# SLAM Tabanlı Otonom Temizlik Robotu Tasarımı
# SLAM Based Autonomous Cleaning Robot Design

*Mert Çolak*[1], *Halil İbrahim Erbaş*[2], *Haluk Altay*[3], *Volkan Sezer*[4]

[1]Mekatronik Mühendisliği
İstanbul Teknik Üniversitesi, İstanbul
colak.mrt@outlook.com
erbas18@itu.edu.tr
altayh17@itu.edu.tr

[2]Kontrol ve Otomasyon Mühendisliği Bölümü
İstanbul Teknik Üniversitesi, İstanbul
sezerv@itu.edu.tr

## Özetçe

Modern ev aletlerinin gün geçtikçe otonom olması ile evde harcanan zamanın azaltılması ve yapılan işlerin kolaylaştırılması sağlanmıştır. Elektrikli süpürgeler, temizlik işlerini kolaylaştırmalarına rağmen günlük kullanım için fazla gürültülü ve taşıması zor olan büyük bir gövdeye sahiptir ve bunun yanısıra kullanıcı için büyük zaman kaybına yol açmaktadır. Bu bildiride iç mekan temizlik problemini çözmek için çevresinin haritasını çıkaran ve bu haritaya uygun olarak üreteceği navigasyon çıktıları ile vakumlama işlemini gerçekleştirecek otonom bir robot tasarımı sunulmaktadır. Bildiri kapsamında SLAM tabanlı otonom temizlik robotunun tasarım metodu, mekanik tasarımı, elektronik tasarımı, modelleme ve simülasyon ortamı tasarımı, yazılım algoritmaları tasarımı, robotun matematiksel modeli ve kontrolü konularında yapılan çalışmalar anlatılmaktadır.

## Abstract

Thanks to the being autonomous of modern household appliances day by day, the time spent at home is reduced and the works were provided getting easy. Although vacuum cleaners make cleaning jobs easier, they have a large body that is too noisy and difficult to carry for daily use, and it also causes a huge waste of time for the user. In this paper, an autonomous robot design is presented to maps the surroundings to solve the indoor cleaning problem and performs vacuuming with the navigation outputs that it will produce in accordance with this map. The scope of this paper describes the design method, mechanical design, electronical design, modeling and simulation environment design, software algorithms design, mathematical modelling and control design of SLAM based autonomous cleaning robot.

## 1. Giriş

Temizlik robotları elektrikli süpürgelerin aksine mobil olduklarından kısıtlı çalışma sürelerine sahiptirler ve bu sürelerde en verimli biçimde çalışmaları gerekmektedir. Bu noktada tasarımlar kullanılan sensörler ve yazılımların büyük önemi vardır. 2002 yılında iRobot firması Roomba ismiyle ilk temizlik robotunu piyasaya sürmüştür. Bu robot IR ve RF teknolojilerini kullanarak otonom olarak temizlik yapabilmekte ve otomatik şarj olabilmektedir [1]. NEATO Robotics tarafından 2010 yılında üretilen Neato XV-11 isimli temizlik robotunda ise LiDAR sensörü kullanılmakta ve SLAM algoritması ile çalışmaktadır [2]. 2016 yılında Dyson firması tarafından üretilen EYE-360 temizlik robotu 360° panoramik kamera kullanmaktadır ve otomatik şarj olma yeteneğine sahiptir [3].

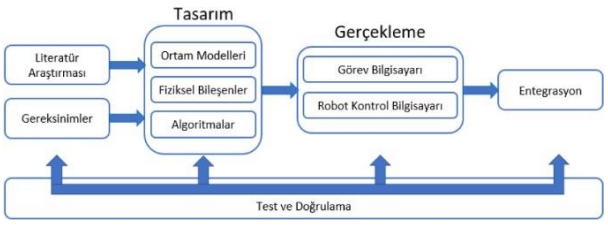
Temizlik robotlarında çeşitli otonomi yaklaşımları bulunmaktadır. iRobot Roomba robotu, iAdapt algoritmasını kullanmaktadır. Herhangi bir harita çizmeyen, odanın neresinde olduğu konusunda bilgisi bulunmayan, engele çarptığında yönünü değiştiren, rastgele hareketler sergileyen bir temizlik robotudur. Bu otonomi yaklaşımı daha az maliyetli fakat zaman alıcı ve enerji tasarrufu konusunda yetersiz bir yaklaşımdır.

Bir diğer yaklaşım ise LiDAR sensörü yardımıyla 360° tarama yaparak odayı haritalayabilen SLAM algoritmasıdır. Eş Zamanlı Konum Belirleme ve Haritalama Algoritmasında(SLAM) amaç adından da anlaşıldığı üzere robotların konum belirlemesini ve ortamın haritalanmasını eş zamanlı olarak yapabilmesidir. Robotun otonomiyi yerine getirebilmesi için üç ana yeteneğe sahip olması gerekir [4][5][6]:

- Bulunduğu noktalara ait pozisyon ve mutlak konum bilgilerinin elde edilmesi (Localization),
- İçinde bulunduğu ortamın haritasının çıkarılması ve bu haritanın sensörlerden alınan ölçümlere göre sürekli güncellenmesi (Mapping),
- Ortam içinde bir noktadan diğerine gidebilmesi için yolun planlanması(Navigation)

## 2. Robotun Tasarım Metodu

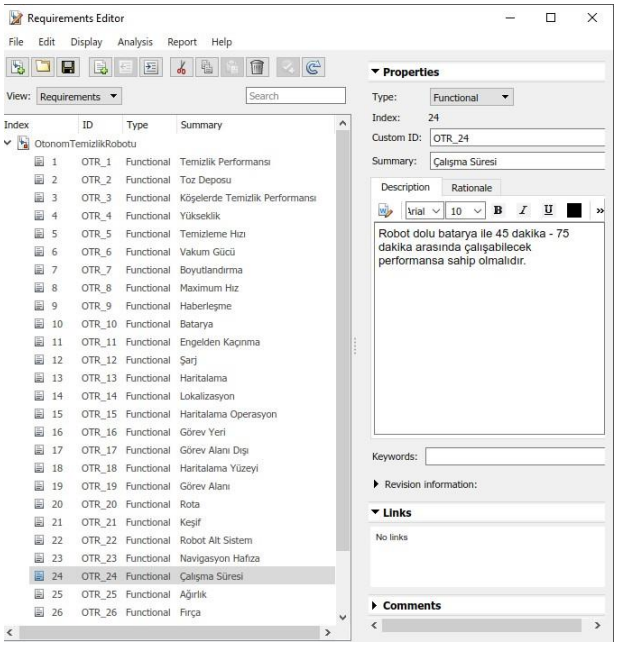
Tasarım çıktılarını elde etmek için en önemli nokta tasarım yöntemini belirlemektir. SLAM tabanlı otonom temizlik robotu sistemi için model tabanlı tasarım yaklaşımı uygulanılmaktadır. Model tabanlı tasarım yaklaşımının diyagram şeklinde ifade edilişi Şekil 1'deki gibidir.




