirilen yazılımlar ve ROS paketleri ile yapılmıştır. Projede gerekli olan tüm haberleşme işlemleri ROS haberleşme yapısı üzerinden gerçekleştirilmiştir. ROS serial yazılım paketi ile gömülü sistemlerden gelen paketler okunarak ROS master ile bilgisayar içi ve bilgisayarlar arası haberleşme sağlanmıştır.

Simülasyon çalışmasının tamamlanmasından sonra üzerinde LIDAR, IMU ve iki adet enkoder bulunan bir mobil robot geliştirilmiştir. Mobil robotun enerji tüketimini azaltmak ve işlem kapasitesini etkin kullanmak için Raspberry Pi 4 bilgisayar kullanılmıştır. Bu bilgisayarın ana görevi robot üzerindeki gömülü sistemler yardımı ile sensör bilgilerinin okunmasıdır. Robot bilgileri toplandıktan sonra, enerji kısıtlaması olmayan bir uzak servera göndermektedir. Bu bilgiler server tarafından işlenerek haritalama ve navigasyon işlemleri yapılmaktadır. Server otonom navigasyon işlemi sırasında mobil robota doğrudan teker emirleri göndermektedir.

Mobil robot çalıştırıldıktan sonra genel kontrolleri yapılır. Kontroller tamamlandıktan sonra server ile iletişimleri kontrol edilir. Tüm kontroller yapıldıktan sonra server üzerinde robot için hazırlanan launch dosyaları çalıştırılır. Bu launch dosyalarında robotun işlemleri yapması için gerekli yazılımlar ve ROS paketleri çalıştırılır. Launch dosyalarında ayrıca iç alan haritalandırma, navigasyon, güzergâh (trajectory) hesapları, odometri hesapları gibi karmaşık işlemler için gerekli ayarlamalar yapılır. Robot işleme ilk başladığı zaman hangi yazılımın nasıl paket göndereceği ve bu paketleri kimin kullanacağı bu launch dosyaları ile belirlenir.

Haritalama işlemi başladığı zaman robotun tekerlek kontrolü kullanıcı tarafından yapılmalıdır. Kullanıcı robotu haritalanacak alanda gezdirirken harita otomatik olarak değişmekte anlık sonuçlar server ekranındaki RVIZ programı üzerinden izlenebilmaktedir. Robot haritalama işlemini yaptığında oluşan haritaların test ortamının

gerçek haritasına göre farkı olmadığı görülmüştür. Çıkarılan haritalarda sadece alan hatalar tespit edildi. Haritalama işleminde yapılan testlerde;

- EKF ile oluşan haritanın gerçek haritaya göre % 1,3 hata payı olduğu saptanmıştır.
- EKF olmadan yapılan haritalama işleminde gerçek haritaya göre % 2 hata oranı olduğu tespit edilmiştir.
- Bu sonuçlara göre EKF kullanıldığında robotun hata oranı % 53,8 oranında düşürülmüştür.

Mobil robotta navigasyon işlemi başlatılması için launch dosyasına haritanın yolu verilmemiştir. Sistem tüm ayarlar yapıldıktan sonra çalıştırıldığında server ekranında RVIZ otomatik olarak açılmıştır. Kullanıcı bu aşamada RVIZ üzerinde daha önce çıkarılmış harita üzerinde robotun anlık konumunu seçer. Robot konumu tanımlandıktan sonra kullanıcı mobil robotun gitmesini istediği konumu RVIZ üzerinden verir. Robotun konumu ve hedefi verildikten sonra tüm işlemler otonom olarak başlamaktadır. Robot A* algoritması ile en kısa yolu bulur ve bu yola doğru yönelir. Navigsyon işleminin gerçekleşmesi için yapılan işlemin zamanına bakılmıştır. Robot hedef konuma geldiğinde kendisini konumlandırmakta ve yön vermesi zaman aldığından hedef noktaya ilk gelmesi testlerde referans alınmıştır. Navigasyon işlemi için yapılan testlerde;

- Robotun konuma ulaşmasında EKF uygulamalarında 10 metre mesafe için 14 sn' lik iyileşme sağladığı görülmüştür.
- EKF ile yapılan testlerin ortalaması alındığında % 30,5 daha hızlı hedefe ulaşıldığı tespit edilmiştir.

