

## İçindekiler

1.	Giriş.....	4
2.	Kurumla İlgili Bilgi: Yapay Zeka ve Robotik Laboratuvarı .....	4
3.	Staj Projesinin Tanımı ve Analizi .....	4
3.1	Kullanılan Teknolojiler.....	4
3.1.1	Pioneer P3-DX Mobil Yer Robotu.....	4
3.1.2	ROS .....	5
3.1.2.1	Tf Paketi.....	5
3.1.2.2	Gmapping Paketi .....	5
3.1.2.3	Navigation Modülü .....	6
3.1.2.3.1	AMCL Paketi.....	6
3.1.2.3.2	Global_planner ve Local_planner .....	6
3.1.2.3.3	Move_base Paketi.....	6
3.2	Proje.....	7
3.2.1	Proje Ortamının Oluşturulması, Yapılandırılması.....	7
3.2.2	Haritalandırma ve Yol Planlama .....	9
3.2.3	Robotun Kütüphane Ortamında Denenmesi .....	11
4.	Staj Deneyimiyle İlgili İzlenimler ve Değerlendirmeler .....	12
5.	Sonuç .....	12
6.	Tavsiyeler .....	12
7.	Referanslar.....	13

## 1. Giriş

Bu rapora konu olan staj, 2015 yaz dönemi, 15.06.2015 – 10.07.2015 tarihleri arasında İstanbul Teknik Üniversitesi Bilgisayar ve Bilişim Fakültesi bünyesindeki Yapay Zeka ve Robotik Laboratuvarı'nda yapılmıştır.

## 2. Kurumla İlgili Bilgi: Yapay Zeka ve Robotik Laboratuvarı

Yapay Zeka ve Robotik Laboratuvarı(AIR Lab) , İTÜ Bilgisayar ve Bilişim Fakültesi koridorunda yer alan, fakülteye bağlı bir araştırma laboratuvarıdır. Laboratuvarı yapay zeka algoritmaları kullanılarak çeşitli robotik projeleri yürütülmektedir. Başlıca araştırma konuları arasında Çoklu Robot Sistemleri, Robot-Sensör Sistemleri, Otonom Robotlar için Öğrenme ve Planlama gibi konular yer alır.

## 3. Staj Projesinin Tanımı ve Analizi

Staj projesi, ROS platformu üzerinde Pioneer P3-DX mobil yer robotu için, robotun bulunduğu ortamın haritasını çıkarması, kendini harita üzerinde konumlandırması ve harita üzerinde istenen hedefe gitmesi üzerinedir.

### 3.1 Kullanılan Teknolojiler

Projede Pioneer P3-DX robotu, robotun kontrolü için Ubuntu Linux 14.04 işletim sistemi üzerinde ROS platformu kullanılmıştır. ROS platformu üzerindeki programlar C++ diliyle yazılmış, ayrıca derleme için CMake gibi araçlar kullanılmıştır.

#### 3.1.1 Pioneer P3-DX Mobil Yer Robotu

Pioneer P3-DX mobil yer robotu, iki tekerlekli diferansiyel sürücüye sahip bir araştırma robotudur. Robotun hareketi üzerinde bulunan bütünleşik mikrokontrolcü yardımıyla sağlanır. Mikrokontrolcüye gerekli komutlar bir bilgisayar tarafından seri port üzerinden aktarılır. Robot, ön ve arka bölümlerinde çarpma sensörüne; ön bölümünde ultrasonik mesafe ölçüm sensörlerine sahiptir.[1]



Resim 3.1: Pioneer P3-DX robotu [2]

Laboratuarda bulunan robot, mevcut donanımlarına ek olarak Asus Xtion Pro stereo kamera, mesafe ölçümü için Hokuyo marka lazer sensör ve uzaktan kontrol amaçlı bir joystick'e sahiptir. Stereo kamera robotun nesne tanımlamayla ilgili başka projelerde kullanılmasını sağlamaktadır.



Resim 3.2: Projede kullanılan lazer sensör [3]

### 3.1.2 ROS

ROS (Robot Operating System) robotik projelerinde kullanılmak için tasarlanan çeşitli kütüphanelere ve araçlara sahip açık kaynak bir platformdur.

