OTONOM ROBOTLAR ARACILIĞIYLA VERİ MERKEZLERİNDEKİ KABİNET İÇİ SICAKLIK DAĞILIMININ BULANIK MANTIK İLE KONTROLÜ

Mehmet Bayram BAŞCI

Yüksek Lisans Tezi
Bilgisayar Mühendisliği Anabilim Dalı
Dr. Öğr. Üyesi Bariş ÖZYER
2019

ATATÜRK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

OTONOM ROBOTLAR ARACILIĞIYLA VERİ MERKEZLERİNDEKİ KABİNET İÇİ SICAKLIK DAĞILIMININ BULANIK MANTIK İLE KONTROLÜ

Mehmet Bayram BAŞCI

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

ERZURUM
2019

T.C.
ATATÜRK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

TEZ ONAY FORMU

OTONOM ROBOTLAR ARACILIĞIYLA VERİ MERKEZLERİNDEKİ KABİNET İÇİ SICAKLIK DAĞILIMININ BULANIK MANTIK İLE KONTROLÜ

Dr. Öğr. Üyesi Barış ÖZYER danışmanlığında, Mehmet Bayram BAŞCI tarafından hazırlanan bu çalışma 01/08/2019 tarihinde aşağıdaki jüri tarafından Bilgisayar Mühendisliği Anabilim Dalı'nda Yüksek Lisans Tezi olarak **oybirliği/~~oy çokluğu~~ (.../...)** ile kabul edilmiştir.

Başkan: Prof. Dr. Abdulsamet HAŞILOĞLU *İmza:*

Üye: Dr. Öğr. Üyesi Barış ÖZYER *İmza:*

Üye: Dr. Öğr. Üyesi Saeid AGAHIAN *İmza:*

Yukarıdaki sonuç;

Enstitü Yönetim Kurulunun 23./ 08./ 2019 tarih ve 23 / 68 nolu kararı ile onaylanmıştır.

Prof. Dr. Mehmet KARAKAN Y.
Enstitü Müdürü

Not: Bu tezde kullanılan özgün ve başka kaynaklardan yapılan bildirişlerin, çizelge, şekil ve fotoğrafların kaynak olarak kullanımı, 5846 sayılı Fikir ve Sanat Eserleri Kanunundaki hükümlere tabidir.

ÖZET

Yüksek Lisans Tezi

OTONOM ROBOTLAR ARACILIĞIYLA VERİ MERKEZLERİNDEKİ KABİNET İÇİ SICAKLIK DAĞILIMININ BULANIK MANTIK İLE KONTROLÜ

Mehmet Bayram BAŞCI

Atatürk Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Barış ÖZYER

Günümüzde teknolojinin ilerlemesi ile bilgisayar verileri ve ağları bu duruma paralel olarak çok hızlı bir biçimde büyümektedir. Dolayısıyla veri ve bu hizmetlerin kullanıcılara hızlı bir biçimde sağlanması çok büyük önem arz etmektedir. Bu sebeple günümüzde veri merkezlerinin varlığı ve önemi her geçen gün artmaktadır. Veri merkezlerinde kullanılan büyük enerjinin kontrol atında tutulması ve en ideal şartlarda kullanılması gerekmektedir. Veri merkezlerinde iklimlendirme sistemlerinin var olması tek başına yeterli değildir. Bu sistemlerin veri merkezinde kullanılan elemanların performanslı çalışması için doğru konumlandırılması ve sistem odalarındaki hava akımını kontrol edebileceğimiz yardımcı ürünler ile desteklenmesi gerekmektedir. Literatür incelendiğinde, veri merkezlerinde sabit olarak konumlandırılmış sıcaklık ve nem sensörleri vasıtasıyla yapılan ölçümlerin güvenilirliğinde dikkat çeken problemler tespit edildiği gözlemlenmiştir.Daha kaliteli bir sıcaklık ölçümü yapmak amacıyla veri merkezinde, ortamın haritasını çıkaran ve otonom olarak hareket ettiği esnada belirli rota üzerinde sıcaklık ölçümü yapan gezgin robot kullanılmıştır. Böylece sıcaklık ölçümü yapılırken sabit konumlandırılmış sensörler yerine robotik bir bakış açısıyla belirli bir yerden sabit bir ölçüm yapmak yerine gezgin robot ile otonom olarak farklı noktalardaki ortam sıcaklık ölçümü yapılması tasarlanmıştır. Kabinet içi sıcaklıkları her kabinet içinde konumlandırılmış aynı tip sensörlerle elde edilmiştir. Bu ölçümler sonuçu yapay zeka yöntemlerinden biri olan bulanık mantık kullanılarak bir bulanık sistem tasarımı oluşturulmuştur. Giriş parametreleri, bulanık kurallar ve çıkış parametresi yapılan çalışmalar ve uzman görüşleri alarak belirlenmiştir. Kabinet içi ve ortam sıcaklık ölçümleri bulanıklaştırılarak zemin ızgaralarının açısal değeri hesaplanarak uygun şartlarda bulunmayan kabinet içi sıcaklıklarının uygun hale getirilmesi amaçlanmıştır.

2019, 111 sayfa

Anahtar Kelimeler: Veri merkezi, mobil robot, rota planlaması, bulanık mantık, arduino

ABSTRACT

MS Thesis

FUZZY LOGIC CONTROL OF IN-CABINET TEMPERATURE DISTRIBUTION IN DATA CENTERS THROUGH AUTONOMOUS ROBOTS

Mehmet Bayram BAŞCI

Atatürk University
Graduate School of Natural and Applied Sciences
Department of Computer Engineering

Supervisor: Asst. Prof. Dr. Barış ÖZYER

Nowadays, in accordance with the development of technology, computer data and network have been rapidly growing. Therefore, it is very important that data and these services are provided to users quickly. For this reason, the existence and importance of data centers are increasing every day. Large energy used in data centers should be kept in the control horse and used in the most ideal conditions. The existence of air conditioning systems in data centers alone is not enough. These systems need to be positioned correctly for the performance operation of the elements used in the data center and supported by auxiliary products that we can control the airflow in the system rooms. When the literature was examined, it was observed that remarkable problems were detected in the reliability of measurements made by means of temperature and humidity sensors fixed in data centers. In order to make a higher quality temperature measurement, the data center used a wandering robot that maps the environment and measures temperature on a specific route while acting autonomously. Thus, when measuring temperature, it is designed to measure ambient temperature at different points autonomously with the wandering robot instead of making a fixed measurement from a particular location from a robotic point of view, rather than with fixed positioned sensors. In-cabinet temperatures were measured with the same type of sensors positioned within each cabinet. A fuzzy system design was created using fuzzy logic, which is one of the methods of artificial intelligence. Input parameters, fuzzy rules and output parameters were determined by taking Studies and expert opinions. The aim of this study is to calculate the angular value of the floor grilles by blurring the internal temperature measurement and ambient temperature measurements of the cabinet and to configure the internal temperature of the cabinet which is not available under the appropriate conditions.

2019, 111 pages

Keywords: Data center, autonom roboat, path planning, fuzzy logic, arduino

TEŞEKKÜR

Yüksek lisans tezi olarak hazırladığım bu çalışmamda, çalışmamın her aşamasında bilimsel katkılarını ve manevi desteğini esirgemeyen kıymetli danışmanım Sayın Dr. Öğr. Üyesi Barış ÖZYER'e teşekkürlerimi sunarım.

Tez sürecinde tecrübeleriyle çalışmama katkı sağlayan Sayın hocam Dr. Öğr. Üyesi Bilal USANMAZ'a teşekkürü borç bilirim.

Yardımlarını esirgemeyen Atatürk Üniversitesi Bilgisayar Bilimleri Araştırma ve Uygulama Merkezi'nde beraber çalıştığım değerli çalışma arkadaşlarım doktora öğrencisi Ömer Çağrı YAVUZ'a, yüksek lisans öğrencisi ve çalışma arkadaşım Berat ÇAĞLAR'a, Öğr. Gör. Özgür Fırat ÖZPOLAT'a teşekkürlerimi sunarım.

Manevi desteklerini benden esirgemeyen ablalarım Dr. Hümeyra BAŞCI ve tecrübeleriyle bana yol gösteren Dr. Öğr. Üyesi Zeynep BAŞCI NAMLI ile kıymetli aileme büyük katkılarından dolayı şükranlarımı sunarım.

Mehmet Bayram BAŞCI
Temmuz 2019

İÇİNDEKİLER

SİMGELER ve KISALTMALAR DİZİNİ

º	Derece
C	Celsius
µ	Mu

Kısaltmalar

AMCL	Adaptive Monte Carlo Localization
BİT	Bilgi ve İletişim Teknolojisi
BTU	British Thermal Unit – İngiliz Isı birimi
CPU	Central Processing Unit
CRAC	Computer Room Air Conditioner
DWA	Dynamic Window Approach
FPS	Frame Per Second
GPU	Graphical Processing Unit
GUI	Graphical User Interface
ICSP	In-Circuit Serial Programming
IDE	Integrated Development Environment
RFID	Radio Frequency Identification
ROS	Robot Operating System
SLAM	Simultaneous Localization and Mapping
SSD	Solid State Disk
TF	Transform

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

1. GİRİŞ

Günümüzde tüm alanlarda yapılan çalışmalar bilgisayar disiplini ile ortak çalışmaktadır. Bu çalışmalar sonucunda elde edilen verilerin önemi her geçen gün artmaktadır. Veriler ile bu verilerin hızlı bir biçimde işlenmesi içinde bulunduğumuz şartlar sebebi ile çok önemlidir. Verilerin daha hızlı, daha güvenilir, daha doğru ve uygun maliyet ile sunulması ve yedeklerinin depolanması çok önemlidir. Bu iş akış sürecini göz önüne aldığımızda veri merkezlerinin günümüzde bilgi işlem sektöründe ne kadar önemli olduğunu ortaya çıkmaktadır.

Sahip olduğumuz enerji kaynaklarının etkili kullanılması günümüzde her alanda olduğu gibi veri merkezleri içinde büyük önem taşımaktadır. Veri merkezlerinde iklimlendirme sistemlerinin kullanılıyor olması zaruri bir hal almıştır. Ayrıca veri merkezleri içinde belirli noktalarda konumlandırılan ısı ve nem sensörleri içeride bulunan hava koşullarının sabit olması için sistem yönetcilerine kontrol imkanı sunmaktadır. Aynı zamanda çok ciddi yatırımlar yapılarak veri merkezlerinde konumlandırılan sunucu, depolama üniteleri, network cihazları ile güvenlik cihazları performanslı çalışmaları verilen hizmetlerin kaliteli olması sebebiyle çok önemlidir. Bu cihazların performanslı çalışmaları için iklimlendirme sisteminin iyi tasarlanmış olması lazım. Bu cihazlardan gelen uygun sıcaklıktaki havanın veri merkezi elemanlarının tamamına ulaştırmak, işleyen sistemin tam performanslı çalışması için gereklidir. Her 15℃ sıcaklık artması ile cihazların devre gecikmesi zamanı %10-15 oranında artmaktadır (Wan *et al.* 2018). Bu sebepten kabinet içi sıcaklıklığı ortam sıcaklığına yakın olması büyük önem taşımaktadır.

Sabit bir noktaya konumlandırılmış sensör veri merkezleri için ilk problemdir (Mansley *et al.* 2011). Düşük bütçeli, ortam haritasını çıkaran, otonom hareket eden ve gerçek zamanlı kullanılan bir robot ile mekânsal olarak daha yoğun bir sıcaklık algılaması sağlayabilir (Mansley *et al.* 2011). Bu doğrultuda bu çalışmada hareketli robot kullanılarak veri merkezi içinde bulunan ortam sıcaklığının elde edilmesi amaçlanmıştır.

Böylece sabit bir noktaya konumlandırılmış ortam sıcaklığını ölçen sensör yerine ölçümün daha homojen bir biçimde elde edilmesi amaçlanmıştır.

Robotik teknolojilerinin ilerlemesi ile birlikte robotlar, kendi kendine hareket edebilmekte, engelleri algılamakta, otomasyon sistemlerinde kullanılmakta, bulunduğu ortamların haritalarını çıkarmaktadır. Bu çalışmada otonom hareket eden gezgin robotumuz ile robotun hareket rotası boyunca belirli aralıklarla ortam sıcaklığı elde edilmiştir. Her kabinet içinde ayrı ayrı konumlandırılmış sıcaklık ve nem sensörlerinden gelen sıcaklık verileri ile ortam sıcaklığı verileri bulanık mantık tekniği ile bulanıklaştırılmış ve oluşturulan kurallar ile birlikte anlamsal bir ilişki oluşturulmuştur. Veri merkezinde yükseltilmiş zemininde üzerinde ve kabinetlerin önünde konumlandırılan ızgaraların hangi açı ile açılıması gerektiği belirlenerek kabinet içi sıcaklık değerinin uygun şartlarda tutulması amaçlanmıştır. Böylece kabinet içinde bulunan sunucu, depolama birimi, network cihazları ve diğer cihazların uygun performansta çalışması sağlanabilir.

2. KAYNAK ÖZETLERİ

2.1. Veri Merkezleri

Günümüzde gelişen internet, veri depolama ve yüksek iş gücüne sahip bilgisayarlara olan ihtiyacın artması veri merkezlerinin önemini ortaya koymasıyla beraber eski dönemlere dayanmaktadır. 1960'lı yıllarda üretilen bilgisayarların fiziksel boyutları çok büyük olduğundan, cihazlar kendileri için özel odalarda tutuluyorlardı. Bu bağlamda bilgisayarlar sadece kamu ve askeri çalışmaların yapıldığı alanlarda kullanılmaktaydı. Bu cihazların çalışması esnasında ortamda sıcaklık değerlerinin artması soğutma sistemlerine olan ihtiyacı ortaya çıkarmıştır. Ek olarak ihtiyaçların artması ve teknolojilerin gelişmesiyle, cihaz sayısının artması ve tüm bu cihazların bir düzen içinde çalışma gereksinimi ortaya çıkmıştır. Bilgisayarlar 1980'li yıllarda fiziksel boyutlarının küçülmesiyle her yerde kullanılabilen cihazlara dönüştüler. 1990'lı yıllarda sunucu – istemci mimarisi standart bir hale geldi. Böylece sistem odaları bütün bu sunucu – istemci mimarisine sahip cihazları kendi bünyesinde toplamaya başladı. İnternetin teknolojisinin hızla ilerlemesi ile 7x24 kesintisiz hizmet veren sistem yapıları kurulmaya başlandı. Tüm bu alanda gelişen teknolojilerin sonucunda; daha hızlı internet altyapısı, yüksek iş yüküne cevap verebilen sunucular, büyük verilerin saklandığı depolama alanlarına olan ihtiyaç ile birlikte daha büyük, düzenli ve güvenli veri merkezlerine talep her geçen gün artmaktadır (Moral 2016) .