*Şekil 1: Robotun Tasarım Metodu*

### 2.1. Sistem Gereksinimleri

SLAM tabanlı otonom temizlik robotunun davranışının nasıl olması gerektiği, sistem olarak neleri barındırması gerektiği ve tasarım kısıtlamalarının neler olması gerektiğini belirlemek için detaylı bir gereksinim çalışması gerçekleştirilmiştir. Detaylı gereksinim çalışması kapsamında literatür araştırmaları sonucunda benzer robotlar ve akademik çalışmalar göz önünde bulundurularak nihai sistem gereksinimleri türetilmiştir. Nihai sistem gereksinimleri Simulink Requirements Editor aracı kullanılarak Şekil 2'de gösterildiği gibi oluşturulmuştur.




*Şekil 2: Sistem gereksinimleri*

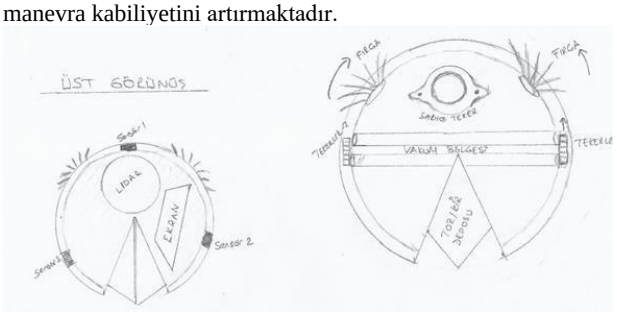
## 3. Robotun Tasarımı

### 3.1. Mekanik Tasarımı

SLAM tabanlı otonom temizlik robotunun mekanik tasarımı üç aşamalı bir süreç ile oluşturulmuştur. İlk aşamada literatürdeki benzer otonom temizlik robotları incelenerek farklı konseptlerde ön tasarımlar gerçekleştirilmiştir. İkinci aşamada ön tasarım konseptlerinden en uygun olanı seçilip sistem gereksinimleri ve elektronik donanım tasarımları dikkate alınarak detaylı bir tasarım yapılmıştır. Üçüncü aşamada ise robotun ürün haline dönüşeceği düşünülerek tasarımın son hali oluşturulmuştur.

#### *3.1.1. Mekanik Ön Tasarımı*

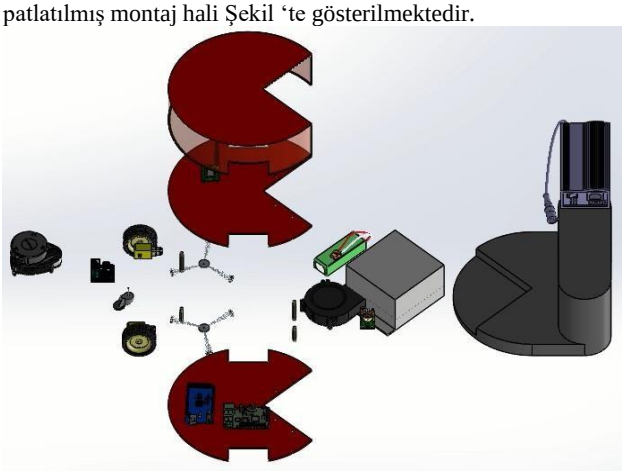
Robotun mekanik tasarımı için dairesel, kübik ve kompakt olmak üzere üç alternatif ön tasarım bulunmaktadır. Bu ön tasarımlar farklı fonksiyonel özellikler ile birbirlerine üstünlük sağlamaktadır.
Ön tasarım alternatiflerinden sistem gereksinimlerine en uygun olan kompakt tasarım, 90° yakın bir tasarım detayı ile duvar kenarlarındaki performansını artırırken dairesel tasarımı ile de manevra kabiliyetini artırmaktadır.




*Şekil 3: Ön Tasarım Alternatifi - Kompakt*

#### *3.1.2. Detaylı Mekanik Tasarım*

SLAM tabanlı otonom temizlik robotu 10 farklı alt sistemden oluşmaktadır. Alt sistemler sırasıyla; robot gövdesi (çöp kutusu dahil), tekerlek, motor ve sürücüsü, fan motoru ve sürücüsü, robot kontrol bilgisayarı, robot görev bilgisayarı, haberleşme, LiDAR sensörü, batarya, şarj istasyonu ve şarj cihazıdır.
Robotun alt sistemleri ile beraber 3B - CAD modelinin patlatılmış montaj hali Şekil 'te gösterilmektedir.



*Şekil 4: Robotun patlatılmış montaj CAD modeli*

*Robot gövdesi alt sistemi*, robotun ana yapısalını oluşturmaktadır. Robotun 2 katlı bir tasarımı bulunmaktadır. Alt kat hareket için gerekli olan alt seviye kontrolün gerçekleştiği olan kattır. Üst kat ise robotun davranışının belirlendiği üst seviye kontrolünün gerçekleştiği kattır. Ek olarak robotun toz/kir deposu gövdeye uygun bir şekilde duvar kenarlarındaki temizlik performansını artırmak için 90° olacak

şekilde tasarlanmıştır. Gövde malzemesi olarak 2 mm kalınlığında akrilik (pleksiglass) kullanılmıştır.

*Tekerlek alt sistemi,* robotun diferansiyel sürüş metodu ile hareketini sağlayacak olan tekerlek sisteminde robotun yan tarafında iki bağımsız tahrikli tekerlek ve ön tarafında birçok yöne hareket edebilen sarhoş tekerlek bulunmaktadır. Bu tekerlek konfigürasyonu birçok iç mekân mobil robotları içinde geçerlidir.