Robotun çalışması için enerji kullanımı oldukça önemlidir. Enerjinin etkin kullanımı robotun çalışma süresini artıracaktır. Robot yaptığı işlemleri kendisinden çok daha güçlü bir uzak bilgisayarda yaptığı için robotun kısıtlı enerjisini daha etkin kullanması sağlanmıştır. Ayrıca işlemler için uzak bilgisayarın kullanılması robot üzerinde yüksek kapasiteli bilgisayar kullanımı gerekliliğini de ortadan kaldırmış, düşük kapasiteli bir bilgisayarla da navigasyon ve haritalama işlemlerinin yapılmasına olanak vermiştir.

Yapılan testlerden elde edilen sonuçlar odometri hesapları yapılırken, Kalman filtrelerinin kullanılmasının faydalarını göstermiştir. Robot EKF ile odometri hesabı yaparken sadece enkoder verilerini değil, aynı zamanda LIDAR ve IMU verilerini de

kullanmıştır. Bu veriler sayesinde haritalama işlemi yapılırken çok daha doğru sonuçlara ulaşılmıştır. Navigasyon işlemlerinde ise robotun işlemlerinde zaman kazancı sağlanmıştır. Kalman filtreleri tüm sensörleri birleştirdiği için çok daha etkin sonuçlara, hızlı şekilde ulaşma imkânı vermiştir. EKF kullanımı hem navigasyon hem de haritalama işlemlerine fayda sağladığı tespit edilmiştir.

KAYNAKLAR

Shen, D., Huang, Y., Wang, Y., & Zhao, C. (2018, August). Research and implementation of SLAM based on LIDAR for four-wheeled mobile robot. In 2018 IEEE International Conference of Intelligent Robotic and Control Engineering (IRCE) (pp. 19-23). IEEE.

Gong, Z., Li, J., & Li, W. (2016, July). A low cost indoor mapping robot based on TinySLAM algorithm. In 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS) (pp. 4549-4552). IEEE.

Chen, Y., Tang, J., Jiang, C., Zhu, L., Lehtomäki, M., Kaartinen, H., ... & Chen, R. (2018). The accuracy comparison of three simultaneous localization and mapping (SLAM)-based indoor mapping technologies. Sensors, 18(10), 3228.

Tomoiagă, T., Predoi, C., & Coşereanu, L. (2016). Indoor mapping using low cost LIDAR based systems. In Applied Mechanics and Materials (Vol. 841, pp. 198-205). Trans Tech Publications Ltd.

Karam, S., Lehtola, V., & Vosselman, G. (2020). Strategies to Integrate IMU and LIDAR SLAM for Indoor Mapping. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 1, 223-230.

Megalingam, R. K., Teja, C. R., Sreekanth, S., & Raj, A. (2018). ROS based autonomous indoor navigation simulation using SLAM algorithm. International Journal of Pure and Applied Mathematics, 118(7), 199-205.

Meuli, G., Soeken, M., Roetteler, M., Wiebe, N., & De Micheli, G. (2018, January). A best-fit mapping algorithm to facilitate ESOP-decomposition in Clifford+ T quantum network synthesis. In 2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC) (pp. 664-669). IEEE.

Xiang, Y., Yang, X. Q., Yang, W. W., & Miao, W. H. (2020, April). Localization and Mapping Algorithm for the Indoor Mobile Robot Based on LIDAR. In IOP Conference Series: Materials Science and Engineering (Vol. 831, No. 1, p. 012021). IOP Publishing.

Alqahtani, E. J., Alshamrani, F. H., Syed, H. F., & Alhaidari, F. A. (2018, April). Survey on Algorithms and Techniques for Indoor Navigation Systems. In 2018 21st Saudi Computer Society National Computer Conference (NCC) (pp. 1-9). IEEE.

Omara, H. I. M. A., & Sahari, K. S. M. (2015, August). Indoor mapping using kinect and ROS. In 2015 International Symposium on Agents, Multi-Agent Systems and Robotics (ISAMSR) (pp. 110-116). IEEE.

Abdelrasoul, Y., Saman, A. B. S. H., & Sebastian, P. (2016, September). A quantitative study of tuning ROS gmapping parameters and their effect on performing indoor 2D SLAM. In 2016 2nd IEEE international symposium on robotics and manufacturing automation (ROMA) (pp. 1-6). IEEE.

Ibragimov, I. Z., & Afanasyev, I. M. (2017, October). Comparison of ROS-based visual SLAM methods in homogeneous indoor environment. In 2017 14th Workshop on Positioning, Navigation and Communications (WPNC) (pp. 1-6). IEEE.

Li, Y., & Shi, C. (2018, November). Localization and navigation for indoor mobile robot based on ROS. In 2018 Chinese Automation Congress (CAC) (pp. 1135-1139). IEEE.

da Silva, B. M., Xavier, R. S., & Gonçalves, L. M. (2019). Mapping and Navigation for Indoor Robots under ROS: An Experimental Analysis.