Artan Cihaz kapasitesi ile birlikte yüksek enerji tüketimi, sistem odalarında ciddi anlamda ekonomik ve çevresel problemleri beraberinde getirmektedir. Günümüzde veri merkezleri için her detay önem arz etmektedir. Benzer şekilde enerji verimliliğini arttırmak için kullanılan iklimlendirme klimaları, cihazların konumları, altyapı ve protokoller önem arz etmektedir (Alagöz and Çavdar 2013).

Her nesne için ihtiyaç duyulan uygun sıcaklık değerleri sistem odaları için büyük önem taşımaktadır. Sunucu – istemci mimarisinde çalışan; sunucular, depolama üniteleri,

anahtar ürünleri hatta daha giriş seviyede veriyi ve elektrik iletimini sağlayan altyapı malzemeleri ve kablolar belirli bir sıcaklık değerleri arasında istenilen çalışma performansını yakalayabilir. Bu cihazlar için en ideal veri merkezi ortam sıcaklığı 15-32℃ arasında olmalıdır (Patterson 2008). Sistem odasındaki sıcaklık değeri çok yüksek olduğunda cihazlar üzerinde bulunan havalandırma üniteleri doğru çalışmayacak ve cihazların iç bileşenleri yüksek sıcaklıktan dolayı arızalara sebebiyet verebilecektir. En iyi ihtimal ile cihazlar kendilerini güç kesintisi ile korumaya alırlar fakat bu durum sürekli çalışması gereken sistemlerde hizmet kesintisine sebep olabilmektedir. Düşük sıcaklıkta çalışma durumunda sistemler çalıştırıldığında şok etkisi altında kalarak elektronik devrelerde çatlamalara sebep olabilir.

Sistem odalarında sıcaklık ve nem ölçüm yönetimi ile ilgili yapılması gereken işlemler özetlenerek aşağıda verilmiştir.

- Sistem odasında bulunan cihazların optimal çalışma sıcaklıkları tespit edilir
- Sıcaklık ve nem ölçüm cihazlarının sistem odasında belirli noktalarda konumlandırarak ölçüm verilerini elde ederiz
- Sistem odasına alarm cihazları yerleştirilerek, ideal çalışma değerinin üstünde ve altında olacak değerlerde müdahale uyarı sistemi oluşturulmalıdır.

2.2. İklimlendirme Sistemleri

İklimlendirme ve tarihçesini inceleyecek olursak, temel olarak mevcut bir ortamın içinde bulunan sıcaklık, nem, hava hareketi ve içeride bulunan havanın kalitesini uygun şartlarda tutulmasını sağlamaktır (Akın ve Ketenci 2010).

İklimlendirmeye ait kullanılan 2 adet ısı birimi mevcuttur, bunlar: BTU (İngiliz ısı birimi) ve TON'dur. Su sıcaklığını bir fahrenayt değiştirmek için ihtiyaç duyulan ısı miktarına bir libre (0,454 kg) denir. Büyük iklimlendirme yapılarının kapasite ölçümünü belirlemek için kullanılan sıcaklık birimine ise TON denilmektedir. 2 TON iklimlendirme için kullanılan soğurma miktarı ise 24000 BTU'ya eş değerdir (Akın ve Ketenci 2010).

Sistem odası içinde bulunan sunucu depolama ve diğer ünitelerin uygun şartlar altında güvenli ve performanslı bir biçimde çalışması için sistem odası iklimlendirmeye ihtiyaç duyulmaktadır. Bu süreçte sistem odası içi sıcaklığı ve nem değeri, kaliteli hava ve oda içi hava hareketlerini uygun şartlarda tutulması istenilmektedir. Küçük ölçekli sistemlerde temel seviye klimalar kullanılabilir fakat sistem odaları gibi yüksek oranda enerji tüketimine sahip olan, oda içinde çok yoğun bir hava akımı mevcut olan ve kullanılan fiziksel cihazların yoğunluğundan dolayı hassas ayar yapabilen klima sistemlerine ihtiyaç vardır. Sistem odaları için seçilen iklimlendirme cihazlarının soğutma kapasitesi tespit edilirken seçilirken sistem odası içinde bulanan cihaz yoğunluğu ve bu cihazların güç tüketimleri belirlenerek hesaplama yapılır (Akın ve Ketenci 2010).

The American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE) tarafından belirlenmiş sistem odası sıcaklık ve nem verileri aşağıdaki Çizelge 2.1'de belirtilmiştir.

Çizelge 2.1. Önerilen sıcaklık ve nem oranları

	2004 Yayını	2008 yayını
Sıcaklık Alt sınırı	20℃	18℃
Sıcaklık Üst Sınırı	25℃	27℃
Nem alt sınırı	%40 bağlı nem	5,5 Çiy noktası
Nem üst sınırı	55% bağıl nem	60% bağıl nem ve 15 ℃ çiy noktası

2.3. Robot ve Yol Planlama

Robot insan ve insanın etkileşim içinde olduğu çevreyi taklit eden makinelerdır. (Özel 2018). Robotların kullanım amaçları insanlar için tehlikeli olan alanlarda kullanılmış olsa da günümüzde sağlıktan tarıma, üretim sistemlerinden eğlenceye, ulaşım sektöründen uzay çalışmalarına kadar 40'tan fazla alanında kullanılmaktadır (Yazıcı 2016).

Robotik teknolojilerinin ilerlemesi ile birlikte robotlar, kendi kendine hareket edebilmekte, engelleri algılamakta, otomasyon sistemlerinde kullanılmakta, bulunduğu

ortamların haritalarını çıkarmaktadır. Otonom robotlar çeşitli donanım birimleri ve yazılımlar ile bilinmeyen bir ortamı tanıyarak, bu ortamda hareket edebilir hale gelmiştir. Bu alanda yapılan çalışmalar günümüzde yaygınlaşmış ve farklı alanlarda kullanılmaya başlanmıştır (Lafcı 2016). Hareketli robotlar için en büyük problem bulundukları ortamı tanımada karşılaşılan problemlerdir. Bu problemler çözüldükten sonra istenilen noktaya nasıl gideceği ve ortamda bulunan engelleri aşabilme yetenekleridir.

Mobil robotların kullanımının artması ile ortaya çıkan yol planlama problemi üzerine çok çalışılan bir araştırma konusu olmuştur. Yol planlama probleminde asıl amaç robotun başlangıç pozisyonundan istenilen noktaya ulaşmasıdır. Bu süreç esnasında hedef noktaya varılması gereken yol üzerinde bulunan engel olabilecek nesnelerin algılanması ve bu engellere çarpmadan yeniden rotasını belirliyor olmasıdır (Suvaydan 2011). Robotun bulunduğu ortamda sabit nesneler bulunabileceği gibi hareketli ve dinamik olarak konumu değişen nesneler olabilir. Gezgin robotlar istenilen varış noktasına gitmek için çeşitli kesin olmayan farklı yollar kullanacaktır (Lafcı 2016).

Harita oluşturma algoritmaları; ortamın haritasını oluşturmaya başlarken, harita üzerinde belirlemiş olduğu farklı olasılık noktalarını dağıtacaktır. Hareket planlama algoritmaları; ortamda bulunan engellere çarpmadan, çeşitli yol planları oluşturarak istenilen noktaya otonom bir biçimde gitmesi amaçlanmaktadır (Lafcı 2016).

Mobil robotlarda yol planlama probleminde esas alınan temel noktalar şunlardır: Başlangıç, hedef noktası ve sabit nesnelerin belirlendiği 2 boyutlu çalışma haritası. Problemi çözerken asıl amaç belirlenmiş iki nokta arasında engellere çarpmaksızın ideal en uygun (genel olarak en kısa) yolu bulmaktır. Yol planlama probleminin çözümü çok maliyetli olmasına rağmen bu alanda yapılan birçok çalışma mevcuttur. Geleneksel yaklaşımların bazı sorular için esnek cevaplarının olmayışı ki bu sorular:

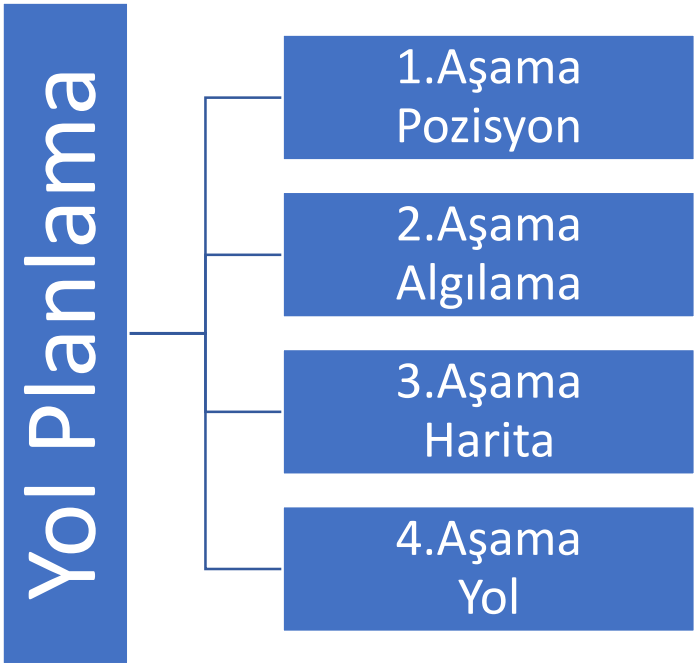
- Farklı optimal hedefleri ve değişen hedefler
- Ortam içindeki belirsizlikler
- Hesaplamalı kaynaklar üzerinde farklı kısıtlamalar

Bu sorulara daha net cevaplar bulmak adına yapılan çalışmalar, araştırmacıları ideal yol bulma, değişen mesafe metotlarına ve değişken programlama metotlarına sevk etmiştir (Nagıp and Gharieb 2004).

Yol planlama problemi çözülürken bazı bileşenlerin biliniyor olması gerekmektedir. Birbirine bağlı bu bileşenler süreci takip edilerek robot yol planlama problemini çözebilir (Suvaydan 2011).

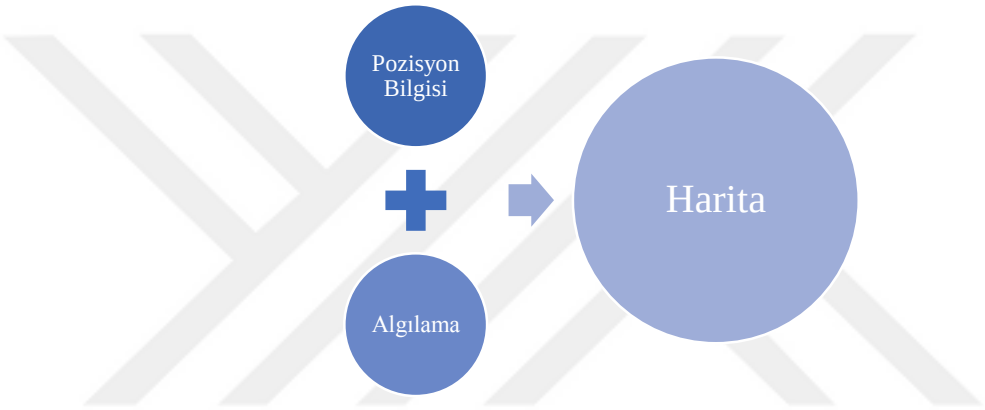
Bu süreçte akla gelen ilk soru: Robot yolu bulmak için nelere ihtiyaç duyar?

- Pozisyon: Ölçümleme / tahmin yaparak robotun pozisyonunun bilinmesi
- Algılama: Nesnelerin ve ortamı sınırlayan engellerin belirlenmesi
- Harita: Gidilecek yolu ve engel bilgisine sahip olma
- Yol: Başlangıç noktasından hedef noktaya kadar takip edilen yollar için hesaplanan yollardır.

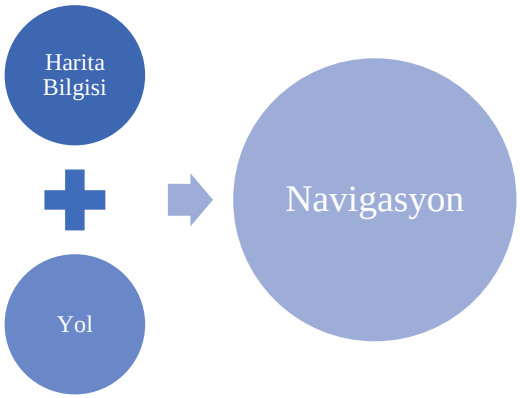


Şekil 2.1. Yol planlama süreci

Şekil 2.1'de görüldüğü gibi süreci incelendiğinde birbiri ile ilişkili bu bileşenlerin bir bütün halinde sağlanıyor olması yol planlama probleminin çözümü için önem arz etmektedir. Yol planlama süreci kendi içinde iki basamakta incelenmektedir. Şekil 2.2'de gösterildiği gibi, pozisyon ve algılama ile haritanın elde edildiği SLAM (Eşzamanlı lokalizasyon ve haritalama) çalışması. İkinci kısım ise Şekil 2.3'de gösterildiği gibi harita bilgisinin yol ile ilişkilendirilmesi sonucunda yol probleminin çözümü olan navigasyondur.