*Motor ve sürücüsü alt sistemi,* robotun hareketinde bataryadan aldığı gerilimi mekanik enerjiye çevirip tekerlere tahrik sağlamaktadır. Kullanılacak olan motorun gücü, robotun sistem gereksinimleri göz önüne alınarak hesaplanmıştır. Bu hesaplar sonucunda uygun motor seçimi yapılmıştır.

Denklem 1'de $F_z$ bir tekerleğin yere uyguladığı kuvvet, $c$ zemin ile tekerlek arasındaki sürtünme katsayısı [7], $m_{robot}$ robotun toplam kütlesi ve $g$ yerçekimi ivmesidir.

$$F_z = c\frac{m_{robot}}{n}g = 0.8x\frac{2.05}{3}x9.81 = 5.36\ N \quad (1)$$

Denklem 1' de hesaplanan 5.36 N, tekerlerin patinaj çekmeden sağlayabileceği en büyük kuvvet değeridir. Bu kuvvet değeri kullanılarak aracın patinaj çekmeden gidebileceği en büyük ivme değeri elde edilmiştir.

Denklem 2'de $\alpha_{max}$ robotun patinaj çekmeden gidebileceği en büyük ivme, $n_{tahrik}$ motor ile tahriklenen teker sayısıdır.

$$\alpha_{max} = \frac{n_{tahrik}F_z}{m_{robot}} = \frac{2x\,5.36}{2.05} = 5.23\frac{m}{s^2} = 171.48\frac{rad}{s^2} \quad (2)$$

Robotun ilerleme hızı 0.55 m/s olarak belirlenmiştir. Tekerlek yarıçapı ise 0.0305 m' dir. Bu iki değer kullanılarak tekerlerin açısal hızı 18.03 rad/s olarak bulunmuştur. Açısal hız değeri $w$ ile robotun ulaşabileceği en büyük açısal ivme değeri $\alpha_{max}$ kullanılarak tekerlerin patinaj çekmeden ilerleyebileceği en düşük zaman değeri $t$ Denklem 3'te gösterilmektedir.

$$t = \frac{w}{\alpha_{max}} = \frac{18.03}{171.48} = 0.105\ s \quad (3)$$

Elde edilen zaman değeri robotun ilerleme hızına çıkması için gereken minimum zaman değeridir. Bu değerden daha düşük değerlerde tekerlek patinaj çekecektir. Motorun kalkış anında sağlaması gereken en büyük tork değeri $T_{kalkış}$ Denklem 4' te hesaplanmıştır.

$$T_{kalkış} = F_z r = 5.36x3.05 = 16.35\ Ncm \quad (4)$$

Robotun ilerleme hızına ulaşması için geçen süre 0.5 sn olarak belirlenmiştir. Bu durumda robotun minumum 1.1 $m/s^2$' lik bir ivmeye ihtiyacı vardır. Böylece her bir motorun sağlaması gereken minimum tork değeri $T_{min}$ Denklem 5' teki gibi hesaplanmıştır.

$$T_{min} = \frac{m_{robot}\alpha_{min}r}{2} = \frac{2.05x1.1x3.05}{2} = 3.44\ Ncm \quad (5)$$

Yapılan bütün hesaplamalar sonucunda motorun kalkış anında en fazla 16.35 Ncm, ivmelenme performansı için de en az 4.5 Ncm tork sağlaması gerektiği hesaplanmıştır. Bu değerler göz önüne alınarak robot için ForceUp 16A050 DC motor modeli seçilmiştir. Motor sürücü olarak Full H bridge mantığına sahip L298N seçilmiştir.

*Fan motoru ve sürücüsü alt sistemi,* robotun temizlik görevini vakum yöntemi ile yapmasını sağlayan sistemdir. Fan motoru seçiminde en önemli parametre debidir. Robot için fan motoru debi hesabı Denklem 6'da yapılmıştır.

Denklem 6' da gösterilen $Q$ debi değerini, $D$ fan motoru yarıçapını, $V$ emiş yapılan havanın hızını göstermektedir

$$Q = \frac{\pi*D^2}{4} * V * 60 = \frac{\pi*0.025^2}{4} * 30 * 60 = 0.882\ \frac{m^3}{dk} = 0.01475\ \frac{m^3}{s} \quad (6)$$

Literatür araştırmaları sonucunda vakum ile otonom temizlik yapan robotlarda debi 0.013 ile 0.047 $m^3$/s aralığında olduğu bulunmuştur. Denklem 6'dan elde edilen sonuç ile literatürdeki debi değeri aralığında kalınmıştır. Hesaplamalar ve literatür araştırması sonucunda Şekil' teki özelliklere sahip BCB1012UH fan motoru tercih edildi.

| Özellik | Açıklama |
|---|---|
| Boyutu | 97.22mm*94.4 mm*25mm |
| Debi | 0.988 $\frac{m^3}{dk}$ |
| Statik Basınç | 1009.6 Pa |
| Fan tipi | Blower - Salyangoz |
| Gürültü | 67.8 dB |
| Hız | 9300 rpm |
| Güç | 38.4 W |

*Şekil 5: Fan motoru ve özellikleri*

*Robot kontrol bilgisayarı alt sistemi*, robotun hareketini ve manevra yapmasını kontrol eden alt seviye algoritmaları içeren sistemdir. Robot kontrol bilgisayarı olarak Arduino UNO R3 kullanılmaktadır.

*Robot görev bilgisayarı alt sistemi*, robotun davranışını belirleyen otonom yazılımların bulunduğu üst seviye algoritmaları içeren sistemdir. Robot görev bilgisayarı olarak Raspberry Pi 4 kullanılmaktadır.

*Haberleşme alt sistemi*, robotun güncel durumunu kullanıcıya aktaran sistemdir. Robot kullanıcı arası haberleşmeyi sağlar. Haberleşme sistemi için Raspberry Pi ile uyumlu SX1262 LoRa HAT kullanılmaktadır.

*LIDAR Sensörü alt sistemi*, robotun güncel durumu üzerine ölçümler yapan sistemdir. Bu alt sistem sayesinde robot görev bilgisayarının nasıl davranması gerektiğine dair bir geri bildirim ortaya çıkmaktadır. LIDAR olarak A1M8 RPLidar kullanılmaktadır.

*Batarya alt sistemi*, robotun enerji ihtiyacının karşılandığı sistemdir. Robotta 11.1 V 3S Lipo Batarya 6000 mAh 35C modeli kullanılmaktadır.