Aagela, H., Al-Nesf, M., & Holmes, V. (2017, September). An Asus_xtion_probased indoor MAPPING using a Raspberry Pi with Turtlebot robot Turtlebot robot. In 2017 23rd International Conference on Automation and Computing (ICAC) (pp. 1-5). IEEE.

Afanasyev, I., Sagitov, A., & Magid, E. (2015, October). ROS-based SLAM for a Gazebo-simulated mobile robot in image-based 3D model of indoor environment. In International Conference on Advanced Concepts for Intelligent Vision Systems (pp. 273-283). Springer, Cham.

Portugal, D., Araújo, A., & Couceiro, M. S. (2020). A Guide for 3D Mapping with Low-Cost Sensors Using ROS. In Robot Operating System (ROS) (pp. 3-23). Springer, Cham.

Li, Z., Xiong, Y., & Zhou, L. (2017, December). ROS-based indoor autonomous exploration and navigation wheelchair. In 2017 10th International Symposium on Computational Intelligence and Design (ISCID) (Vol. 2, pp. 132-135). IEEE.

Okumuş, F., & Kocamaz, A. F. (2019, September). Cloud based indoor navigation for ros-enabled automated guided vehicles. In 2019 International Artificial Intelligence and Data Processing Symposium (IDAP) (pp. 1-4). IEEE.

Pajaziti, A., & Avdullahu, P. (2014). Slam–map building and navigation via ros. International Journal of Intelligent Systems and Applications in Engineering, 2(4), 71-75.

Voisan, E. I., Paulis, B., Precup, R. E., & Dragan, F. (2015, May). ROS-based robot navigation and human interaction in indoor environment. In 2015 IEEE 10th Jubilee International Symposium on Applied Computational Intelligence and Informatics (pp. 31-36). IEEE.

Le, X. S., Fabresse, L., Bouraqadi, N., & Lozenguez, G. (2018, August). Evaluation of out-of-the-box ros 2d slams for autonomous exploration of unknown indoor environments. In International Conference on Intelligent Robotics and Applications (pp. 283-296). Springer, Cham.

Kashi, M. S., Sriram, T. B., & Mohan, R. (2019, June). An Approach to Labelled Indoor Mapping using SLAM and Object Detection. In 2019 IEEE Region 10 Symposium (TENSYMP) (pp. 321-325). IEEE.

Rojas-Fernández, M., Mújica-Vargas, D., Matuz-Cruz, M., & López-Borreguero, D. (2018, February). Performance comparison of 2D SLAM techniques available in ROS using a differential drive robot. In 2018 International Conference on Electronics, Communications and Computers (CONIELECOMP) (pp. 50-58). IEEE.

Köseoğlu, M., Çelik, O. M., & Pektaş, Ö. (2017, September). Design of an autonomous mobile robot based on ROS. In 2017 International Artificial Intelligence and Data Processing Symposium (IDAP) (pp. 1-5). IEEE.

Yu, Y., Piao, Y., Ni, Y., & Si, T. (2019, May). Research on Accurate Positioning of Indoor Objects Based on ROS and 3D Point Cloud. In Journal of Physics: Conference Series (Vol. 1229, No. 1, p. 012005). IOP Publishing.

Achmad, H., Daud, M. R., Razali, S., & Pebrianti, D. (2016). A ROS-based human-robot interaction for indoor exploration and mapping. International Journal of Advanced and Applied Sciences, 3(4), 88-92.

Lin, Q., Ke, Z., Bi, S., Xu, S., Liang, Y., Hong, F., & Feng, L. (2017, December). Indoor mapping using gmapping on embedded system. In 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO) (pp. 2444-2449). IEEE.

Ghani, M. F. A., Sahari, K. S. M., & Kiong, L. C. (2014, December). Improvement of the 2D SLAM system using Kinect sensor for indoor mapping. In 2014 Joint 7th International Conference on Soft Computing and Intelligent Systems (SCIS) and 15th International Symposium on Advanced Intelligent Systems (ISIS) (pp. 776-781). IEEE.

Saman, A. B. S. H., & Lotfy, A. H. (2016, August). An implementation of SLAM with extended Kalman filter. In 2016 6th International Conference on Intelligent and Advanced Systems (ICIAS) (pp. 1-4). IEEE.

Liu, C., Zhou, C., Cao, W., Li, F., & Jia, P. (2020). A Novel Design and Implementation of Autonomous Robotic Car Based on ROS in Indoor Scenario. Robotics, 9(1), 19.