Şekil 2.1. SLAM (Simultaneous localization and mapping) süreci



Şekil 2.2. Navigasyon süreci

Çalışmamızda sistem odası sıcaklık verilerini toplamak için arduino kartı ve hassas sıcaklık ve nem ölçer sensörler kullandık. Gezgin robot üzerinde sıcaklık ölçümü yapacak bir sensör olmaması sebebiyle rabot üzerine entegre edilmiş ardunio kartı ve sensörü

eklenmiştir. Çalışmanın bir bölümünde nesnelerin internetine ve gömülü sistemler üzerine geçiş yapmış olması planlanmıştır.

2.4. Nesnelerin İnterneti

Son yıllarda nesnelerin interneti kavramı hayatımıza çok hızlı bir biçimde girmiştir. IoT uygulamaları gelişen teknolojiler ve insanların bazı ihtiyaçlarının bu teknoloji ile kontrol edilebilir olması sonucunda, Iot cihazları her geçen gün artmaktadır (Ceyhan 2019). Yıkıcı Teknolojilerin Değişen Yüzü Araştırması'nda (The Changig Landscape of Disruptive Technologies) önümüzdeki üç yılda şirketlerin dijital bir dönüşüm içinde olacağı ve yıkıcı teknolojiler gündeme gelmiştir. Araştırmanın sonuçlarına göre robotik, nesnelerin interneti (IoT) ve yapay zekâ çalışmaları ön plana çıkmıştır.

IoT nesneleri arasındaki iletişimde veriler düzenli alınmış, analiz edilmiş ve eylem başlatmak için kullanılmıştır. Bu süreci sağlamak adına planlama, yönetim ve karar verme süreçleri devreye girmiştir. İşte bu yapı IoT yani nesnelerin interneti olarak adlandırılır (Ali 2018).

Nesnelerin interneti kavramının kesin bir tanımı olmamasına rağmen en genel anlamda fiziksel nesnelerin ağ yapısıdır. Her gün eriştiğimiz cihazlara bağlanma yoludur. Bu kavram ilk olarak 1999 yılında Kelvin Ashton tarafından ortaya atılmıştır.

Bilgi ve iletişim teknolojisi (BİT) nedir diye soracak olursak; bilginin elde edilmesi, saklanması, işlenmesi ve çeşitli ağlar yardımı ile bir yerden başka bir yere nakil edilmesidir (Digitatek 2018). IoT sistemi bilgi ve iletişim teknolojisi standartları tarafından çalıştırılabilir. Radyo frekansı ve bluetooth teknolojisi ile çalışan sensörler çevreden bilgi toplarlar. Bir diğer grup ise hazırlamış uygulamalar ile birlikte uzak sunuculara veri iletimi yapmaktadır. Bilgi ve iletişim teknolojileri aynı zamanda elde edilen verileri analiz ederek geri bildirim sağlamak için kullanılır.

Bu çalışmada, hareketli gezgin bir robotun sistem odasında otonom gezebiliyor olması ve belirlediğimiz noktalarda sıcaklık ölçümü yapabilmesi bu alanda uygulanan sabit konumlandırılmış ölçümlere farklı bir alternatif olabileceği düşünülmüştür.

Otonom hareket eden robot ile ortam sıcaklığı tespit edilmiş ve her kabinet içinde bulunan sensör verilerinden gelen sıcaklık değerleri yapay zeka yöntemlerinden biri olan bulanık mantık (fuzzy logic) tekniği ile bulanıklaştırılarak anlamsal bir ilişki oluşturulmuştur. Bunun sonucunda yükseltilmiş zeminde buluna ızgaraların, ilişkilendirilen sıcaklıklara göre kaç derece açılması gerektiği bulanık mantık ile çözülmesi amaçlanmıştır.

2.5. Bulanık Mantık

Bulanık mantık ilk olarak, 1965 yılında California Berkeley Üniveristesinden L.A.Zadeh'in makalelerinde bu konu hakkında bahsetmesiyle ortaya çıkmıştır. İnsanoğlunun var olduğu günden bu güne kadar karşılaştığı olaylar ve bunların çözümü hep karmaşık olmuştur. Bu karmaşıklık beraberinde belirsizlik getirmiştir. Bunun temel sebebi ise kesin bir düşünceye sahip olmanın yanı sıra karar verememekten ileri gelmiştir. Günlük hayatta, bilgisayar dünyasının kullanmış olduğu Aristo mantığından farklı bir biçimde yaklaşık ve kesinlik belirtmeyen veri ve bilgi ile işlem yapabilme eğilimi vardır (Şen 2004). Bulanık mantık, kesin olmayan ifadeleri içerdiği için insanların düşünce yapısına daha uygundur ve mantıksal olarak örtüşmektedir. Bulanık mantık kesin olarak ifade edilmiş sıcak-soğuk, yavaş-hızlı, alçak-yüksek gibi değişkenleri az sıcak-az soğuk, az yavaş-az hızlı, az alçak-az yüksek gibi kesinlik ifadesi içermeyen daha esnek bir yapı haline getirir (Ertuğrul 1996). Çizelge 2.2'de gözüktüğü gibi bulanık mantık kesin olmayan değerlerin anlatımı ve bu belirsizliklerle çalışmak için ortaya çıkmıştır. Olasılık ve istatistik kavramlarında genellikle kesin değerler üzerine çalışılır ama gerçek yaşamda daha çok belirsizlikler hakimdir. Bu sebepten dolayı, insanoğlu çıkarım yapmak ve sonuca ulaşmak adına belirlizlik yapısını kullanır.

Çizelge 2.2. Klasik mantık ve bulanık mantık arası farklılıklar

Klasik Mantık	Bulanık Mantık
Kesin	Kısmi
Hepsi veya Hiçbiri	Belirli Derecelerde
0 veya 1	0 ve 1 Arasında Süreklilik
İkili Birimler	Bulanık Birimler

Bulanık mantık kavramı, tam olarak makinelere, insanlara ait özel bilgilerini işleyebilme ve onların tecrübe ve sezgilerinden yararlanma fırsatını sunar. Bu yetenek kazandırılırken sayısal ifadeler kullanmak yerine sembollerin kullanılması amaçlanır. Bu sembollerin makinelere taşınması tamda matematiksel bir temele dayanır. Bu matematiksel temele bulanık kümeler kuramı denilmektedir (Elmas 2003).

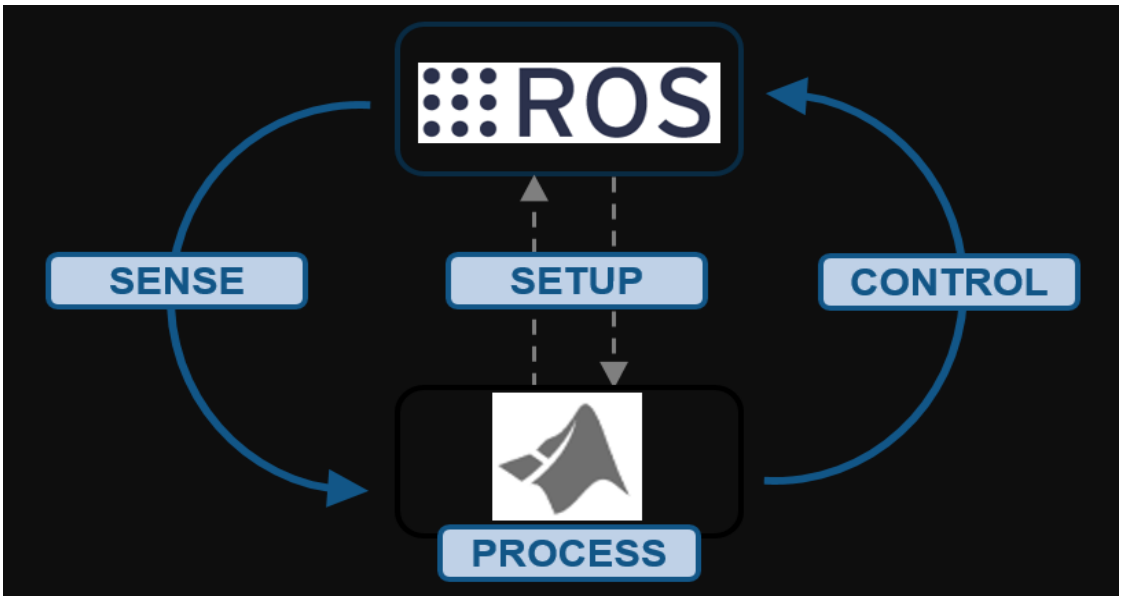
3. MATERYAL ve YÖNTEM

3.1. Materyal

3.1.1. ROS

3.1.1.a. Robot işletim sistemi (ROS)

Robot işletim sistemi, Şekil 3.1'de görüldüğü üzere robotlar için açık kaynaklı bir işletim sistemidir. Bir işletim sisteminden beklenen; donanım katmanının soyutlaştırılması, yüksek çalışma performansı olmayan cihaz kontrolü, işlevselliğin yaygın olarak kullanılması, yapılan işlemler arasında mesaj iletimlerinin kontrolü, paket trafiğinin yönetimi gibi hizmetleri sağlamaktadır. Çeşitli işlemlerin yapılması amacı ile araç ve kütüphaneler robot işletim sistemi içinde bulunmaktadır. ROS iletişim alt yapısı olarak gevşek bağ ile birleştirilen eşler arasındaki süreç ağıdır. ROS gerçek zamanlı çalışan bir yapı değildir, fakat gerçek zamanlı bir kod ile çalışabilir. Gerçek zamanlı çalışmasını sağlamak için bazı eklentiler ile bu problemin önüne geçilebilir (Dattalo 2018).



Şekil 3.1. Robot işletim sistemi (Mathworks 2017)

İlk olarak 2000'li yılların ortasında Stanford yapay zekâ laboratuvarlarında kişisel robotların geliştirilmesini desteklemek ve yenilikçi çalışmalar yapmak için ortaya çıkmıştır. 2007 yılında Californiya Menlo Park'ta kurulmuş olan Willow Garage şirketi Stanford yapay zekâ laboratuvarlarında yapılan ROS geliştime çalışmalarında yer almıştır. Böylece robotik kullanımı için geliştirilen bu yapı daha esnek ve dinamik bir hal almıştır. Bu yapı BSD lisansı altında geliştirilmiş ve uygulamalar yapılması amacıyla birçok uzman geliştirme sürecinde çalışmıştır. Bu gelişmeler sonucunda, robotik çalışmalar geniş kullanıcılara ulaşan bir platform haline gelmiştir.

2013 yılında robot işletim sistemine ait çekirdek gelişimi ve bakımı, çalışmalarına açık kaynak kod olarak devam eden robotik vakfına transfer olmuştur. Günümüzde küçük çapta yapılan hobi çalışmaları dahil olmak üzere çok büyük endüstriyel otomasyon sistemlerinde kullanılmaktadır. Son yıllarda artan robotik çalışmaları ile tüm dünya genelinde birçok kullanıcı uygulamalarını robot işletim sistemi ile geliştirmektedir (Jusuf 2016).

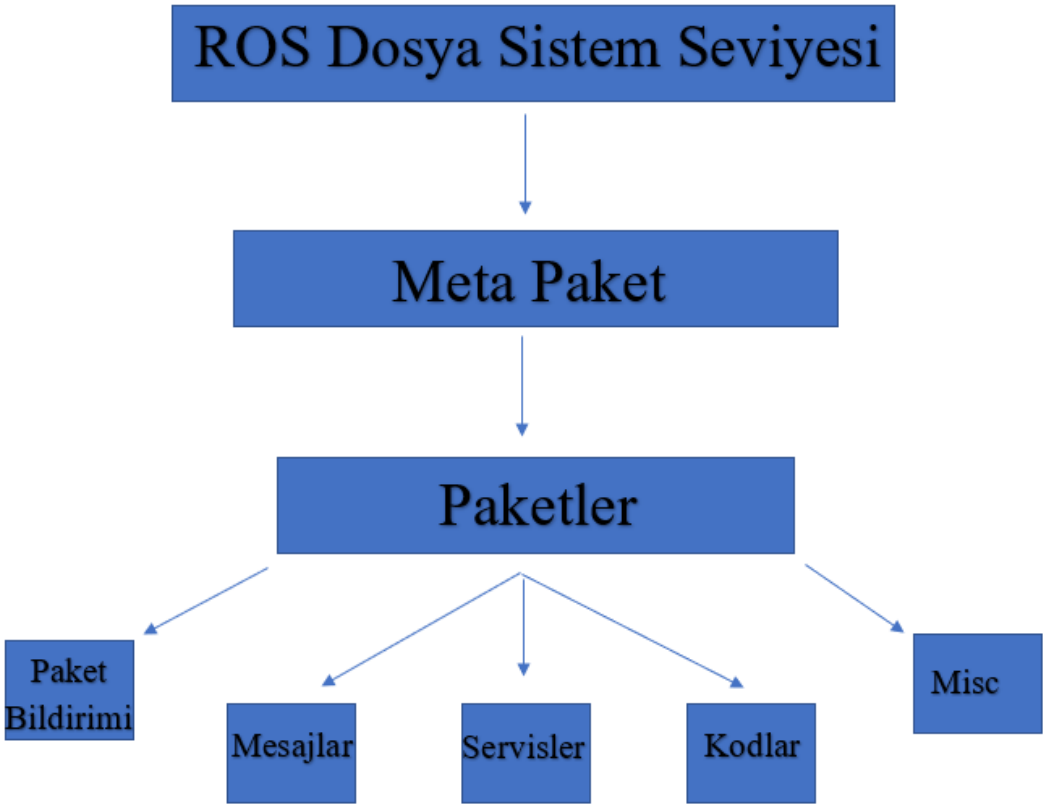
Robot işletim sistemi Linux tabanlı işletim sistemlerinde çalışmaktadır. Fakat bu işletim sistemleri, hala deneysel aşamadadır. Ubuntu işletim sistemi ROS için daha ideal bir işletim sistemidir. Ayrıca düşük seviyeli cihazlar için erişilebilirlik ve birçok yazılım paketleri sağlar. Belirli düğümleri (nodes) kullanarak bir ağ yapısının oluşturulduğu araç tabanlı dağıtık mimariye sahip bir yazılımdır (Claessens *et al.* 2013).