*Şarj istasyonu ve şarj cihazı alt sistemi*, robot bataryasının istenilen seviyede dolmasını sağlayan sistemdir. Robot bataryası yüzde %10 'un altına düştüğü zaman kendini korumaya alarak şarj istasyonuna gelmektedir. Robot şarj istasyonunda Lipo bataryasını şarj cihazı sayesinde istenilen seviyede doldurmaktadır.

SLAM tabanlı otonom temizlik robotu için ağırlık hesaplamaları Tablo ' te anlatılmaktadır. Bu hesaplama robotun toz/kir toplamadan boş ağırlığı üzerinden yapılmıştır.

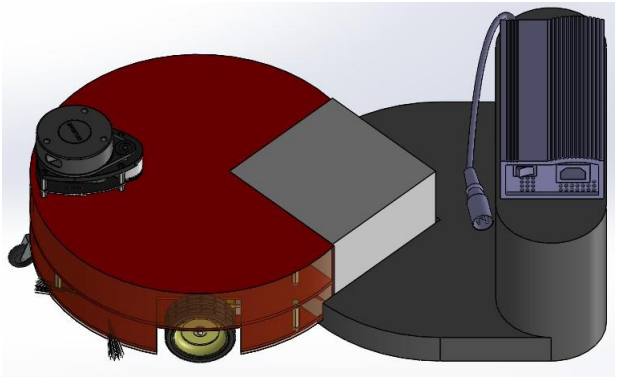
*Tablo 1: Robot bileşenlerinin ağırlıkları*

| Bileşen | Ağırlık |
|---|---|
| Raspberry Pi | 42 gr |
| Tekerlek ve Motorları | 300 gr |
| Vakum Fan Motoru | 190 gr |
| Lipo Batarya | 405 gr |
| LIDAR | 170 gr |
| Robot Gövdesi (Çöp kutusu dahil) - Solidwoks | 650 gr |
| Kablolama | 50 gr |
| Elektronik Malzemeler(Motor Sürücü vb.) | 150 gr |
| Arduino UNO | 25 gr |
| Fırçalar | 60 gr |
| **Toplam** | **2050 gr** |

Bu hesaplama sonucunda robotun boş ağırlığı yaklaşık 2 kg olarak bulundu ve sistem gereksinimlerinde bulunan "OTR_25: Robot maximum 5 kg ağırlığında olmalıdır." gereksinimi karşılanmıştır.

### *3.1.3. Robotun Mekanik Final Tasarımı*

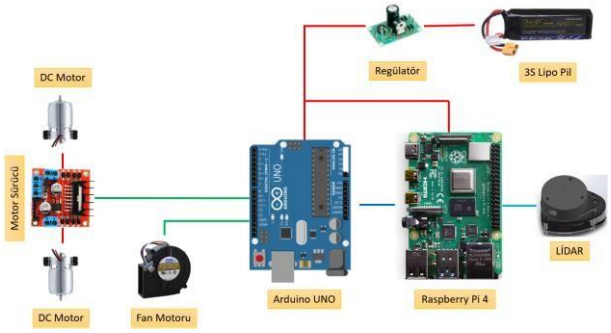
Robotun mekanik tasarımının son aşaması olan üçüncüaşamada ise robotun ürün haline dönüşeceği düşünülerek tasarımın son hali oluşturulmuştur. SLAM tabanlı otonom temizlik robotunun 3D-CAD modeli Şekil 6' da gösterilmiştir.



*Şekil 6: Robotun 3B - CAD modeli*

## 3.2. Elektronik Tasarımı

SLAM Tabanlı Otonom Temizlik Robotu projesinde kullanılacak olan elektronik bileşenlerin genel bağlantı şeması Şekil 7'de gösterilmektedir.




*Şekil 7: Elektronik bağlantı şeması*

SLAM tabanlı otonom temizlik robotunun batarya seçimi ve maximum çalışma süresi için güç tüketim hesabı yapılmıştır. Hesaplanan güç tüketimi Tablo 'te gösterilmektedir.

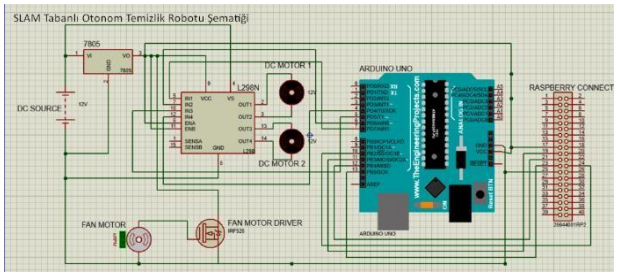
*Tablo 2: Güç tüketim hesabı*

| Bileşen | Güç Tüketimi |
|---|---|
| Elektronik Bileşenler(Arduino UNO, Raspberry Pi 4, LIDAR vb.) | 7 W |
| Tekerlek Motorları | 9.6 W |
| Vakum Fan Motoru | 38.4 W |
| **Toplam** | **55 W** |

Robotun sistem gereksinimlerini karşılaması için seçilen Lipo batarya kapasitesi 6000mAh 'dir. Güç tüketimi toplamına göre robotun maksimum çalışma süresi 73 dakika olarak hesaplanmıştır.

Bu hesaplama sonucunda sistem gereksinimlerinde bulunan "OTR_24: Robot dolu batarya ile 45 dakika- 75 dakika arasında çalışabilecek performansa sahip olmalıdır." gereksinimi karşılanmıştır.

SLAM Tabanlı Otonom Temizlik Robotu elektronik devre şeması Şekil 8'de gösterilmektedir.