ROS projeleri genel olarak *packet* adını alır. ROS projelerinin çalışma prensibi *node* adı verilen program parçalarına ve *topic* adı verilen, bu programların kendi aralarında TCP-IP protokolü yardımıyla haberleşmesine dayanır. ROS projelerinde *node* ve *topic* gibi yapılara ilaveten başlatma dosyaları (*launch file*), çeşitli konfigürasyon dosyaları bulunabilir. Bir ROS projesi oluşturmak için ROS ile birlikte gelen *catkin* sistemi kullanılır.[4]

Projede kullanılan başlıca paketlerle ilgili bilgiler aşağıda verilmiştir:

#### 3.1.2.1 Tf Paketi

Bu paket, çoklu koordinat çerçevelerinin (*coordinate frame*) tutulmasını sağlar ve bu çerçevelerin kendi aralarındaki zamana bağlı ilişkilerini düzenler. Ayrıca bu paket sayesinde koordinat çerçeveleri arasında vektör ve nokta dönüşümleri yapılabilir.[5]

#### 3.1.2.2 Gmapping Paketi

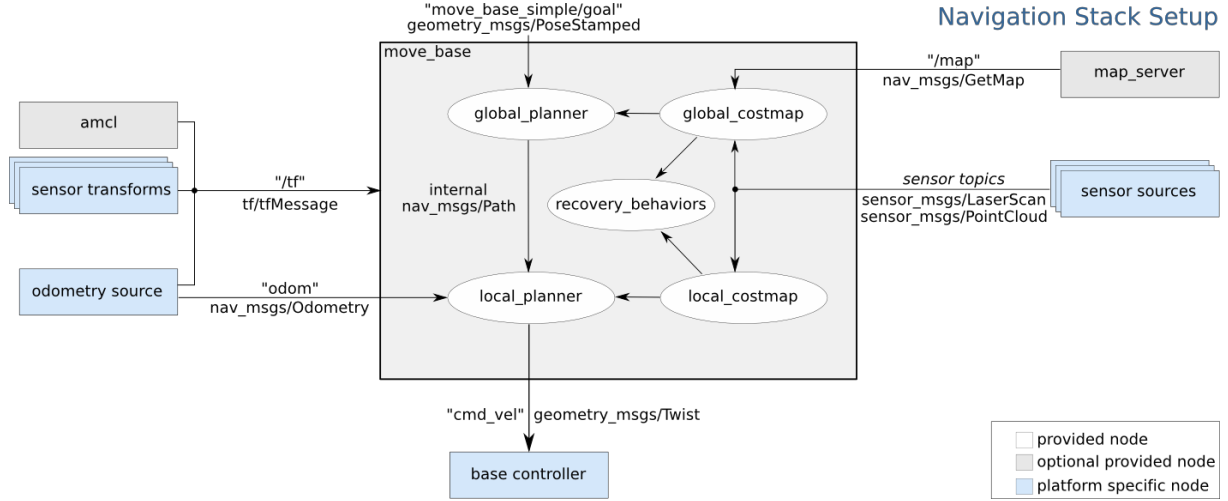
Gmapping paketi, lazer sensörden aldığı veriler yardımıyla robotun bulunduğu ortamın iki boyutlu haritasını çıkarması için kullanılır.



Şekil 3.3: Gmapping ile simulasyon üzerinde oluşturulan örnek bir harita.

### 3.1.2.3 Navigation Modülü

Navigation modülü, robotun konum belirlemesi, hedef tanımlaması, yol planlaması gibi işlerini yapan paketleri bulundurur.



Şekil 3.4: Navigation modülünde kullanılan bazı bileşenler [6]

#### 3.1.2.3.1 AMCL Paketi

AMCL paketi (*Adaptive Monte Carlo Localization*), Monte Carlo Lokalizasyon algoritmasını kullanarak robotun bilinen bir harita üzerinde kendini konumlandırmasını sağlar.