Robot işletim sisteminin yapısını inceleyecek olursak 3 temel üzerine kurulmuştur. Bunlar (Jusuf 2016):

- Dosya sistem düzeyi
- Hesaplamalı grafik düzeyi
- Topluluk düzeyi

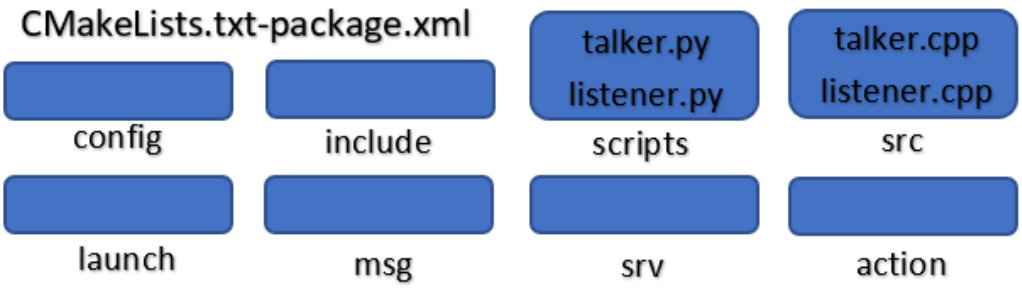
3.1.1.b. Dosya sistem düzeyi

Robot işletim sistemi, dosya sistemi çalışma mantığı olarak normal bir işletim sisteminden farklı değildir. ROS dosyaları sabit disk üzerinde belirli bir biçimde düzenlenir. Dosya yapısının nasıl şekillendiği aşağıdaki Şekil 3.2'de belirtilmiştir.



Şekil 3.2. ROS Dosya yapısı (Joseph and Cacace 2018)

Paketler: ROS paketleri robot işletim sisteminin en temel birimleridir. Bu paketler içerisinde kullanılan nesneler; ROS çalışma zamanı süreci, kurulum dosyaları ve kütüphanelerdir. Bu bahsedilen nesneler bir araya geldiğinde tekbir birim olarak çalışırlar.



Şekil 3.3. ROS birimleri (Joseph and Cacace 2018)

Yukarıda gösterilen Şekil 3.3'de belirtilen ifadeleri incelemek gerekirse (Joseph and Cacace 2018)

- Config : ROS paketleri tarafından kullanılan bütün kurulum dosyalarının bulunduğu dosyadır. Bu dosya kullanıcı tarafından oluşturulur.
- Include / Package: Paketin içinde kullanılması gereken üstbilgilerden ve kütüphanelerden oluşur.
- Scrpits: Bu dosya içinde çalıştırılabilir hale getirilmiş python dosyaları bulunur.
- Src: C++ programlama dili ile yazılmış kaynak kodların depolandığı dosyadır.
- Launch: Bir veya daha fazla nodu çalıştırmak için kullanılan başlangıç dosyalarını barındırır.
- Msg: Özel mesaj tanımları bu klasör içindedir.
- Srv: Servis tanımlarının yapıldığı klasördür.
- Action: hareket tanımlarının yapıldığı dosyadır.
- Package.xml: bir paketin paket bildirim dosyasıdır.
- Cmakelist.txt: paketin cMake yapı dosyasıdır.

Ayrıca ROS paketi ile çalışırken ihtiyacımız duyulan oluşturma, değişiklik ve ROS paketiyle çalışmak için bazı komutları bilmemiz gerekiyor (Joseph and Cacace 2018).

- Catkin_create_pgk: Bu komut satırı yeni bir paket oluşturmak için kullanılır.
- Rospack: Dosya sistemi içinde bulunan paket hakkında bilgi almak için kullanılır.

- Catkin_make: Çalışma alanı içinde bulunan paketin yapılandırılması için kullanılır.
- Rosdep: Paket için gerekli olan tüm bağımlı sistem bileşenlerini yüklemek için kullanılır (Joseph and Cacace 2018).

ROS, paketleri çalıştırmak için Linux tabanlı işletim sistemlerinde bulunan Bash komut satırı işlemleri gibi işlemleri çalıştırır.

- Roscd: Bu komut geçerli dizini değiştirmek için kullanılır.
- Roscp: Bir paket içinden dosya klopyalamak için kullanılır.
- Rosed: Vim editör ile dosyayı düzenlemek için kullanılır.
- Rosrun: Paket içinde bulunan çalıştırılabilir dosyanın çalışmasını için kullanılır (Joseph and Cacace 2018).

ROS dosya sistem yapı seviyesi birimleri aşağıda açıklanmıştır.

Paket Bildirimi: Paket bildirim dosyası, paket hakkında bazı bilgiler içerir, paketin lisansı, bağlı olduğu diğer paketler, derleme bayrakları. Paket bildirim dosyası package.xml içinde yer alır.

Meta paketi: Özel bir amaç için kullanılan paketler grubudur.

Meta paketler bildirimi: Çalışma mantığı olarak paket bildirimine çok benzemektedir. İçinde bulunan paketlerin çalışma zaman bağımlılıklarını ve bunların dışa aktarımını bildirirler.

Mesajlar: Robot işletim sisteminde yapılan bir işlemin diğer yapılacak veya yapılmakta olan işlemlere gönderdiği bilgi türüdür. Ayrıca paket içinde bulunan msg klasörü içine özel iletiler tanımlanarak mesaj dosya uzantısı .msg olarak kaydedilir.

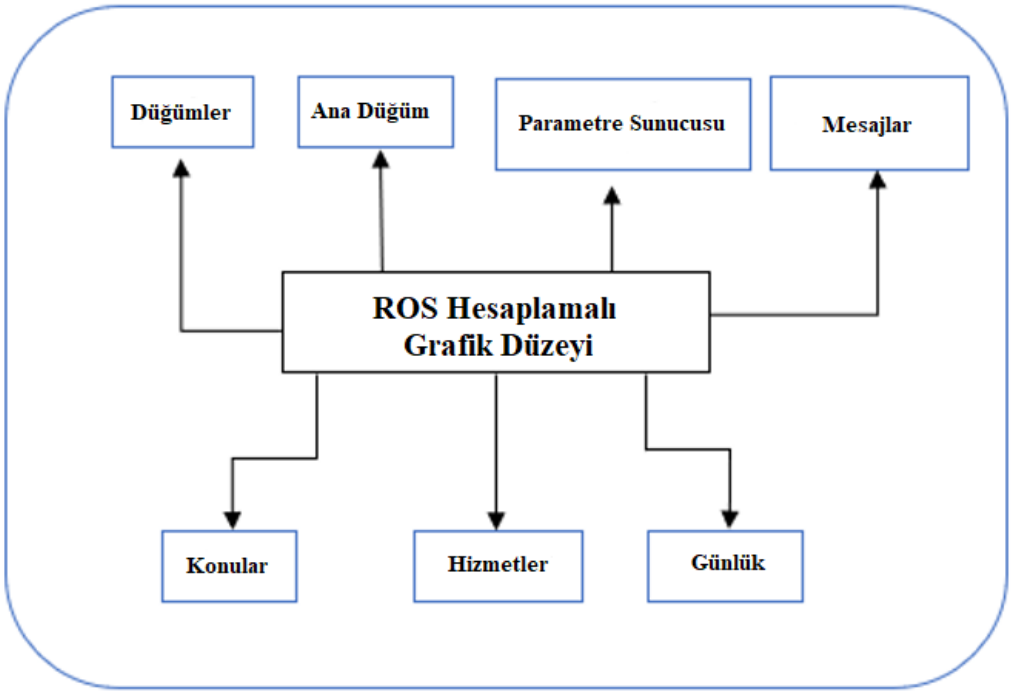
Hizmetler: ROS hizmetleri süreçler arasında bir tür istek/cevap etkileşimidir. Yanıt ve istek türleri paketin içinde srv klasörü altında tanımlanabilir.

Depolar: ROS paketlerinin birçoğu sürüm kontrol sistemleri kullanılarak korunmaktadırlar. Depolarda toplanan bu paketler catkin adı verilen bir çalışma alanında kullanılır.

3.1.1.c. Hesaplamalı grafik düzeyi

Robot işletim sisteminde yapılan işlemlerin tamamı, ROS düğümleri (nodes) ile oluşturulmuş bir ağ yapısı üzerinden yapılmaktadır. Bu hesaplama ağı aslında hesaplama grafiği olarak isimlendirilir. Ana kavramlardan bahsedecek olursak; ROS düğümleri (nodes), Parametre sunucusu (parameter server), Master, mesajlar (messages), çantalar(bags), servisler(services) ve konulardır(topics) (Joseph and Cacace 2018).

Roscpp ve rospython gibi çekirdek istemci kütüphanelerinin barındırıldığı ROS iletişimini sağlayan paketler, düğümler, konular, parametreler ve kavramların hizmetlerini ve uygulanmasını ros_comm adlı bir yığında bulmak mümkündür (Joseph and Cacace 2018). Bu süreçte katman/iletişim paketleri bulunmaktadır. Bu paketler bir bütün olarak ROS “Graph” katmanı olarak bilinir. Katman/iletişim paketleri topics, nodes, services ve parametreler için araç ve uygulama sağlar. Bu süreç kullanıcı kütüphanelerinden sağlanmaktadır (ROS.org 2015).



Şekil 3.4. ROS hesaplamalı grafik düzeyi (Joseph and Cacace 2018)

Yukarıda gösterilen Şekil 3.4'de belirtilen ifadeler incelendiğinde

- Düğümler: ROS istemci kütüphaneleri kullanılarak yazılmış olan düğümler hesaplama işlemlerinin yapıldığı birimdir. Bu süreçte farklı düğümler birbirleri arasında iletişim halinde olurlar. Burada düğümlerin asıl amacı çok kapsamlı bir iş akışı oluşturmak yerine daha basit bir yapı kurmak ve birçok düğümü bir araya getirerek daha işlevsel bir sistem oluşturmaktır. Böylece bir hata ile karşılaşıldığında hatayı ayıklamak çok daha basit olur.
- Master: İsim kaydı sağlayan Master geri kalan tüm düğümlerin arama işlemini yapar. Hiçbir düğüm Master olmadan hizmet çağıramaz ve başka düğümler ile iletişim sağlayamaz.
- Parametre Sunucusu: Verilerin tek merkezli bir konumda saklanmasını sağlar. Dağıtık mimaride bulunan tüm düğümler bu merkeze ulaşabilir ve bunları değiştirebilir. Bu sunucu Master'ın bir parçasıdır.

- Mesajlar: Bütün düğümler mesajları kullanarak kendi aralarında bir iletişim kurarlar. Özet olarak üzerinde bir veri sınıfı tutabilen ve başka bir düğüme bunu iletebilen bir veri yapısıdır. Standart tipte değişken değerleri vardır (ondalıklı sayılar, tamsayı, Boolean, vb.). Bunlar ROS mejaları tarafından destelenen değerlerdir.
- Konular: Her bir mesaj konu olarak adlandırılan yapılar vasıtası ile taşınır. Bir düğüm ancak bir konu üzerinden mesaj gönderdiğinde konunun yayınlandığı anlaşılmaktadır. Bir düğüm bir konu vasıtası ile mesaj aldığında düğümün bu konuya bağlandığını söyleyebiliriz. Yayınlama ve abone olma düğümleri birbirinden habersiz çalışır. Her konu için kendine has bir adı vardır. Herhangi bir düğüm bu konuya çok rahat erişebilir ve doğru mesaj türüne sahip oldukları sürece veri gönderilebilir.
- Hizmetler: Bazı durumlarda robot uygulamaları yayın / abone mimarisine uygun bir iletişimi olmayabilir. Bazen bir düğüm başka bir düğümden çok hızlı bir biçimde işlemin yürütülmesini isteyebilir. Böyle bir durumda istek/yanıt mimarisi devreye girer.Robot işletim sisteminde servis etkileşimi uzaktan çağrı yöntemi gibidir.
- Günlüğe Kaydetme: Robot üzerinde bulunan ve herbirinden veri alınan sensörler bulunmaktadır. Çeşitli algoritmalar geliştirmek ve test etmek için sensörlerden gelen tüm verileri depolandığı bir kayıt sistemdir. Çanta dosyaları çok fonksiyonlu ve karışık robot bileşenleri ile çalışırken süreci ciddi anlamda kolaylaştırmaktadır (Joseph and Cacace 2018).

3.1.1.d. Topluluk düzeyi

Burada amaçlanan gelişmekte olan robot işletim sisteminin gelişimi için yazılım ve bilgi paylaşımda bulunulabileceği bir topluluk sağlamaktır. Bu topluluk için bazı kaynaklar mevcuttur. Linux işletim sistemlerinin dağıtım sürecine benzer bir yapıya sahiptir. Meta paketler bir araya getirilerek yazılımın daha kolay kurulması ve toplanması hedeflenmiştir. Tutarlı sürümleri belirli dönemlerde kullanıcıların hizmetine sunulur. Kodların güvenliğini sağlamak amacıyla yazılım bileşenlerinin depolandığı bir yapı mevcuttur (Joseph and Cacace 2018).

ROS Wiki tam olarak robot işletim sistemi hakkında sahip olunabilecek tüm dokümanların farklı farklı kullanıcılar tarafından hazırlandığı bir platformdur. Herkes buraya kayıt olarak bilgilerini paylaşabilmektedir. Ayrıca hata bileti sistemi ile ROS üzerinde yaşanan bir hata olduğunda veya yeni bir şey eklenmek istenirse bu kaynak kullanılabilir. Ayrıca kullanıcılara ROS'üzerinde yapılmış güncellemeler posta listeleri ile kullanıcılara iletilmektedir (Joseph and Cacace 2018).