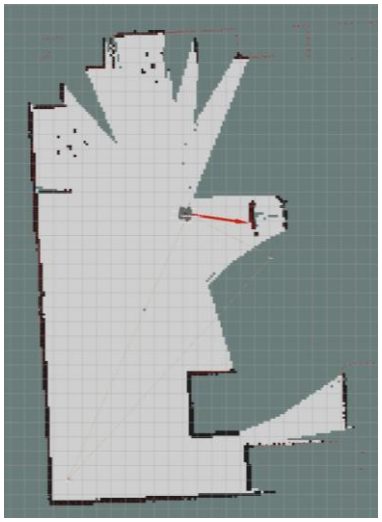



*Şekil 8: Elektronik devre şeması*

## 3.3. Otonom Yazılım Algoritmaları Tasarımı

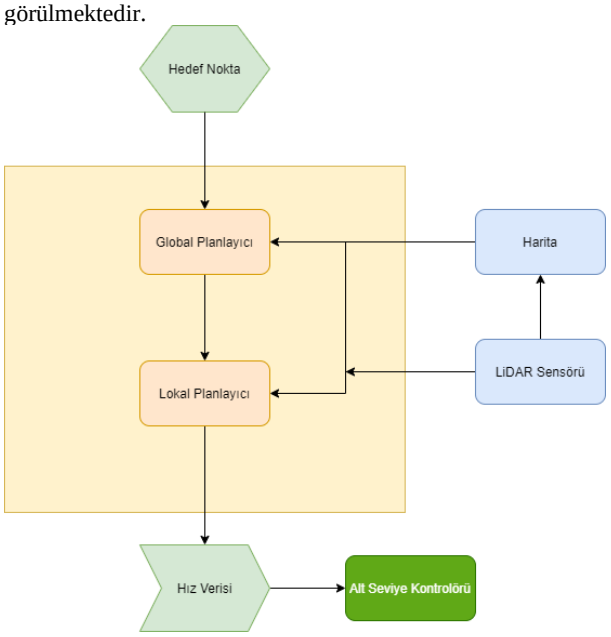
Otonom sürüş özelliğine sahip robotların bulundukları ortam içerisinde istenilen hedefe doğru ilerleyebilmeleri için kendilerini konumlayabilmeleri gerekmektedir. Bunu yapabilmesi amacıyla robotun konumlama algoritmasına sahip olması gerekmektedir. Robotun bulunduğu ortam sürekli değişebilmektedir. Bu sebeple konumlama işlemi sürekli devam etmeli ve robotun bulunduğu ortamı haritalayabilmesi gerekmektedir. Bu çalışmada mobil robot önceden belirlenmiş hareket deseninde rota hesaplar ve hesaplanan rotayı takip ederek temizlik işlemini gerçekleştirir. Robotta kullanılan otonom sürüş algoritmaları şu şekildedir:

*Hector mapping*: Bu algoritma ile robot 2B LIDAR sensöründen aldığı veriler ile eş zamanlı konumlama ve haritalama (EZKH) işlemini gerçekleştirmektedir [8]. Bu sayede sürekli hareket halinde olan robot ortamının haritasını çıkartır ve devamlı güncellenen harita üzerinde kendisini konumlandırır. Şekil 9'da, Gazebo'da hazırlanan test ortamının haritası ve robotun haritada yönelimini belirten ok gösterilmiştir.

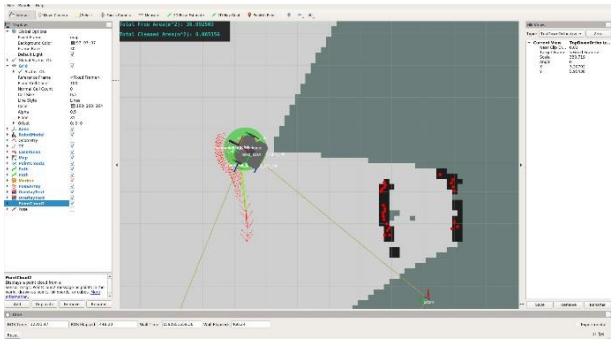


*Şekil 9: Hector mapping ile eş zamanlı konumlama ve haritalama*

*Move_base:* ROS'un navigasyon kütüphanesini oluşturan temel parçadır. Robotun mevcut konumundan hedef noktasına ulaşmasını ROS navigasyon kütüphanesi içerisinde bulunan rota planlama ve takip algoritmalarıyla ya da bu kütüphaneye eklenebilecek farklı algoritmalarla sağlar. Move_base çalışma diyagramı Şekil 10'da görülebilmektedir. Şekil 10'da görüldüğü gibi robotun pozisyon bilgisini, robotun içinde bulunduğu harita bilgisini ve LIDAR sensörü verilerini alarak robotun hedefe ulaşması için gerekli olan lineer ve açısal hız komutlarını üretir. Move_base içerisinde global ve lokal olarak birbirleriyle ortak olarak çalışan iki çeşit rota planlayıcısı bulunmaktadır. Global rota planlayıcısı robotun fiziksel ölçülerini dikkate alarak harita üzerinde seçilen hedefe ulaşmak için yol hesabı yapar. Lokal rota planlayıcı ise robotun engellerden sakınmasını ve global rota planlayıcı tarafından üretilen rotayı takip etmesi için gereken hız komutlarını üretir. Bu çalışmada global rota planlayıcı olarak A* algoritması ve lokal rota planlayıcı olarak TEB Lokal Planlayıcı algoritması kullanılmıştır [9]. Şekil-XX'de hedef noktasına doğru hareket eden robotun lokal planlayıcı çıktıları Rviz ortamında görülmektedir.