Monte Carlo Lokalizasyonu algoritması, öncelikle robotun bulunabileceği konum olasılıklarını tüm haritaya eşit olarak dağıtır. Daha sonra, lazer sensörden aldığı veriler yardımıyla bu olasılıklara katsayı ağırlıkları verir ve robotu bulunma olasılığı en yüksek bölgede konumlandırır. Robotun hareketi sonrası odometre sayacına göre konumunu değiştirir ve tekrar lazer sensörden aldığı verilerle konumunu düzeltir. [7]

#### 3.1.2.3.2 Global\_planner ve Local\_planner

Navigation modülünde, istenen hedefe ulaşmak için iki türlü planlama sistemi çalışır: *global planner* ve *local planner*. Global planlayıcı, haritadaki sabit engelleri göz önünde bulundurarak hedefe ulaşacak en kısa mesafeli rotayı çizer. Lokal planlayıcı ise robotun yakın çevresindeki, harita üzerinde bulunmayan ancak lazer sensör tarafından algılanan engelleri aşmasını sağlar. [8]

#### 3.1.2.3.3 Move\_base Paketi

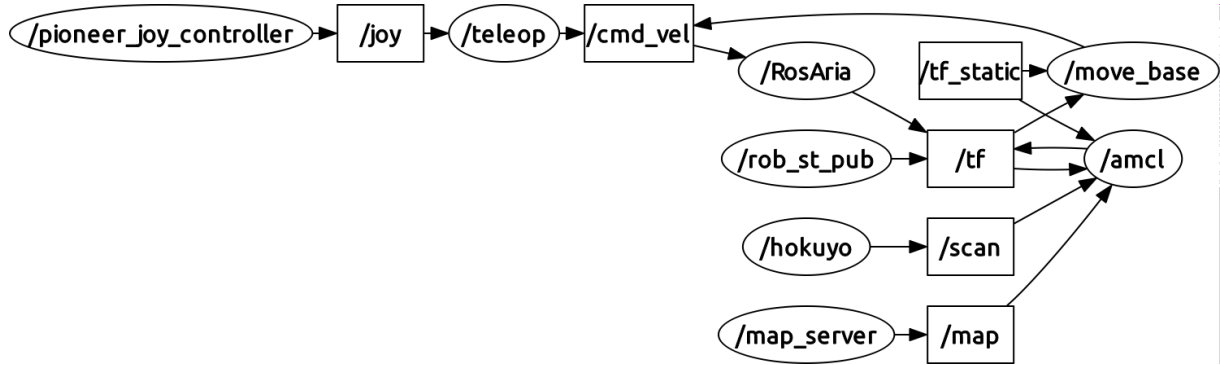
Move\_base paketi, robotun hareketini sağlayan bileşenleri oluşturur. Hedef bilgisini alır, global ve lokal planlayıcıların oluşturduğu planlara göre robota gönderilecek hız verisini oluşturur. Robotun mevcut konumunu ve hedefe ulaşma durumunu bildirir.

Ayrıca, robotun lokal engeller sebebiyle hedefe ulaşacak rota çizememesi durumunda kurtarma hareketleri (*recovery behaviours*) yardımıyla plan oluşturmaya yardımcı olur. Kurtarma hareketlerinde robotun kendi etrafında dönmesi sağlanarak lokal haritasındaki engeller güncellenir. [9]

### 3.2 Proje

Stajda yapılan iş, robotun kütüphanede kullanılmasını amaçlayan projenin bir parçasıdır. Projede, robot öncelikle *gmapping* ile ortamın haritasını çıkarır. Bu esnada robot, manuel olarak hareket ettirilir. Daha sonra bu harita kaydedilerek *navigation* modülü içindeki *move\_base* paketi yardımıyla robotun gitmesi istenen konum belirlenir ve robot, bu konuma gitmek için *local* ve *global* plan oluşturur, oluşturulan plana hız komutları robotun içindeki mikrokontrolcüye aktarılır.

Projenin çalışma prensibini açıklayan, *rqt\_graph* ile oluşturulan bir grafi aşağıdaki gibidir.



Şekil 3.5: Projedeki node ve topic'leri gösteren grafik

Dikdörtgen şekiller *topic*'leri, elips şekiller paketlerin içindeki *node*'ları ifade eder. Yönlü oklar ise aradaki iletişimi belirtir.

Robotun manuel kontrolünü sağlayan *pioneer\_joy\_controller*, joystick'ten gelen mesajları */joy* üzerinden *teleop*'a iletir. *Teleop*, bu bilgileri hız formatına çevirip *RosAria*'ya iletir. *RosAria*, gelen mesajları seri port üzerinden robota göndererek robotun hareketini sağlar.