ROS işletim sistemi ile uyum içinde çalışan, üzerinde bulunan sensörler vasıtası ile otonom harekete elverişli olan bir robot. Düşük bütçeli olması sebebiyle bu çalışmada Turtlebot 2 kullanılması amaçlanmıştır.

3.1.2. Turtlebot 2

Turtlebot düşük maliyetli açık kaynaklı kodlamaya sahip profesyonel veya hobi amaçlı robotik çalışmalarda kullanılan bir robot kittir. 2010 yılında Melonee Wise ve Tully Foote'un çalışmaları ile Willow Garage'da yaratılmıştır (Open Source Robotics Foundation 2014).

Turtlebot bir yıl içinde araştırmacılar ve hobi için kullanılanlar arasında çok ciddi bir başarı yakalanarak ve yeni nesil platformlar ile kullanıcıların tanıştırılması gerektiği düşünülmüştür. Yeni platform olarak tanıtılan Turtlebot 2 aslında ilk olarak, piyasaya sunulan Turtlebot'dan çokta farklı değil ve Turtlebot'a benzetilerek tasarlanmıştır. Turtlebot'un yerine Turtlebot 2 almasının temelinde yatan en büyük neden ürünün ABD dışına satılamamasıdır. Çünkü elektronik sertifikaya sahip olmaması ve 220 voltta çalışmıyor olmasıdır. Bu sebep ile farklı kıtalar için dağıtım yapılması amacı ile yeni bir model tasarımına ihtiyaç duyulmuştur (Ackerman 2012).

Güney Korede yerli bir robotik şirketi olan Yujin Robot, yeni bir platformu yapmaya karar vermiştir. İlk versiyonda yaşanan ciddi problemler masaya yatırılarak Yujin Robot'tan istenilen iyileştirmeler ile yeni bir platform oluşturmak esas alınmıştır. Temel olarak revize edilmiş bir Turtlebot ortaya çıkmış fakat firma daha fazla özelliği

bünyesinde barındıran bu robota Kobuki adını vermiş ve Turtlebot 2 olarak piyasaya piyasaya sunmuştur (Ackerman 2012).

İlk nesil Turtlebot ile Kobuki arasında çok fazla değişiklik mevcuttur. Bu değişiklikler aşağıda belirtilmiştir.

- Daha hassas ve doğru konum bulabilen tekerlekler kullanılmaktadır.
- Daha fazla kullanım ömrü sağlayacak fazladan bir pil paketi mevcuttur. Yaklaşık olarak 2-3 saatlik bir kullanım ömrü sağlamaktadır.
- Robot üzerine konumlandırılmış güç ve iletişim soketleri mevcuttur. Kolay bir biçimde erişimi mümkündür.
- Daha fazla enerji çıkışı ile birlikte dizüstü bilgisayarlar, sensörler ve farklı diğer bileşenlere güç sağlamaktadır.
- Hareket motorları daha hızlı ve daha sessiz çalışmaktadır.

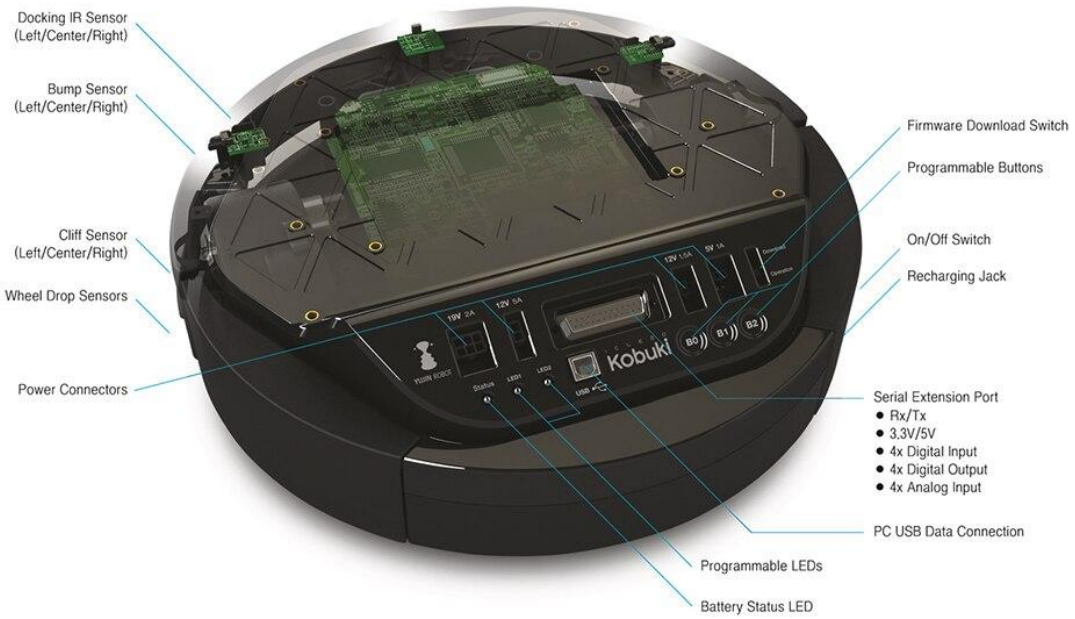
Ayrıca Turtlebot2 ile gelen kendi kendine sarj edebilen bir istasyonu kullanabiliyor olmasıdır. Bu ekipman ile robutu her zaman sarj etme gereksinimi ortadan kaldırılmıştır (Ackerman 2012).

Teknik özellikleri bakımından Turtlebot2 incelenecek olursa

- Ebat ve ağırlık
  - 6.3 kilogram ağırlıkta, 420 mm yükseklik, 354 mm genişlik, uzunluğa sahip olan robotun tekerlek yüksekliği 76 mm ve yerden yüksekliği ise 15 mm dir (Clearpath Robotics 2014).
- Hız ve performans
  - Maksimum taşıma kapasitesi 5 kg olan robotun, en yüksek hızı 0.65 m/s ve maximum dönme hızı 180˚/s dir (Clearpath Robotics 2014).
- Batarya ve Güç Sistemi

- Standart olarak üzerinde 2200 mAh Li-lon bataryaya sahip olan robotta genişletilmiş 4400 mAh Li-lon bir batarya daha mevcuttur. Kullanıcıya 5V, 19V, 12V (1.5A), 12V(5A) güç sağlamaktadır (Clearpath Robotics 2014).

- Sensörler
  - 3d Görsel Sensör (Kinect Kamera) : Kinect kamera kızıkötesi ışın yayan aynı zamanda RGB bir sensördür (Fu *et al.* 2012) . Cihaz üzerinde bir adet standart bir kamera, bir adet mikrofon, bir adet mesafe ölçer ve kızılötesi sensör mevcuttur.Bu sönsörler temel olarak derinliği ölçmek (640x480 piksel, 30 FPS derinlik), hareket algılamak(640x480 piksel) ve ses taramak(4 kanal) için kullanılır. Bütün bu sistemlerin toplandığı tüm bir sistemin başarılı bir biçimde çalışması için 6,25 m^2 bir alan gerekmektedir. Aksi taktirde nesnelerin belirlenmesinde problem yaşanmaktadır. Hareketlerin algılanması için en az 2 metrelik bir mesafe gerekmektedir. Ayrıca ortamda bulunan siyah renge sahip cisimler tanımlama işlemi esnasında problem çıkarabilmektedir (Terun 2011).
  - Yardımcı sensörler : 3 adet forwer bump, 3 adet cliff ve 2 adet whell drop sönsörü mevcuttur (Clearpath Robotics 2014).



Şekil 3.5. Turtlebot 2 sensörler birimi

Şekil 3.6. Turtlebot 2 genel görünümü

Bu çalışma süresi boyunca turtlebot 2 üzerinde yapılan tüm çalışmalar gerçek robottan önce gazebo simülasyon ortamında test edilerek ne gibi sonuçlar verdiği gözlemlenmeye çalışılmıştır.

3.1.3. Gazebo simülasyon

Robot simülasyonu her robot çalışmasında önemli bir yere sahiptir. Etkili bir şekilde tasarlanmış bir simülatör, gerçek bir robot ile test yapmak yerine sanal bir ortamda algoritmaları test etmek, yeni bir robot tasarlamak, çeşitli seneryoları oluşturmak ve yapay zeka sistemlerini eğitmek için kullanmamızı mümkün kılar. Gazebo, belirli veya kullanıcı tarafından istediği gibi yaratılan karmaşık iç ve dış mekanlarla, robotların daha

doğru ve daha yüksek başarı oranına sahip verimli bir biçimde simüle etmeye yardımcı olmaktadır. Alt yapısında ciddi bir fizik motoruna sahip olan Gazebo, kalite seviyesi yüksek grafikler ve kullanılması kolay grafik arayüzüne sahiptir. Gazebo simülasyonu canlı bir topluluk ile yaşam döngüsünü sürdürmekle beraber ücretsiz olarak kullanıcılara hizmet vermektedir (Open Source Robotics Foundation 2014).

Gazebo Simülasyonunun gelişimi 2002 yılında Güney Californiya Üniversitesi'nde başlamıştır. İlk olarak Dr. Andrew Howard ve onun öğrencisi Nate Koenig tarafından ortaya koyulmıştur. Temel olarak dış mekan ortamlarında bulunan robotların yüksek oranda uygunluk simülatör ihtiyacına cevap verebilmek için yaratılmıştır. Dış ortam simülasyonu olarak ortaya konulmasına rağmen birçok kullanıcı kapalı ortam simülatörü olarak kullanmaktadır (Open Source Robotics Foundation 2014).

2009 yılına kadar doktora çalışmalarına devam eden Nate Koenig bir yandan da Gazebonun gelişimi için yaptığı çalışmaları sürdürmüştür. Willow'da üst düzey bir araştırma mühendisi olarak çalışan Jhon Hsu, ROS toplulukları için birincil araç olarak kullanılacak olan Gazebo robot işletim sistemine entegre edilmiştir. 2011 yılında Willow Garage Gazebo simülasyonunun gelişimi için maddi destek sağlamaya başlamıştır. 2012 yılında Willow Garage'dan ayrılan Açık Kaynak Robotik Kuruluşu (OSRF) Gazebo simülasyon projesinin yöneticisi konumuna gelmiştir. Çok sayıda başarılı geliştiricinin çalıştığı bu proje ile 2013 yılında yapılan ve Robotik alanda ciddi çalışmaların olduğu Virtual Robotics Challange'a Gazebo ile katılmışlardır. Günümüzde aktif olarak görev alan çok çeşitli topluluklar ile beraber yapılan çalışmalar ile Gazebo Simülasyonu gelişmeye devam etmektedir (Open Source Robotics Foundation 2014).

Gazebo karmaşık yapıda bulunan iç ve dış ortamlarda robotlar için doğru ve verimli bir biçimde simüle etme özelliğine sahip 3 boyutlu değişken bir simülatördür. Aslında oyun motorları ile aynı kapsamda yer alıyor olsada birçok sensör ve fizik motoru ile kullanıcı ve programlara yüksek kapasiteli uygunluk sağlar.

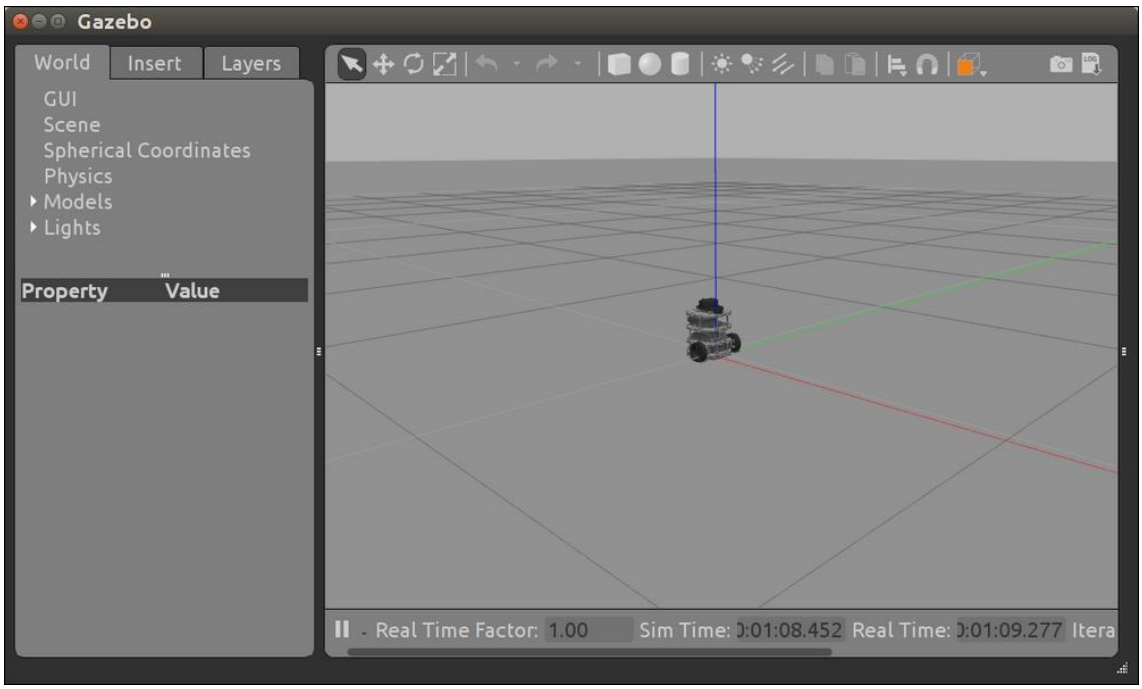
Gazebo için genel kullanım alanları şunlardır :

- Robotik algoritmaları test etme
- Yeni bir robot tasarımı
- Gerçekçi senaryoları yüksek uygunluk ile test etme

Gazebo için temel özellikler ise:

- Çoklu fizik motorlarına sahip olması
- Zengin bir kütüphaneye sahip olması
- Çok çeşitli robot modellerine sahip olmasıdır (Open Source Robotics Foundation 2014)

Gazebo günümüzde en etkili olarak Linux tabanlı Ubuntu işletim sistemi üzerinde çalışmaktadır. Çalışması için sistem gereksinimi olarak; en az intel i5 işlemci ve bu seviyede çalışan başka bir CPU, özel bir GPU; tavsiye edilen Ubuntu işletim sistemi ile uygun çalışma eğiliminde olan Nvidia ekran kartı ve en az 500 MB boş disk alanına ihtiyaç duyulmaktadır (Open Source Robotics Foundation 2014).