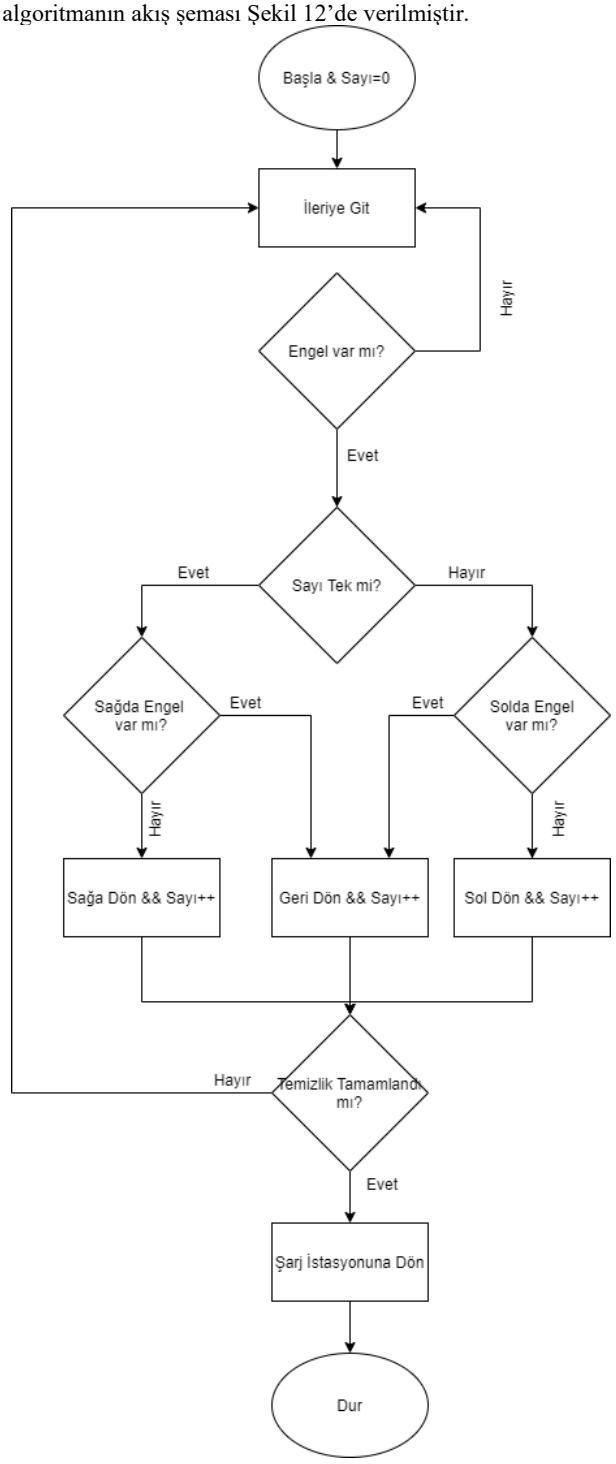



*Şekil 10: Movebase çalışma diyagramı*



*Şekil 11: TEB lokal planlayıcısının çıktısı*

*Hedef Üretme Algoritması:* Robotun otonom navigasyon algoritmasına hedef noktası üretmesi amacıyla geliştirilmiştir. Bu algoritma ile robotun temizlik işlemi sırasında "S" deseninde hareket etmesi sağlanmıştır. Bu algoritmanın akış şeması Şekil 12'de verilmiştir.




*Şekil 12: Hedef üretme algoritması akış şeması*

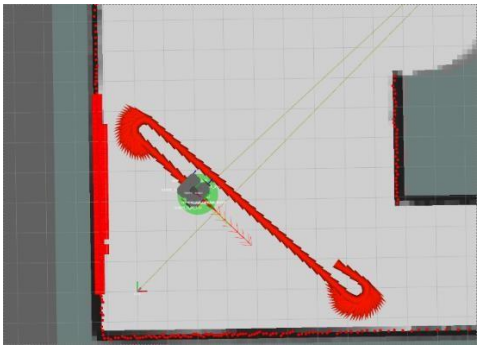
### 3.4. Modelleme ve Simülasyon Ortamı Tasarımı

Otonom sürüş algoritmalarının test edilmesi ve doğrulanması amacıyla açık kaynaklı 3B benzetim yazılımı olan Gazebo kullanılmıştır [10]. Gazebo, otonom sürüş algoritmalarının geliştirildiği ROS ve Linux işletim sistemlerine uyumlu ve yaygın kullanılan

bir benzetim aracıdır [11]. Algoritma çıktılarının ve Gazebo ortam verilerinin görselleştirilmesi amacıyla RViz adı verilen hata ayıklama ve görüntüleme programı kullanılmıştır [12].

Benzetim çalışmaları sırasında sürüş dinamiği ve fiziksel boyut benzerliğinden dolayı Turtlebot3 Waffle Pi robotu Gazebo ortamında kullanılmıştır [13]. Bölüm 3.3'de belirtilen otonom sürüş algoritmaları uygulanarak robotun benzetim çalışmaları gerçekleştirilmiştir. Gazebo'da robotun otonom sürüş algoritmalarının test edileceği ortamın oluşturulması için 3DGEMS 3B model veri seti kullanılmıştır [14]. Şekil 13'te, Gazebo ortamında 3DGEMS 3B model veri seti ile oluşturulan ortam yer almaktadır. Şekil 14'te robotun otonom şekilde belirlenen hedef noktasına giderken RViz'de alınan görüntüsü yer almaktadır.

*Şekil 13: Gazebo test ortamı benzetimi*



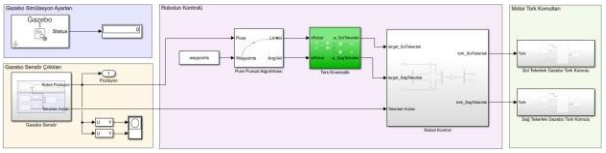
*Şekil 14: RViz'de robotun görselleştirilmesi*

### 3.5. Matematiksel Modellenmesi ve Kontrolcü Tasarımı

#### *3.5.1. Simulink'te Robotun Üst Seviye Kontrolü*

Robotun üst seviye kontrol modeli simülasyonu için Gazebo ve Simulink eş-zamanlı çalışan simülasyon modeli Şekil 15'teki gibi oluşturulmuştur.

Gazebo ortamı, sanal makine üzerinde Ubuntu 18.04 işletim sisteminde; Simulink ortamı ise Windows 10 işletim sisteminde çalışmaktadır. Simulink üzerinden çalıştırılan eş-zamanlı simülasyon modelinde robotun pozisyon, yönelim ve tekerlek hızlarına Gazebo robot modelinden erişilirken takip algoritması Simulink üzerinden çalıştırılmaktadır. Hesaplanan referans komutlar Gazebo ortamına gönderilerek robotun belirlenen rotada takibini sürdürmesi sağlanmaktadır [15][16].



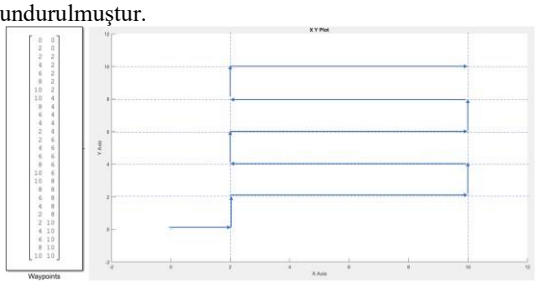
*Şekil 15: Eş zamanlı benzetim modeli*

Bu simülasyon modelinde;

- *Referans Ara Noktalar:* Robotun ulaşmasının arzu edildiği ara noktalar matrisi
- *Kontrol Algoritması:* Ara noktalara ulaşılması için robotun lineer hızını ve açısal hızını kontrol eden Pure Pursuit tabanlı takip algoritması
- *Kinematik Dönüşümler:* Robotun lineer ve açısal hızlarından tekerlek hızlarına dönüşümü için oluşturulmuş olan Ters Kinematik model bloğu
- *Gazebo Sunucusundan Okuma:* Robotun pozisyonu, yönelimi ve tekerlek hızlarının Gazebo sunucusundan okunması
- *Motor Tork Komutları:* Gazebo sunucusunda çalışan robot için motor tork komutları

modelleri kullanılmıştır.