Aksoy, S. and Ozan, E., 2020. Robots and Their Applications. Int. Res. J. Eng. Technol.

Skalfist, P., Robotik Devrimi (Vol. 2). Cambridge Stanford Books.

Okumuş, F. and Kocamaz, A.F., 2019. Multi-robot Path Planning Using Fractional Order Darwinian Particle Swarm Optimization Algorithm. Proceedings Book, p.48.

Afanasyev, I., Sagitov, A. and Magid, E., 2015, October. ROS-based SLAM for a Gazebo-simulated mobile robot in image-based 3D model of indoor environment. In International Conference on Advanced Concepts for Intelligent Vision Systems (pp. 273-283). Springer, Cham.

Yang, C., Chen, C., He, W., Cui, R. and Li, Z., 2018. Robot learning system based on adaptive neural control and dynamic movement primitives. IEEE transactions on neural networks and learning systems, 30(3), pp.777-787.

Karam, S., Lehtola, V. and Vosselman, G., 2020. Strategies to integrate IMU and LiDAR SLAM for indoor mapping. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 1, pp.223-230.

Zhang, L., Wei, L., Shen, P., Wei, W., Zhu, G. and Song, J., 2018. Semantic SLAM based on object detection and improved octomap. IEEE Access, 6, pp.75545-75559.

Kim, P., Chen, J. and Cho, Y.K., 2018. SLAM-driven robotic mapping and registration of 3D point clouds. Automation in Construction, 89, pp.38-48.

Bader, K., Lussier, B. and Schön, W., 2017. A fault tolerant architecture for data fusion: A real application of Kalman filters for mobile robot localization. Robotics and Autonomous Systems, 88, pp.11-23.

Ligorio, G. and Sabatini, A.M., 2013. Extended Kalman filter-based methods for pose estimation using visual, inertial and magnetic sensors: Comparative analysis and performance evaluation. Sensors, 13(2), pp.1919-1941.

Reina, G., Vargas, A., Nagatani, K. and Yoshida, K., 2007, September. Adaptive kalman filtering for gps-based mobile robot localization. In 2007 IEEE International Workshop on Safety, Security and Rescue Robotics (pp. 1-6). IEEE.

de Koster, R.B., 2018. Automated and robotic warehouses: Developments and research opportunities. Logistics and Transport, 38(2), pp.33-40.

Woźniakowski, T., Zmarzłowski, K. and Nowakowska, M., 2018. Automation and innovations in logistic processes of electronic commerce. Information Systems in Management, 7(1), pp.72-82.

Girija, P., Mareena, J., Fenny, J., Swapna, K. and Kaewkhiaolueang, K., 2021. Amazon Robotic Service (ARS).

ROS Documentation Available at: wiki.ros.org (Accessed: 20 Jan 2021).

Pololu 34:1 Metal Gearmotor 25Dx64L mm LP 6V with 48 CPR Encoder (No End Cap) Available at: www.pololu.com/product/2284/resources (Accessed at: 15 Sep 2020).

Pololu Tamiya 70145 Narrow Tire Set (2 tires) Available at: www.pololu.com/product/63 (Accessed: 17 Sep 2020).

Pololu 25D mm Metal Gearmotor Bracket Pair Available at: www.pololu.com/product/2676 (Accessed at: 17 Sep 2020).

AKINROBOTICS Mobil Robotları 15. Travel Turkey İzmir Turizm Fuarında Available at: www.akinrobotics.com (Accessed at: 26 Nov 2021).

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Eren OKUR

EĞİTİM DURUMU

Lisans Öğrenimi : 2014, Bahçeşehir Üniversitesi, Mühendislik Fakültesi, Elektrik Elektronik Mühendisliği

Yüksek Lisans Öğrenimi : 2018, Karabük Üniversitesi, Sosyal Bilimler Fakültesi, Girişimcilik Bölümü

Bildiği Yabancı Diller : İngilizce, Almanca

Bilimsel Faaliyetleri :

İŞ DENEYİMİ

Stajlar : 2013, Görüntü işleme stajyeri, Aselsan

Projeler : Ada Serisi Robotları, Sigara Paketleme Makinesi, Roboliza.com, ArMobil, Arat, Robot Kol, Forklit Otomasyon Sistemi, Hastane RFID Otomasyon Projesi, UVC Serisi Robotları, Servis Robotları.

Çalıştığı Kurumlar : 2014, Elektrik Elektronik Mühendisi, BİLTERA, 2016, Elektrik Elektronik Mühendisi, AYTAÇ GIDA, 2018, Yazılım Mühendisi, AKINSOFT

Tarih: 19 Ocak 2023