Şekil 3.7. Gazebo simülasyon dünyası

Bu çalışmada Gazebo simülasyon ortamında çalışan Turtlebot 2 robotumuzun ve gerçek robottan gelen tüm sensör verilerinin 2 boyutta haritalama görsel sunumu Rviz ile yapılmıştır.

3.1.4. Rviz

Rviz, ROS için kullanılan 3 boyutlu görselleştirme ortamıdır. Gazebo simülasyonunda kullandığımız robotun görünümü, robot sensörlerinden gelen verilerin alınması ve alınan bu verilerin görselleştirme ortamında kullanıldığı bir araçtır (AWS RoboMaker 2019). Kullanılan robotun bir modelinin rviz içinde görüntülenmesi ve robot üzerinde bulunan sensörlerden gelen verilerin kaydedilmesini ve tekrar tekrar bu verilerin kullanılmasını sağlar. Robotun harita üzerinde ne gördüğü, düşündüğünü ve ne yaptığını görselleştirmesi için kullanılır. Rviz stereo kameralardan, kinectden, lazerden ve diğer 3 boyutlu görüntü algılayan sensörlerden gelen 3 boyutlu verileri nokta kümeleri ve bu kümeye ait derinlik verilerine sahip bir biçimde görürler. Ayrıca 2 boyutlu görüntü işleyebilen kameralar, uzaklık ölçebilen lazerlerden gelen veriler, görüntü veri kümesi olarak Rviz'de görüntülenir. Gerçek robotun mevcut yapısı Rviz ile sanal robot modeline dönüştürülür. ROS konuları (topic) ve tüm sensör verileri, gerçek robottan yayınlanan sensör verilerini esas alarak gerçek bir gösterim gibi davranacaktır. Gerçek bir robotu sanal bir ortama aktarmak, robotun sistemlerini, kontol sürecini ve karşılaşılacak hatarı ayıklamak adına ciddi anlamda kolaylık sağlayacaktır. Kullanıcıların similasyon ortamında robota verdiği görev bilgilerinin görüntülenmesini sağlayan ve yapılandırılabilen grafik kullanıcı arabirimi (GUI) görevi sağlar (Fairchild and Harman 2016). Tüm bu işlemlerin yapılması için bir simülasyon ortamanına bağlı olmalıdır (AWS RoboMaker 2019).

3.1.5. Ardunio

Ardiuno, açık kaynaklı kullanımı kolay elektronik platformdur. Bir çok amaç için kullanılan ardiuno kartları; sensör üzerindeki bir işık verisini, herhangi bir düğmeden gelecek veriyi, açma kapama anahtarına ait giriş verilerini okuyabilir ve bunları bir çıkışa çevirir (Arduino 2019).

Bord üzerinde bulunan mikro işlemcilere gönderilen talimatlar seti ile neler yapılabileceği belirtilir. Bunu yapmak için Wiring temelli Ardunio programla dili ve Ardunio yazılımı olan tümleşik geliştirme ortamı kullanılır (Arduino 2019).

Wiring 2003 yılında Hernando Barragán tarafından üzerinde çalışılmaya başlanan bir projedir. Temel olarak mikro işlemciler için kullanılan açık kaynak kodlu programlanabilir bir çerçevedir (framework). Mikro denetleyici kartlarına bağlı olan bütün cihazların kontrolünü sağlayan platform bağımsız yazılım yapılmasına imkan sağlar (Wiring).

Ardunio uno bir mikrodenetleyici karttır. Kart üzerinde 14 adet dijital ve 6 adet analog giriş/çıkış pini,16 Mhz hızında çalışan bir adet seramik resonatör, 1 adet USB portu mevcuttur. USB üzerinden veri gönderimi ve aynı zamanda güç almak için kullanılır. 1 adet reset düğmesi ve bir adet ICSP başlığı bulunur.

3.1.6. Bme 280 Sıcaklık Basınç Nem Sensörü

Mobil uygulamalarda kullanılmak üzere yaratılmış koşul ve şartlara duyarlı entegre yapıda bir çevre sensörüdür. Günümüzde mobil uygulamalarda kullanılan donanım bileşenlerinin boyut olarak küçük ve enerji tüketimi olarak en az seviyede olması beklenmektedir. Bu ünite 3,6 µA @ 1Hz akım tüketimi sağlamaktadır. Böylece uzun çalışma süresi boyunca kararlı bir yapıda çalışma özelliğine sahiptir. 8 pinli metal kapaklı bu çevre sensörü, basınç ile birlikte sıcaklık ve nem değerlerini hassas bir biçimde ölçmektedir (Bosch Sensortec GmbH 2019).

Bu entegre sensör, içinde bulunan nem sensörü geniş bir nem farkındalığı ve sıcaklık değer aralığında oldukça hassas ölçümü ile doğruluk derecesi yüksek sonuçları çok hızlı bir tepki süresinde kullanıcılara verir. Basınç Sensörü içinde bulunan barometre yüksek hassasiyet değerine sahiptir. Sıcaklık sensörü ise çok düşük değerdeki gürültü ve yüksek çözünürlükte hassas ölçüm yapabilecek duruma getirilmiştir. Başınç ve nem sensörleri

sıcaklık sensörünün daha etkili olarak çalışması için sıcaklık yanılma payını azaltmak için kullanılır (Bosch Sensortec GmbH 2019).

Bme 280 teknik açıdan aşağıdaki özelliklere sahip olmalıdır. (Bosch Sensortec GmbH 2019);

- 8 pin ve metal kapak
- 2,5 x 2,5 x 0.93 mm^3 boyutunda
- -40 ℃ ile 85 ℃ arasında sıcaklık ölçümü
- 300hPa ve 1100hPa arasında basınç ölçümü
- 1.71 – 3.6 V besleme gerilimi
- 1.8 µA @ 1 Hz (H, T), 2.8 µA @ 1 Hz (P, T) ve 3.6 µA @ 1 Hz (H, P, T) değerlerinde ortalama akım tüketimi
- 0.1 µA uyku halinde ortalama akım tüketimi

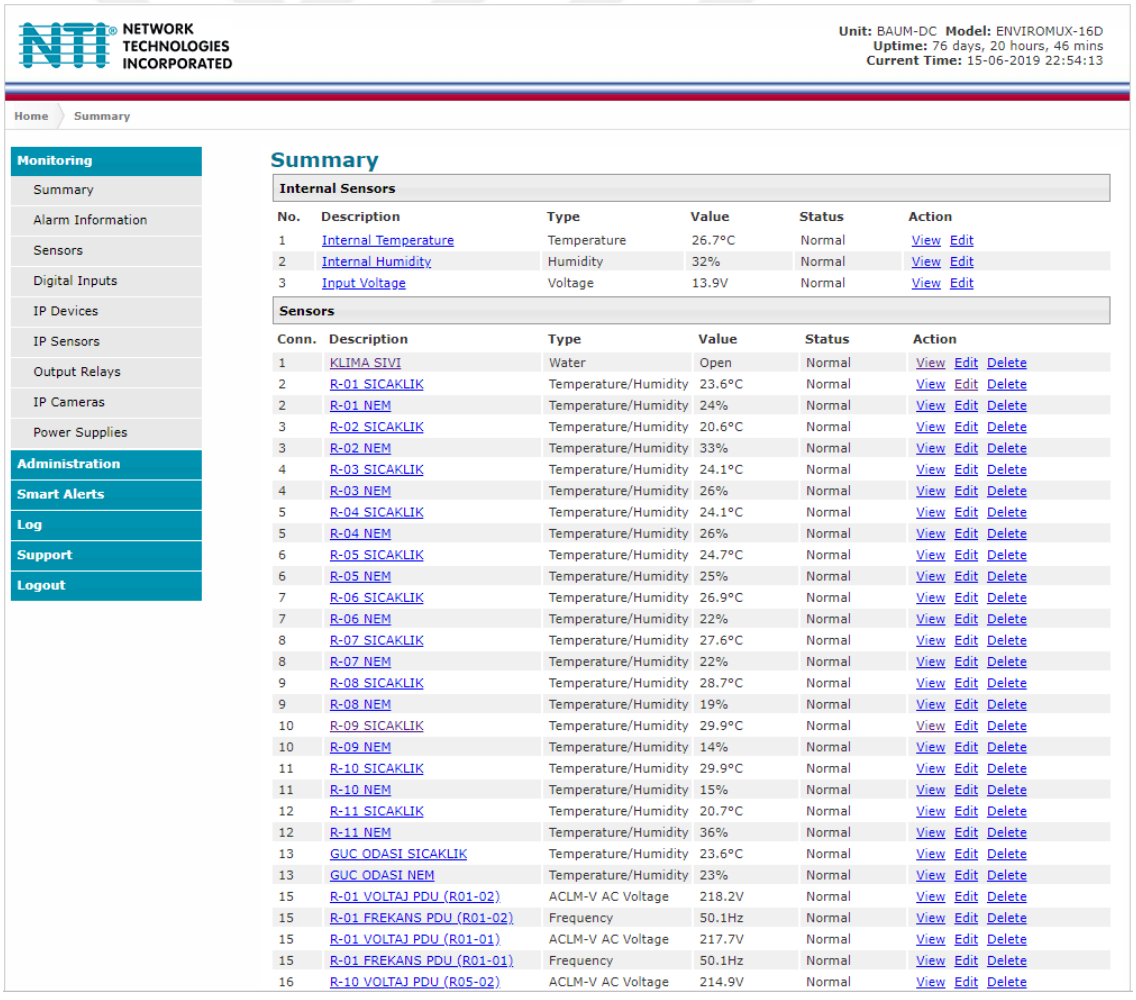
3.1.7. NTI E-STS sıcaklık ve nem sensörü

Büyük ölçekli sistem odalarında kullanılan bu sensörler ortam içi sıcaklık ve nem değer ölçümü yapmaktadır. Şekil 3.10'da gösterilen NTI E-STS Sıcaklık ve Nem Sensörü de onlardan biridir. NTI, çevresel görüntüleme sistemi kullanılır. Bu sistem arayüzü Şekil 3.8'de görüldüğü gibidir. Kabinet içinde konumlandırılmış (EK 14) Şekil 3.9'da gösterilen sensörler vasıtası ile kabinet içi sıcaklık ve nem değerlerini belirli aralıklarda kullanıcılara bildirilir. Teknik özellikler şu şekilde açıklanabilir.

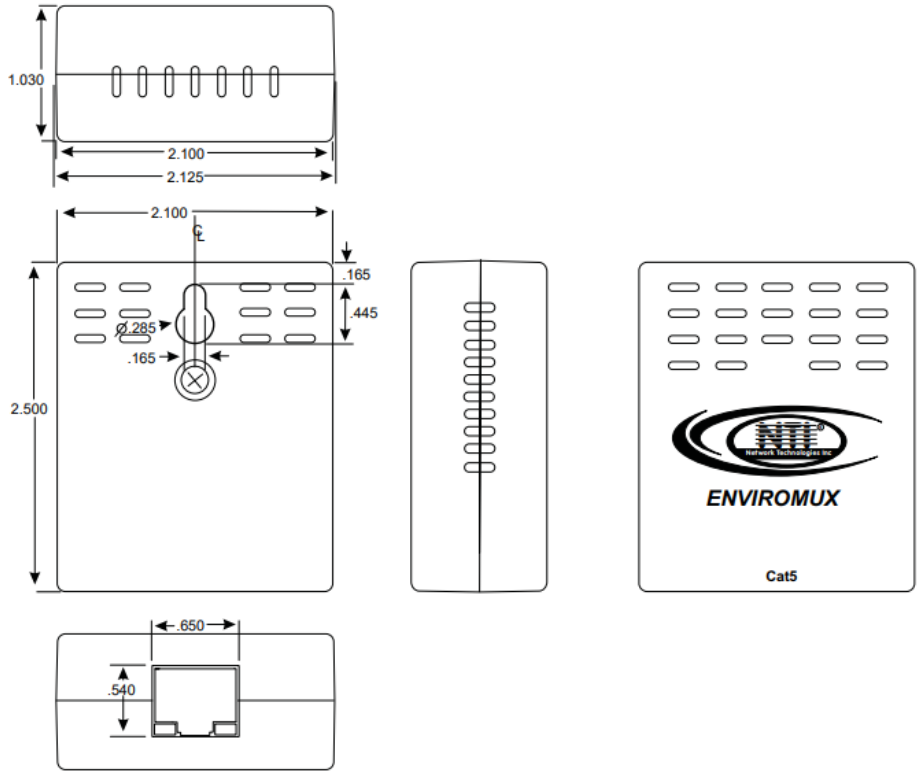
- 0℃ ile 50℃ arasında sıcaklık ölçümü
- ± 0,5℃ dahilinde hassas ölçüm
- Dış etkenlere karşı dirençli dijital çıkış sinyali veren kablolama birimi
- İlk kurulumda kalibrasyonun yapılması
- RJ45 bağlantı noktasına sahip olması
- 20mA akım tüketim değerine sahip olma

- 5 VDC gerilim besleme değerine sahip olma
- Rs485 iletişim protokolüne sahip olma
- Veri hızı olarak 96 kbps'dir
- Cat5 /5e /6 /6a kablolarını 305 metreye kadar destekler
- E-FCS fiber dönüştürücü ile uyumludur.

Sistem odası içinde sıcaklık ölçümleri yapılırken gezgin robotumuzun üzerine yerleştirdiğimiz, ardunio kart ile entegre ettiğimiz BME 280 sensöründen gelen veriler ile elde ederken, Kabinet içi sıcaklık değerlerinin NTI E-STS sıcaklık ve nem sensöründen elde edilmesi amaçlanmıştır.