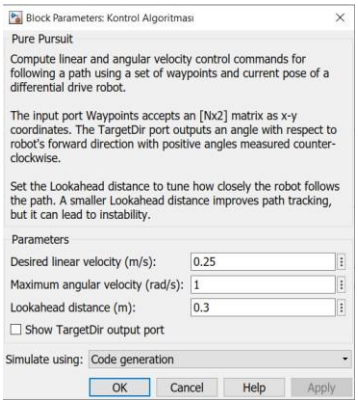
Simülasyon modelinde robotun hedeflenen rotası Şekil 16'da hareket yönleriyle beraber gösterilmiştir. Bu rota üzerinde ilerlenebilmesi için gerekli olan referans ara noktalar bulutu matris şeklinde tanımlanmıştır. Bu tanımlamada SLAM algoritmasının S-tipi hareket deseni göz önünde bulundurulmuştur.



*Şekil 16: Robotun hedeflenen rotası*

Üst seviye kontrol algoritması simülasyon modelinde robotun hedeflenen rotada ara noktalara ulaşması için gerekli lineer hızını ve açısal hızını kontrol eden Pure Pursuit tabanlı kontrolcüden yararlanılmıştır [17]. Robotun o andaki x pozisyonu, y pozisyonu ve phi yönelim açısı bu algoritmanın kullanılması için gereklidir.

Pure Pursuit, MATLAB Robotics System Toolbox bünyesinde bulunan ve belirlenen rotada bir ileri noktaya ulaşmak için robotun açısal hızını sürekli hesaplayan bir rota takip algoritmasıdır [18].

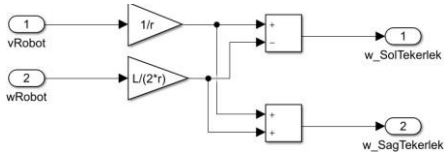



*Şekil 17:Pure pursuit blok parametreleri*

- Robotun lineer hızı 0.25 m/s ve simülasyon sürecinde değişmeyen değer olarak tanımlanmıştır.
- Rota takip edilirken açısal hızının üst limiti 1 rad/s olarak belirlenmiştir.
- Robotun rotasında ilerlerken sürekli baktığı uzaklık 0.3 m olarak belirlenmiştir. Kısa bir uzaklık seçmek robotun agresif dönüşler yapmasına sebep olmakta ve rotayı koruyamamasına yol açmaktadır. Uzun bir uzaklık seçmek ise dönüşlerde geniş bir eğriliğe sebep olmaktadır [17] [19].

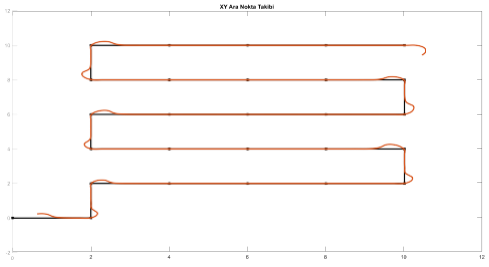
Bu üç parametrenin birbirine uygun şekilde ayarlanması rotanın takip edilmesinde önemli bir yere sahiptir.

Robotun lineer ve açısal hızlarından tekerlek hızlarına dönüşümü için oluşturulmuş olan Ters Kinematik model bloğunun içeriği Şekil 18'de gösterilmiştir. L parametresi aracın tekerlekleri arasındaki uzaklık, r parametresi ise tekerlek yarıçapıdır.




*Şekil 18: Ters kinematik model bloğu*

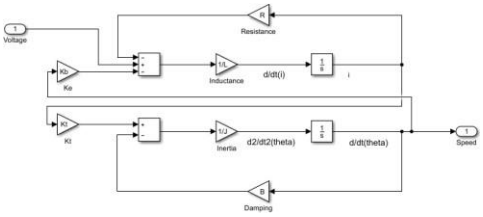
Şekil 16'da tanımlanmış olan hedef rota için simülasyon sonucu Şekil 19'da gösterilmektedir. Simülasyon sonucunda da görüldüğü üzere tanımlanan ara noktalar başarıyla takip edilmiştir.




*Şekil 19: Eş zamanlı benzetim modeli sonucu*

### *3.5.2. Robotun Alt Seviye Kontrolü*

Robotun hareketini sağlayan motorların kontrolünü sağlayan alt seviye algoritmalar Şekil 'de gösterilen "PID Hız Kontrolü" içerisinde tasarlanmıştır. Robotun istenilen noktalara giderken yaptığı hareket için motor kontrolü önemli yer tutmaktadır. Bu amaçla DC motorun modelinin oluşturulması ve buna uygun kontrolcü tasarımının gerçekleştirilmesi gerekmektedir. Tekerlek motorlarının Simulink modeli Şekil 20' de gösterildiği gibi oluşturulmuştur.




*Şekil 20:Dc motor benzetim modeli*

Seçilen motorun parametreleri MATLAB script olarak tanımlandı ve Simulink modeline gönderilmek üzerine wokspace üzerine kaydedildi. Motor parametrelerinin MATLAB scripti aşağıda gösterilmiştir.

```
%DC Motor Parameters
Kt=0.0233;          %motor torque constant
Kb=0.015;           %electromotive force constant
R=20;               %electric resistance
L=2e-03;            %electric inductance
Jload= 3.14e-3; %moment of inertia of the load kgm^2
Jrotor=2e-4; %moment of inertia of the rotor kgm^2
J=Jload+Jrotor;  %total moment of inertia kgm^2
B=5e-04;            %motor viscous friction constant with load
```

Tekerlek motorlarının matematiksel modeli çıkarıldıktan sonra kontrol tasarım aşamasına geçildi. Kontrolcü tasarımı için Tablo 3'te gösterilen tasarım kriterleri oluşturulmuştur.