*Map\_server*, önceden oluşturulmuş olan haritayı tutar. *Hokuyo*, lazer verisini sağlar. *rob\_st\_pub* gerekli çerçeve dönüşümlerini (tf) sağlar. *Amcl* paketi lazer, harita ve dönüşüm bilgilerini alarak robotu konumlandırır. *Move\_base* ise robotun istenen hedefe gitmesi için gerekli hız bilgisini */cmd\_vel* üzerinden *RosAria*'ya gönderir.

#### 3.2.1 Proje Ortamının Oluşturulması, Yapılandırılması

ROS platformunda paketler, çalışma ortamı(*workspace*) bünyesinde oluşturulur. Staj projesinde, çalışma ortamı için *catkin* sistemi kullanılmıştır.

Catkin sisteminde bir çalışma ortamı, belirlenen konum içerisinde, terminalden *catkin\_init\_workspace* komutu ile oluşturulur. Çalışma ortamının içerisinde bir paket oluşturmak için, terminal içerisinden çalışma ortamındaki “src” klasörüne girilir, *catkin\_create\_pkg* <paket adı> <paket gereksinimleri> komutu çalıştırılır. Daha sonra, oluşturulan paket bünyesinde yer alan konfigürasyon dosyaları, paket gereksinimlerini ve paket içerisinde oluşturulacak düğümleri(*node*) içerecek şekilde düzenlenir.

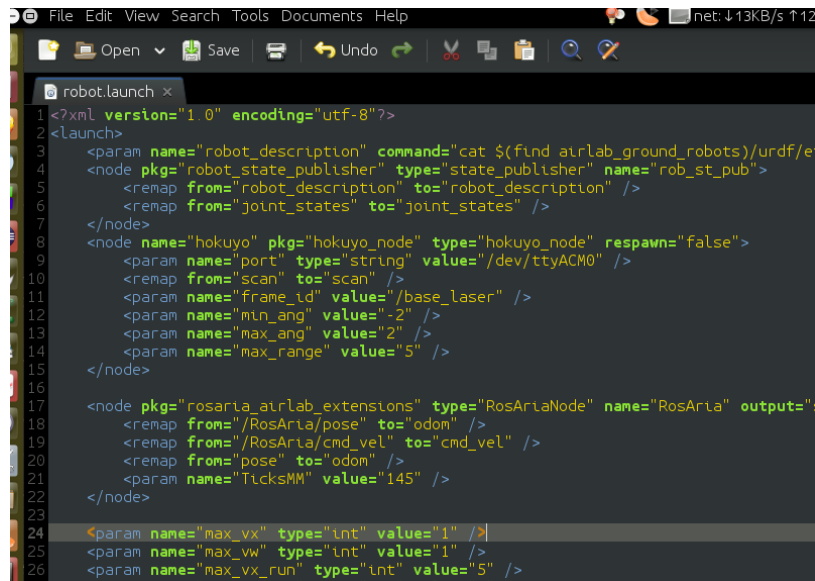
Çalışma ortamındaki paketlerin derlenmesi için, çalışma ortamı dizininde *catkin\_make* komutu kullanılır.

Stajda kullanılan proje, var olan bir projenin düzenlenerek tekrar oluşturulmasıyla meydana getirilmiştir. Staj projesi başlatma dosyalarından, düğümlerden, parametre dosyalarından oluşmaktadır.

launch:	launch:	arg: map_file
efe.launch	amcl.launch	node: map_server
		node: amcl
	launch:	launch: move_base.launch
	robot.launch	node: move_base
		rosparam: costmap
		parametreleri..
		param: robot_description
		node: robot_state_publisher
		node: hokuyo
		node: rosaria
		param: hız parametreleri ...
		node: joy
		node: teleop_airlab