Şekil 3.8. NTI çevresel görüntüleme sistemi arayüzü



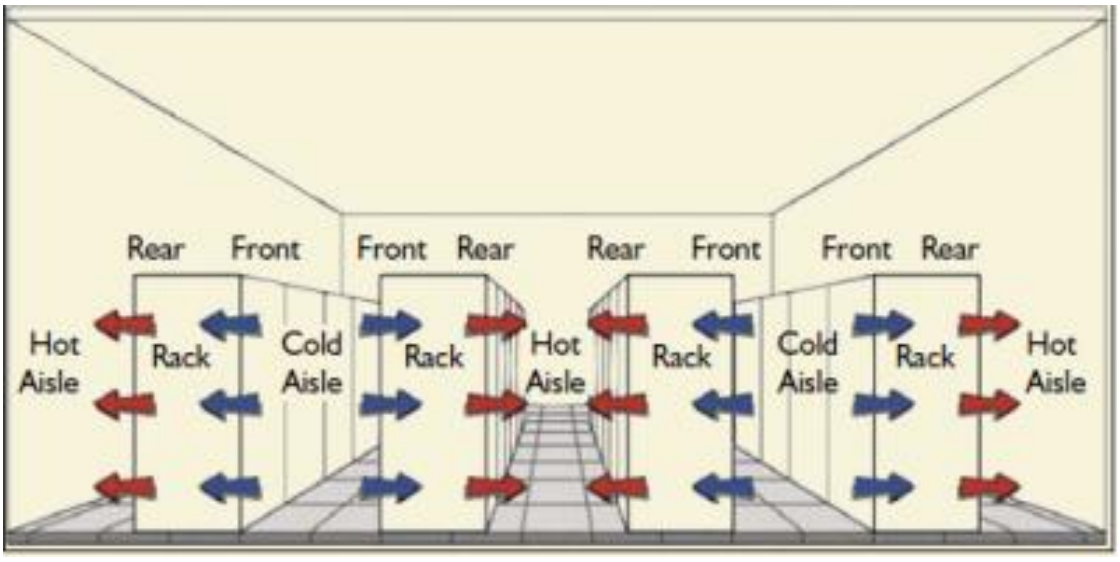
Şekil 3.9. E-STS sıcaklık ve nem sensörü (Network Technologies Incorporated 2019)

Şekil 3.10. E-STS sıcaklık ve nem sensörü görünümü (Network Technologies Incorporated 2019)

3.1.8. Sistem odaları iklimlendirme

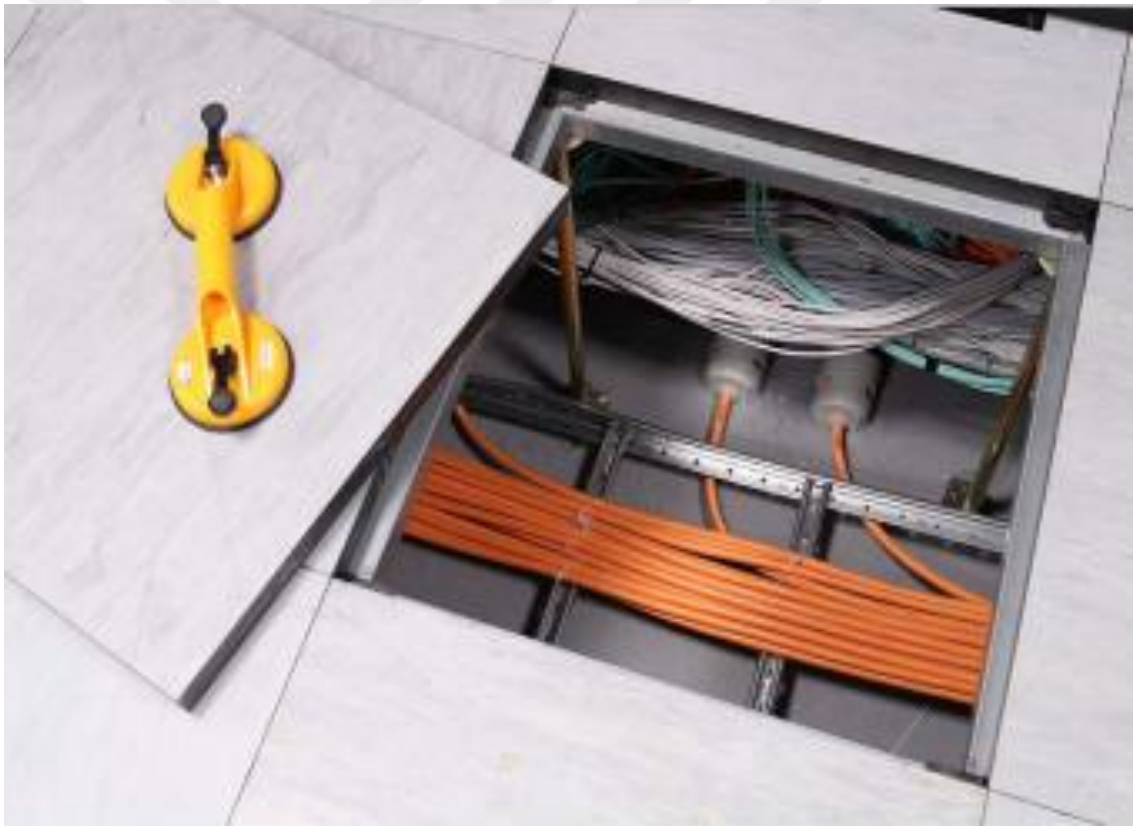
Sistem odaları tasarlanırken bazı standartlara (tier 1, tier 2, tier 3 ve tier 4) uygun bir yapıda tasarlanması önem arz etmektedir. Oda tasarlama sürecinde inşaat özellikleri, klimalar ve soğutma, yangın algılama ve söndürme sistemi, duman algılama sistemleri, tozdan koruma, su ile nem algılama sistemleri ve kablolama ile topraklama tesisatı göz önüne alınarak tasarlanır.

Sistem odalarında iklimlendirme, içinde bulunan sunucu ve diğer ekipmanların güvenilir ve performanslı çalışması için önemlidir. Sistem odası büyüklüğüne ve cihaz sayısını göz önüne alarak hassas ölçüm yapan iklimlendirme kilimaları yerleştirilir. Sistem odasında bulunan tüm cihazların ve gelecekte bu yapıya eklenecek cihazlar göz önüne alınarak bütün bu güç tüketimleri toplanarak kurulacak olan iklimlendirme sistemi tertip edilir. Soğutma işlemi yapılırken kabinlerin yerleşim düzeni önemlidir. Kabinler yerleştirilirken Şekil 3.11'de gözüktüğü gibi soğuk ve sıcak hava koridorları oluşturmaya dikkat edilmelidir.



Şekil 3.11. Kabinet konumlandırılması sıcak soğuk hava blokları (Kaya Sever 2016)

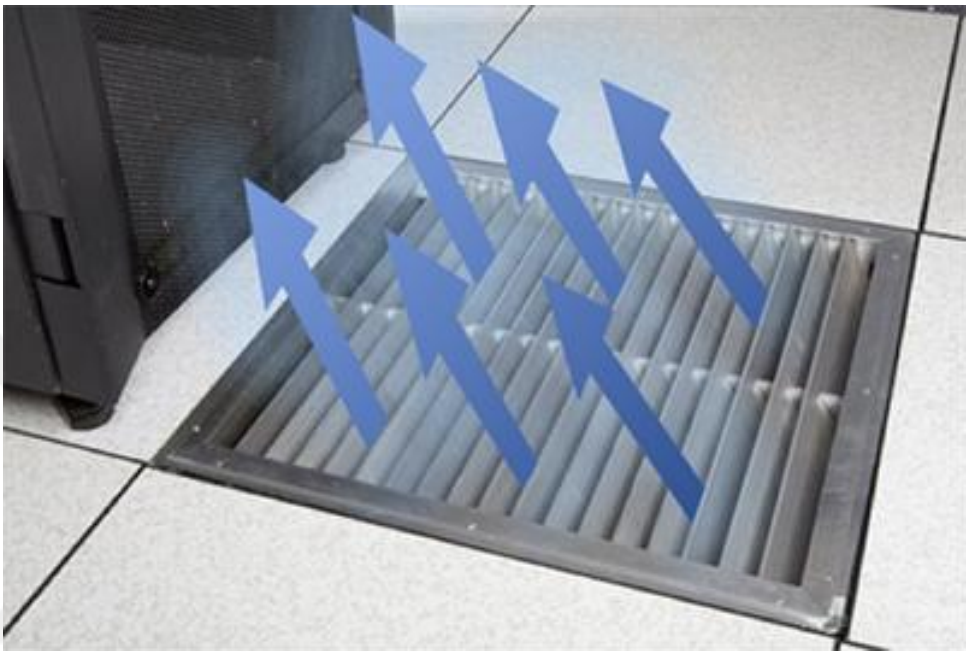
Sunucular, çalışma prensibi olarak önden aldıkları soğuk havayı, arka tarafta bulunan fanlardan dışarı atmaktadır. Bu sebepten dolayı kabin yerleşiminde sunucuların ön yüzleri soğuk hava koridorlarının bulunduğu kısıma denk gelecek şeklide konumlandırılmalıdır. Sistem odaları tasarlanırken mevcut alanın tamamında Şekil 3.12'de gösterildiği gibi yükseltilmiş taban kullanımı gereklidir (Kaya Sever 2016). Sistem odasında çalışan hassas ölçümlü klimalar çalışma prensibi olarak, soğuk ve temizlenmiş havaya yükseltilmiş taban altından sunucuların ön yüzünün bulunduğu soğuk hava koridorlarına gönderir ve ısınan sıcak havayı klimanın üstünden filtreleyerek bu döngünün sürekli olmasını sağlar.



Şekil 3.12. Yükseltilmiş taban (Kaya Sever 2016)

Yükseltilmiş taban kullanılarak zeminden bu boşluk içine soğuk hava basılmalıdır. Soğuk hava koridoru olarak belirlenen noktalarda sunucular tarafından soğuk havanın emilimini sağlamak için Şekil 3.13'da olduğu gibi ızgaralar soğuk hava koridoruna konumlandırılmalıdır. Sunucu ve diğer cihazlar içinden geçen bu hava sıcak hava

koridoruna atılır. Isınan bu hava klima sistemi tarafından emilir ve tekrar soğutma işleminin yapıldığı noktaya iletilerek sistem odası içinde bir devri daim hava akımı sistemi oluşturulmuş olur.



Şekil 3.13. Zemin ızgarası

Şekil 3.14. Zemin ızgarası tersten görünümü

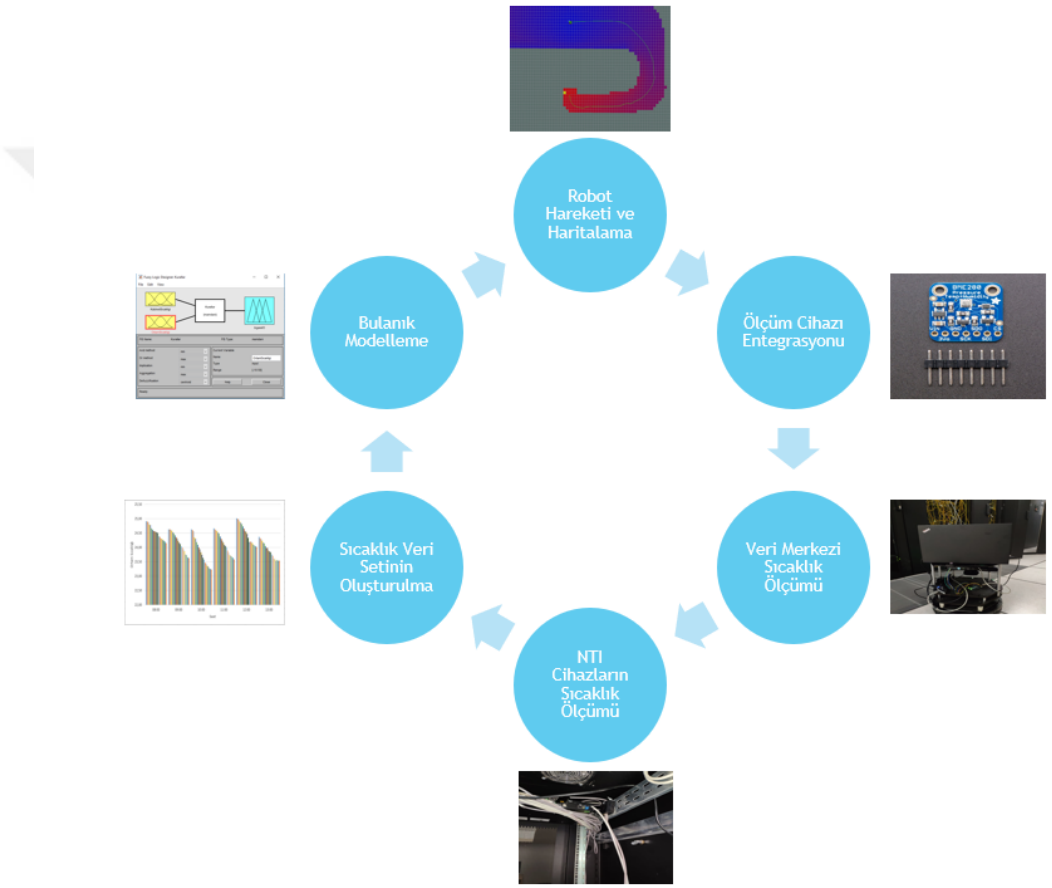
Sistem odasında verimli bir iklimlendirme yaparken yaşanacak bazı sıkıntılar aşağıda belirtilmiştir (Akın and Ketenci 2010).