*Tablo 3: Tasarım kriterleri*

| Tasarım Kriterleri | Değeri |
|---|---|
| Settling time | < 2 saniye |
| Overshoot | < %5 |
| Steady-State Error | < %1 |

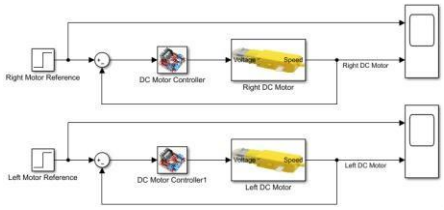
Tablo 3' te belirtilen tasarım kriterlerine göre PID metodu izlenerek kontrolcü tasarımı yapıldı. PID katsayılarını optimize etmek için PID Tuner kullanıldı. Bunun sonucunda P=21, I=41.2 ve D=7 olarak belirlendi. PID kontrolcü için yapılan performans analizi Şekil 'de gösterilmiştir.

Performance and Robustness

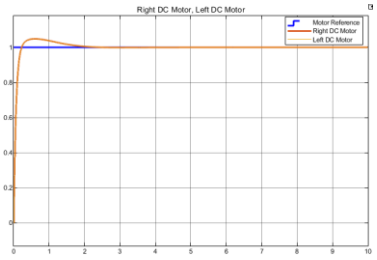
| | Block |
|---|---|
| Rise time | 0.163 seconds |
| Settling time | 1.99 seconds |
| Overshoot | 4.83 % |
| Peak | 1.05 |
| Closed-loop stability | Stable |

*Şekil 21:Kontrölcü performans analizi*

Tekerlek motorlarının matematiksel modeli ve kontrolü için oluşturulan simülasyon modeli Şekil 22'de ve sonuçları Şekil 23'te gösterilmektedir.




*Şekil 22: Motorların benzetim modeli*

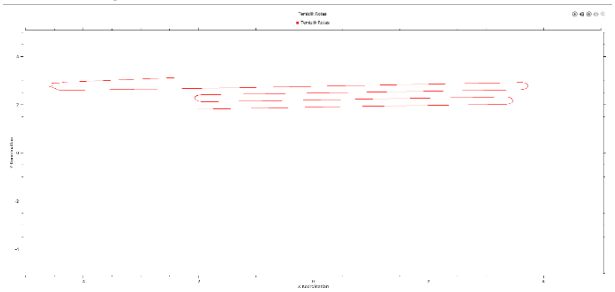



*Şekil 23: Motorların benzetim modeli sonuçları*

## 4. Sonuçlar

Bu çalışmada diferansiyel sürüş mantığına sahip SLAM algoritması tabanlı bir temizlik robotunun; tasarım yönteminin belirlenmesi, mekanik tasarımı, elektronik bileşenlerin seçimi ve devre tasarımı, ROS ortamında SLAM algoritması yazılım tasarımı, Gazebo ortamında robotun benzetimi, Gazebo-Simulink eş-zamanlı benzetim modeli ve robotun alt seviye kontrolü için motor kontrolü konuları işlenmiştir.

Bu çalışmada, Linux'da Robot İşletim Sistemi (ROS-Robot Operating System) yardımıyla, Eş Zamanlı Konum Belirleme ve Haritalama (SLAM) işlemi gerçekleştirilmiştir. Ortam haritasının çıkarılması ve robotun konumunu belirlemesi için lazer ile mesafe ölçümü yapan LIDAR sensörü kullanılmıştır. SLAM için istatistiksel kestirim yöntemlerinden biri olan Gauss-Newton tabanlı Hector SLAM algoritması uygulanmıştır. Böylece ROS kullanılarak özgün tasarım olan robotun otonom kontrolü Şekil 24' te gösterildiği gibi başarılı bir şekilde gerçekleştirilmiştir.



*Şekil 24: Robotun otonom kontrol sonuçları*

## Kaynakça


[1] https://www.irobot.com/
[2] https://www.neatorobotics.com/
[3] https://www.dyson.co.uk/robot-vacuums/dyson-360-heurist-overview.html
[4] Valencia, R., and Andrade-Cetto, J. "Active Pose SLAM" in Mapping, Planning and Exploration with Pose SLAM. Springer, Cham, 2018; 89-108.
[5] Bailey, T. and Durrant-Whyte, H. Simultaneous Localization and Mapping (SLAM): Part i The Essential Algorithms. IEEE Robot Autom. Mag., 2006; 13(2): 99-108.
[6] Leonard, J.J. and Durrant-Whyte, H. F. Simultaneous map building and localization for an autonomous mobile robot. Proc. IEEE/RSJ International Workshop on Intelligent Robots and Systems, 91. Intelligence for Mechanical Systems,IROS'91, 3-5 Nov. 1991; Osaka, Japan,1442-1447.
[7] Oppenheimer,P., Comparing Stopping Capability of Cars with and without Antilock Braking System(ABS),SAE Paper No: 880324,1988
[8] Kohlbrecher, Stefan & Von Stryk, Oskar & Meyer, Johannes & Klingauf, Uwe. (2011). A flexible and scalable SLAM system with full 3D motion estimation. 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR).10.1109/SSRR.2011.6106777.
[9] C. Rösmann, F. Hoffmann and T. Bertram: Integrated online trajectory planning and optimization in distinctive topologies, Robotics and Autonomous Systems, Vol. 88, 2017, pp. 142–153.
[10] Gazebo[Online], Available: http://gazebosim.org
[11] Staranowicz, Aaron, and Gian Luca Mariottini. "A survey and comparison of commercial and open-source robotic simulator software." Proceedings of the 4th International Conference on PervasİVe Technologies Related to Assistive Environments. ACM, 2011
[12] RVIZ[Online], Available: http://wiki.ros.org/rviz
[13] Turtlebot3[Online], Available: http://emanual.robotis.com/docs/en/platform/turtlebot3/specifications/
[14] A.Rasouli, J.K. Tsotsos. "The Effect of Color Space Selection on Detectability and Discriminability of Colored Objects." arXiv preprint arXiv:1702.05421 (2017).
[15] https://www.mathworks.com/help/robotics/examples/perform-co-simulation-between-simulink-and-gazebo.html
[16] https://www.mathworks.com/help/robotics/examples/control-a-differential-drive-robot-in-simulink-and-gazebo.html
[17] Implementation of the Pure Pursuit Path Tracking Algorithm, R.Craig Coulter, Carnegie Mellon University The Robotics Institute
[18] https://www.mathworks.com/products/robotics.html
[19] https://www.mathworks.com/help/robotics/ug/pure-pursuit-controller.html
[20] http://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed&section=SystemModeling