Şekil 3.6: Başlatma dosyaları hiyerarşisi

Projenin çalıştırılması için terminalden *roslaunch airlab\_ground\_robots efe.launch* komutu kullanılır. Bu komut ile projedeki gerekli tüm bileşenler bir başlatma dosyası(efe.launch) altından çalıştırılır. Efe.launch dosyası, amcl.launch ve robot.launch dosyalarını çalıştırır. Bu dosyalar da şekilde verilen parametreleri yükler ve *node*'ları çalıştırır. Başlatma dosyaları xml formatında yazılır.



```

1 <?xml version="1.0" encoding="utf-8"?>
2 <launch>
3   <param name="robot_description" command="cat $(find airlab_ground_robots)/urdf/ef
4   <node pkg="robot_state_publisher" type="state_publisher" name="rob_st_pub">
5     <remap from="robot_description" to="robot_description" />
6     <remap from="joint_states" to="joint_states" />
7   </node>
8   <node name="hokuyo" pkg="hokuyo_node" type="hokuyo_node" respawn="false">
9     <param name="port" type="string" value="/dev/ttyACM0" />
10    <remap from="scan" to="scan" />
11    <param name="frame_id" value="/base_laser" />
12    <param name="min_ang" value="-2" />
13    <param name="max_ang" value="2" />
14    <param name="max_range" value="5" />
15  </node>
16  <node pkg="rosaria_airlab_extensions" type="RosAriaNode" name="RosAria" output="s
17    <remap from="/RosAria/pose" to="odom" />
18    <remap from="/RosAria/cmd_vel" to="cmd_vel" />
19    <remap from="pose" to="odom" />
20    <param name="TicksMM" value="145" />
21  </node>
22
23  <param name="max_vx" type="int" value="1" />
24  <param name="max_vw" type="int" value="1" />
25  <param name="max_vx_run" type="int" value="5" />
26

```

Şekil 3.7: Projede kullanılan robot.launch başlatma dosyası

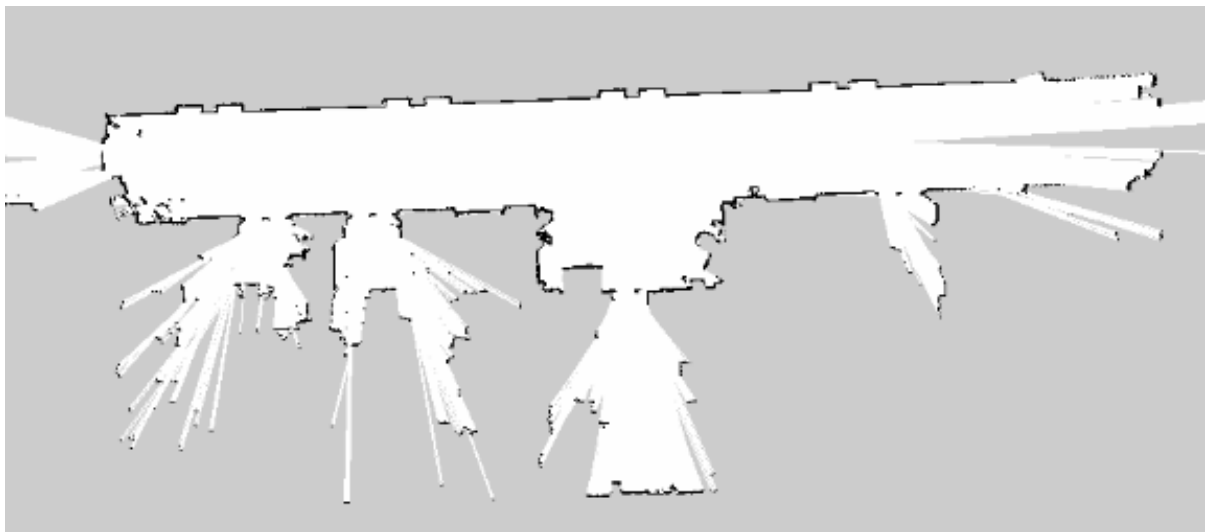
### 3.2.2 Haritalandırma, Konumlandırma ve Yol Planlama

Projede, robotun bulunduğu ortamı haritalandırması için efe.launch dosyasının yanı sıra, proje içinde bulunan gmapping.launch dosyası çalıştırılır. Gmapping.launch dosyası, gerekli parametrelerle gmapping paketi içerisinde aynı isimdeki düğümü çalıştırır.

```
gmapping.launch x
1 <?xml version="1.0" encoding="utf-8" ?>
2 <launch>
3   <node pkg="gmapping" type="slam_gmapping" name="slam_gmapping">
4     <param name="map_update_interval" value="1" />
5     <param name="maxUrange" value="59.0" />
6     <param name="sigma" value="0.05" />
7     <param name="kernelSize" value="1" />
8     <param name="lstep" value="0.05" />
9     <param name="astep" value="0.05" />
10    <param name="iterations" value="5" />
11    <param name="lsigma" value="0.075" />
12    <param name="ogain" value="3.0" />
13    <param name="lskip" value="0" />
14    <param name="srr" value="0.01" />
15    <param name="srt" value="0.02" />
16    <param name="str" value="0.01" />
17    <param name="stt" value="0.02" />
18    <param name="linearUpdate" value="0.5" />
19    <param name="angularUpdate" value="0.436" />
20    <param name="temporalUpdate" value="3.0" />
21    <param name="resampleThreshold" value="0.5" />
22    <param name="particles" value="80" />
23    <param name="xmin" value="-10.0" />
24    <param name="ymin" value="-10.0" />
25    <param name="xmax" value="10.0" />
26    <param name="ymax" value="10.0" />
27    <param name="delta" value="0.05" />
28    <param name="llsamplerange" value="0.01" />
29    <param name="llsamplestep" value="0.01" />
30    <param name="lasamplerange" value="0.005" />
31    <param name="lasamplestep" value="0.005" />
32  </node>
33 </launch>
```

Şekil 3.8: Gmapping.launch dosyası

Haritalandırma işlemi sırasında robot manuel olarak uzaktan kontrol edilir. Gmapping paketi, lazer verisinden aldığı uzaklık değerlerini baz alarak bulunduğu ortamın haritasını oluşturur. Aşağıdaki resimde bölüm koridorunun bir haritası yer almaktadır.

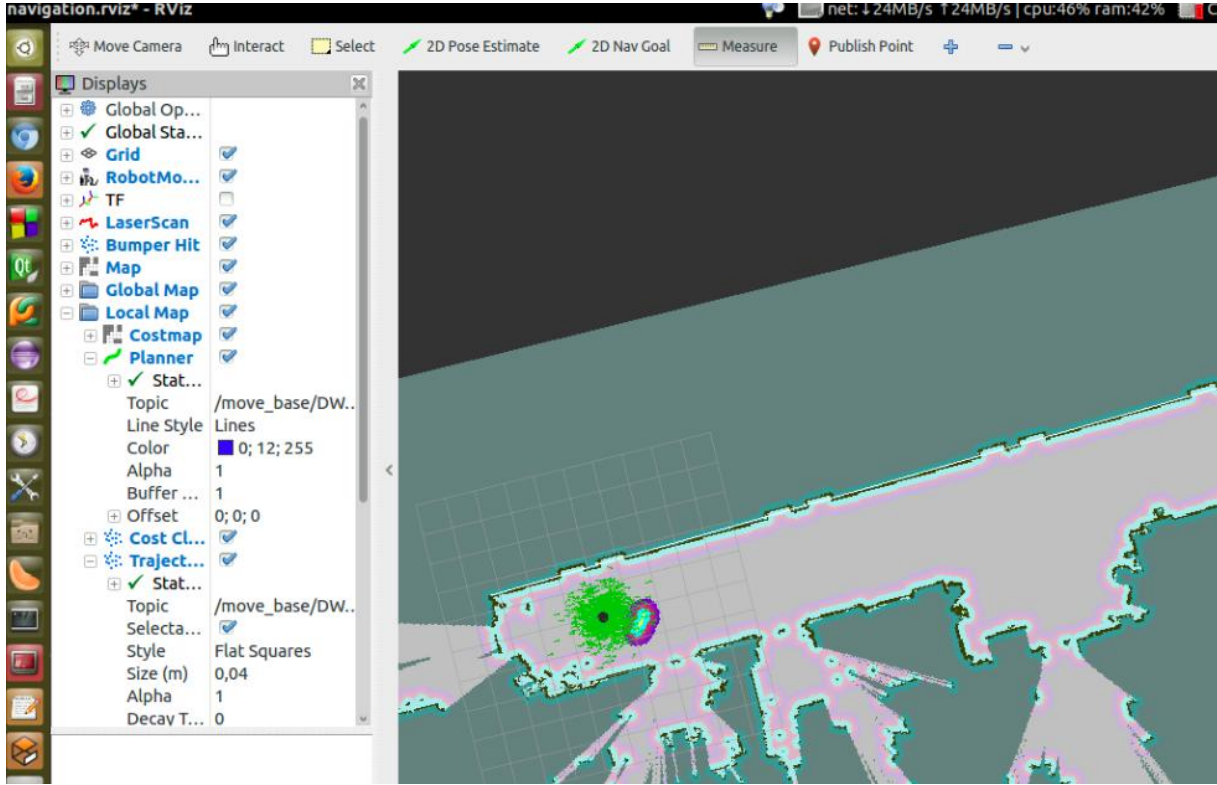


Şekil 3.9: Bölüm koridorunun gmapping paketi ile oluşturulan haritası. Siyah kenarlar duvarları, beyaz bölgeler boşlukları, gri alanlar taranmayan yerleri gösteriyor.



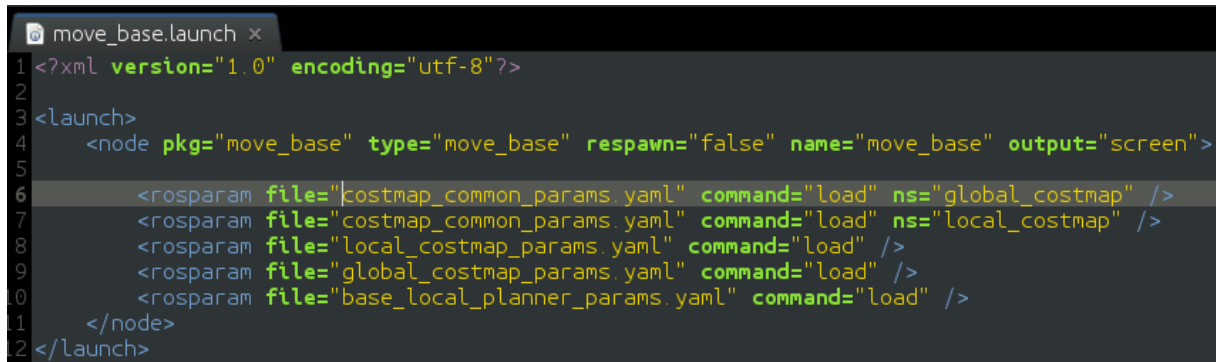
Robotun bilinen harita üzerinde kendini konumlandırması *amcl* paketi sayesinde gerçekleşir. Aşağıdaki resimde *amcl* paketinin oluşturduğu, robotun bulunduğu konum olasılıkları dağılımı, *RViz* görselleştirme aracı altında yeşil oklarla gösteriliyor.

Robota hedef konum bilgisi *Rviz* içinde *2D Nav Goal* butonu ile verilir. Ayrıca robotun mevcut konumunda yanlışlık tespit edilirse, *2D Pose Estimate* butonu ile robotun konumu düzeltilebilir.



Şekil 3.10: Konumlandırmanın bölüm koridorunda denenmesi. Resimdeki yeşil alanlar *amcl* paketinin oluşturduğu veriyi gösteriyor.

Robotun yol planlaması ile ilgili parametreleri *move\_base.launch* dosyası altından çağrılır. Bu parametreler arasında maliyet haritalarının oluşturulması, kurtarma hareketleri, planlayıcı çalışma frekansı gibi özelliklerle alakalı veriler yer alır.



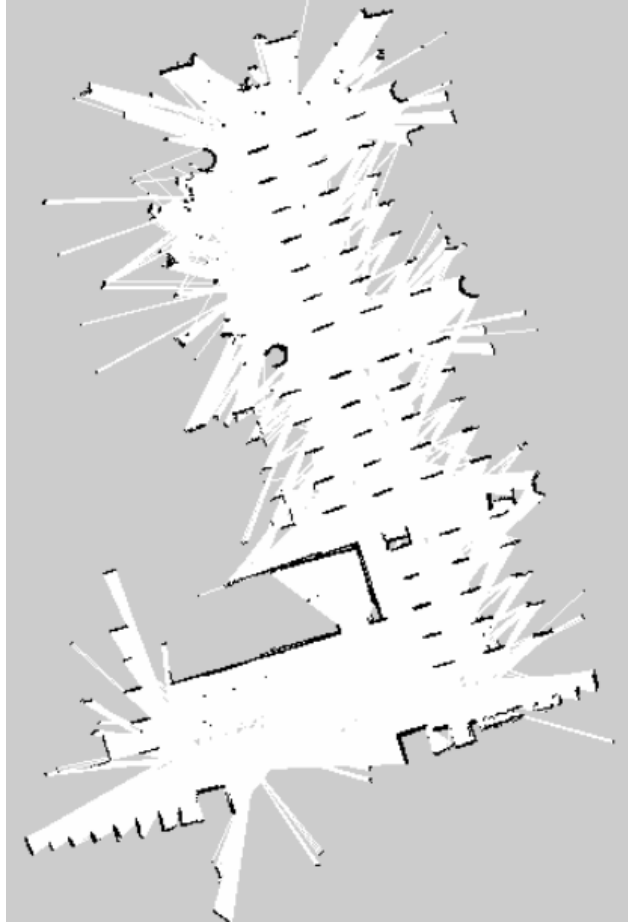
Şekil 3.11: *move\_base.launch* dosyası



### 3.2.3 Robotun Kütüphane Ortamında Denenmesi

Stajın son günlerinde, robotun kütüphane ortamında çalışması denendi. Kütüphanede ilk katın bir bölümünün haritası çıkarıldı. Robota verilen navigasyon hedeflerinde, robotun dar alanlarda performansı gözlemlendi.

Çıkarılan haritada, robotun üzerindeki lazer sensörün yerden yüksekliğinin yetersiz olması dolayısıyla, kitapların bulunduğu rafların, genellikle boş olan en alt kısmını algılayabildiği tespit edildi. Bu durumun, robotun raf sıraları arasından geçerken, performansını düşürdüğü gözlemlendi. İlaveten, haritalama esnasında ortamdaki masa ve sandalye bacaklarının lazer ışığı yansıtması sebebiyle harita üzerinde doğru şekilde yer alamadığını tespit edildi.



*Şekil 3.12: Kütüphane ortamında çıkarılan bir harita*

Ayrıca kütüphane ortamında hareketli pek çok geçici engelin(insan) bulunmasının, robotun hareket kabiliyetini ve haritalandırma, konumlandırma, yol planlama özelliklerini kısıtlayabileceği gözlemlendi.



*Resim 3.13: Kütüphane ortamında yapılan çalışmalardan bir fotoğraf*

#### **4. Staj Deneyimiyle İlgili İzlenimler ve Değerlendirmeler**

Yapay Zeka ve Robotik Laboratuar'ında lisans, yüksek lisans ve doktora öğrencileri bulunuyor. Laboratuardaki çalışmalar gönüllülük esasına dayanıyor. Bu sebepten dolayı, laboratuarda çalışacak olan kişilerin, robotik alanına ilgili olmaları ve kendi kendilerini motive edebilmeleri gerekiyor.

#### **5. Sonuç**

Stajda, bir robotik projesinin genel hatlarıyla nasıl ele alındığı, hangi tür donanım ve yazılıma ihtiyaç duyduğu gözlemlendi.

Stajın ilk haftası ROS platformu ile ilgili temel alıştırmalar yapıldı. İkinci ve üçüncü hafta projede kullanılan paketler üzerinde çalışıldı, ilaveten proje dosyaları eski proje kodlarının düzenlenmesiyle tekrar oluşturuldu. Son hafta projeye ilgili pratik çalışmalar yapıldı.

#### **6. Tavsiyeler**

ROS platformu ile ilgili alıştırmalara erken başlanması, staj projesi sürecinin daha hızlı ilerlemesine yardımcı olabilir. Staj sürecinde yapılan işlerin düzenli olarak dökümantasyonu, staj raporu için oldukça faydalı olacaktır.

Ayrıca staj rapor yazımı sürecinin, rapor teslim tarihine yaklaşıldıkça zorlaşması sebebiyle, staj yapılırken gerçekleştirilmeye başlanmasını tavsiye ederim.

## 7. Referanslar

- [1] <http://www.mobilerobots.com/ResearchRobots/PioneerP3DX.aspx>
- [2] <http://www.smashingrobotics.com/telepresence-robots-reviewed-part-2/>
- [3] [https://www.hokuyo-aut.jp/02sensor/07scanner/urg\\_04lx\\_ug01.html](https://www.hokuyo-aut.jp/02sensor/07scanner/urg_04lx_ug01.html)
- [4] <http://wiki.ros.org/ROS>
- [5] <http://wiki.ros.org/tf>
- [6] [http://wiki.ros.org/nav\\_core](http://wiki.ros.org/nav_core)
- [7] <http://wiki.ros.org/amcl>
- [8] [http://wiki.ros.org/base\\_local\\_planner](http://wiki.ros.org/base_local_planner)
- [9] [http://wiki.ros.org/move\\_base](http://wiki.ros.org/move_base)