- Sıcak ve soğuk hava akımının karışması: Sunucu ve diğer sistem odası elemanlarının içinden geçen havanın ısınarak yükselmesi. Bununlar birlikte sistem odası içinde bulunan klimaların ısınan havayı soğutması için sıcaklığın alçalması sonucunda CRAC (Computer Room Air Conditioner) soğutma sistemlerinin çiğ noktasının altında olması soğutma kapasitesini düşürüyor olmasına sebep olmaktadır.
- Sunucu ve diğer cihazlardan çıkan sıcak havanın cihaz tarafından tekrar emilmesi: Bu süreçte kabinet üs kısımlarında bulunan bulunan cihazların yeteri kadar soğumayan havanın ön kısımdan tekrar emilerek performanslarının düşük olması ve donanımsal arızalara sebep olması.
- Soğuk havanın istenilen düzeyde tüm kabinetlere ulaştırılmaması: Yükseltilmiş zemin altında klima tarafından basılan havanın kablolar vb gibi durumlardan dolayı engel teşkil etmesi.
- Uygun olarak konumlandırılmamış hassas klima sistemleri: yükseltilmiş taban içinde yol alan hava akımının yeterli basınca sahip olmalı ve optimum hızda sabitlenmelidir.
- Koridorlar arasında sıcak ve soğuk havanın karışmasına imkan verecek boşluklardan kaçınılmalıdır.

Bu çalışmada yapılan sıcaklık ölçümleri Atatürk Üniversitesi Bilgisayar Araştırma ve Uygulama Merkezi bünyesinde bulunan veri merkezinde gerçekleştirilmiştir. Bu veri merkezinde 12 adet kabinet, bu kabinetler içinde farklı marka ve modellerde sunucu, depolama üniteleri ve network cihazları mevcuttur. 4 adet hasas iklimlendirme için kullanılan klima sistemi mevcuttur. Sistem odasında bulunan bu klimalar kabinet içinde bulunan sunucu ve diğer sistem cihazlarının uygun koşullarda çalışması adına kesintisiz olarak çalışmaktadır.

3.2. Yöntem

Çalışmamızda farklı bileşenler bulunduğu için ve herbir bileşen bir sonraki için önem arz ettiğinden uygulama geliştirilirken basamak basamak ilerleme tercih edilmiştir. Bu süreç ve yapılan işlemler aşağıda Şekil 3.15'te belirtilmiştir.



Şekil 3.15. Çalışma model süreci

3.2.1. Robot işletim sistemi kurulumu

Ubuntu işletim sistemi versiyonlarından sadece 13.10 Saucy ve 14.04 Trusty versiyonları, ROS indigo'yu desteklemektedir (Open Source Robotics Foundation 2019). Turtlebot2 bileşenleri ROS indigo ile uyumlu bir biçimde çalışmaktadır. Robot üzerinde

uygulanacak olan komutların algılaması için ilk olarak ROS indigo versiyonu ubuntu işletim sistemi üzerine kurulmuştur.

Kurulum için aşağıdaki işlemler sırası ile uygulanmıştır (Open Source Robotics Foundation 2019):

- Kişisel bilgisayarımıza, packages.ROS.org sayfası üzerinden kurulum dosyalarının indirilmesi için gerekli ayarlar yapılandırılmalıdır.
- Ürün anahtarı ile kurulum etkinleştirilmelidir.
- Gerekli paketler indirilerek kurulum işlemi başlatılmalıdır. Erişilebilir paketleri bulmak için bir işlem komutu uygulanır.
- ROS kullanmadan önce ROSdep başlatılmalıdır. ROSdep sistem arası bağımlılıkların kolayca tesbiti ve bağların sağlanması için kullanılır. Derlemek istediğimiz kaynağın sorunsuzca yürütülmesini sağlar.
- Yeni bir komut satırı penceresi açıldığında ROS ortam değişiklikleri bash üzerine otomatik olarak eklenmesi için bir işlem komutu uygulanır.
- ROS paketleri için kullanılan birçok kaynağı tekbir komut ile indirebileceğimiz komut uygulanır.

Yukarıda açıklanmış olan işlemlerin uygulama komutları EK dizinde yer almaktadır.

ROS kurulumu yapıldıktan sonra Turtlebot 2 robotumuz haritalama ve yol planlama uygulamaları için hazır hale gelmiştir.

3.2.2. Temel ROS mesajları

- /scan (tip: sensor_msgs/LaserScan) : Lazer mesafe ölçüm veri seti bu mesaj içinde tutulmaktadır. Lazer mesafe ölçümü yapan bu sensör /hukuyo_node noduna bağlanır.
- /keybord (tip: sensor_msgs/keybord): Robotun klavye arabirimini kullanarak robot hareketi sağlanır. Butonlar, basılmadığı durumda 0, basıldığında ise 1 değerini alır.

- /cmd_vel (tip: geometry_msgs/Twist): Doğrusal ve açısal hareket ve hız verilerinin tamamı bu mesaj içinde bulunur. Bu verileri x,y,z düzlemsel eksene aktaran mesajdır.
- /map (tip: nav_msgs/OccupancyGrid): Haritalama algoritmalarının yayınlandığı mesajdır.
- /tf (tip: tf/tfMessage): ROS nodlarının tamamı tarafından mesajların frame_id'dir. Tüm bu nodelerın ilişkili yapısı mevcuttur. Düğümler arasında ana, çocuk düğüm ilişkisi mevcuttur. Bu bağlamda düğümler ilişkilendirilerek frame ağaç yapısı oluşturulur.
- /odom (tip: nav_msgs/Odometry): Robotun harita veya ortam içinde hangi pozisyonda bulunduğu bilgisini tutar. Bu işlem yapılırken "quaternion" biçiminde tutulur.
- /pose2D (tip: geometry_msgs/Pose2D): Bu mesaj 2 farklı lazer taraması yaparak bu taramaların eşleşmesi sonucu robotun pozisyon bilgisini bulmaya çalışmaktadır.

3.2.3. Navigasyon

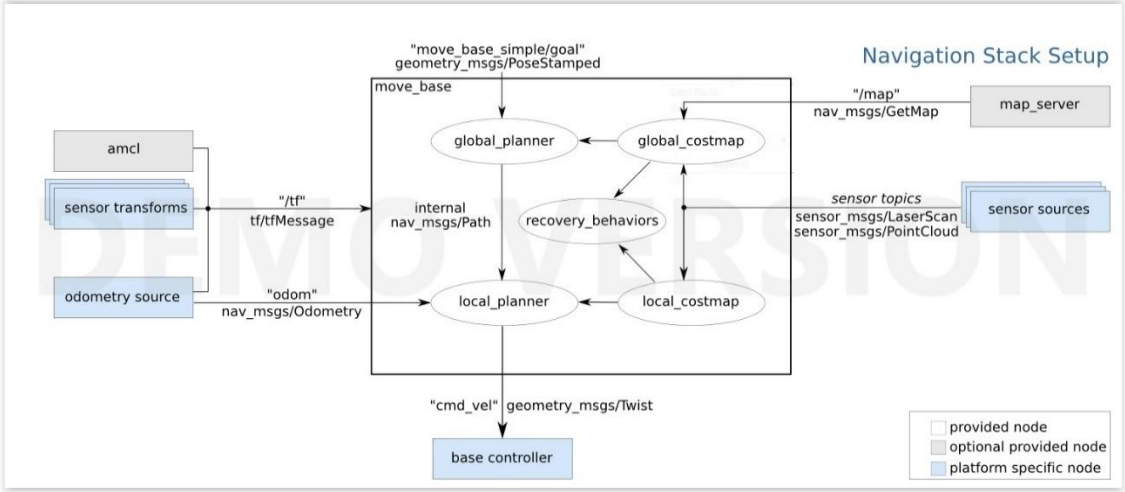
Çalışmanın bu bölümünde kurulumu yapılmış olan ROS platformu ile Turtlebot 2 mobil aracını bilinmeyen bir çevreye konumlandırılarak bu ortamda navigasyon işleminin yapılmasını sağlamaktır. Navigasyon işleminin yapılması için bilinmeyen ortam haritasının çıkarılması gerekir. Ortam haritası olmadan Turtlebot2 robotu navigasyon işlemini gerçekleştiremez. Robotun başlangıç noktası ile hareketi sonlandıracağı ortam içinde bulunan nesnelerin ve çevre düzeninin belli olduğu harita, robota yaptırabileceğimiz bütün eylemler için gereklilik arz etmektedir (Kağızman 2018).

Mobil robotun sadece harita bilgisine sahip olması navigasyon sürecinin tek başına başarılı bir biçimde sonuçlanacağı anlamına gelmez aynı zamanda robotun bulunduğu konum ve hangi yöne doğru işlemi gerçekleştirecek olması yapılması istenilen görevde başarılı olmasını sağlar. Bilinen bir harita üzerinde robotun konumunun tahmin edilmesi işlemine yerleştirme denilmektedir ayrıca bilinmeyen ortam haritasını çıkarmak ve bu harita üzerinde robotumuzun konumlandırlması için mesafe ölçer bir cihaz bulunmalıdır. Harita içine konumlandırılmış robotumuzun bu ortam içinde hangi yöne ve nasıl gideceğini belirleyen bir yapıya ihtiyaç vardır (Kağızman 2018).

Mobil robotlar başlangıç pozisyonundan hareket ederek başka bir yere doğru ve güvenilir bir biçimde gitmek için ROS navigasyon yığınını kullanırlar. Navigasyon yığını, konumundaki değişiklileri tahmin etmek için sönsörlerden gelen verileri kullanır ve çevre harita verileri ile bunları beraber işleyerek robotun hareketini sağlaması için uygun bir yol belirler (Zheng 2019). Yol planlama, Başlangıç noktası girdi olarak belirlenerek hedef noktaya ulaşması için kullanacağı en kısa ve uygun yolu çıktı olarak verir. Bu işlem esnasında sensörlerden ve haritadan gelen veriler kullanılarak mobil robotun yolu üzerindeki engellere çarpmadan navigasyon sürecini başarıyla gerçekleştirilir (Kağızman 2018).

Navigasyon işleminde robottan alınacak verimi üst sınırlara taşımak için yapılması gereken bazı ayarlar vardır ama bu ayarlar tahmin edildiği kadar kolay olmayabilir. Bu işlemler için "nasıl" ve "neden" kavramlarını biliyor olmak rasgele yapılacak denemelerin önüne geçerek zaman kaybını önleyecektir (Zheng 2019).

Çalışmanın bu kısmında yapılan navigasyon işlemleri basamak basamak belirtilerek ROS navigasyon paketleriyle birlikte süreç nasıl işliyor bahsedilecektir. Şekil 3.16'te navigasyon yığın akış şeması aşağıdaki gibidir.



Şekil 3.16. ROS navigasyon yığını

Yukarıdaki şekle göre renklendirilmiş işlemler bir anlam ifade eder. Beyaz kutu içine alınmış olan süreç robot işletim sisteminde mevcut olan ve temel olarak navigasyon işlemini gerçekleştiren işlemlerin bir araya geldiği bir yığını temsil eder (move_base). Mavi renkli şekilde belirtilen işlemler ise kullanılacak her bir farklı robot platformu için oluşturulmalıdır. Gri renk ile temsil edilen şekildeki işlemler ise bir sürece bağımlı çalışan işlemlerdir. Yukarıdaki şekilde gösterilen tüm bileşenlerin kısa açıklamaları aşağıda verilmiştir.

3.2.4. Navigasyon düğümü (Navigation stack)

Harita Sunucusu (Map Server): Harita verilerinin bir ROS servisi olarak ROS harita sunucusu yığınına verilmesini sağlar. Ayrıca harita kaydedici (map_saver) ile dinamik olarak haritadan gelen verileri dosyaya kaydetmeyi sağlar.

Sensör Kaynakları (Sensor Sources): Sensörlerden gelen veriler ortamda bulunan nesnelerin tespit edilmesi ve robotun harita üzerindeki konumunu belirlemek için kullanılır. ROS platformu üzerinde bulunan sensor_msgs/LaserScan veya sensor_msgs/PointCloud mesaj paketleri üzerinden sensör verileri alınarak işlenir.

Taban kontrolü (base controller): Move_base yığını yerine temel seviyede küresel ve lokal planlayıcıları birbirine bağlayarak gezinme görevini yerine getirmek için kullanılır.ROS platformunda bulunan geometry_msgs/Twist mesajından gönderilen verileri robotun hareketini sağlayacak bir biçimde motor hızına çevirmektedir.Hız komutlarını taban kontrolüne göndermek için bazı bileşenlere ihtiyaç duyulmaktadır. Bunlar; doğrudan motorlara kumanda edebilen temel hız control cihazı, temel hız sayacı (odometry) denetleyicisi, hız komutlarını görmek için daha yüksek seviyeli bir programlama (Harikrishnan 2016). Taban kontrolünden gelecek olan hız verileri Turtlebot 2'nin anlayacağı bir protokol işlemi ile motor komutlarına dönüşecektir.

Odometri kaynakları (Odometry sources): Robot bulunduğu ortam içinde konumunu tespit etmek için tf'i kullanır. Burada elde edilen sönsör verileri ile haritayı ilişkilendirir.

Robotun hareket hız ile ilgili bir veri sağlayamaz. Bu sebeple ROS üzerinde yayınlanan nav_msgs/Odometry mesajı üzerinden robotun hız bilgilerine erişebilir (Kağızman 2018). ROS navigasyon işlemi esnasında yerel planlayıcı odometri mesajını ("odom") alır ve çıktı olarak robotun hareketini sağlayan hız komutları (cmd_vel) olarak verir (Zheng 2019).

Sensör dönüşümleri (sensor transform): Robot, sensörlerden gelen dönüşümleri yayınlar ve diğer robot bileşenleri için ROS dönüşümlerini kullanır (Zheng 2019). Bu durumumda kordinat bilgisinin ilişkilendirlmesi için gerekli veriler yayınlanmalıdır. Böylece ROS dönüşümleri ile birlikte ana kordinat çerçevesi arasındaki ilişkilendirlme yapılmalıdır (Kağızman 2018).

Move_base paketi: Move base düğümü (node) move_base olarak isimlendirilen bir paketten meydana gelmektedir. Temel işlevi robot tarafından başlangıç noktası belli olan bir rota boyunca hedef konuma gelmektir (Joseph and Cacace 2018). 2B düzlemde oluşturulmuş harita üzerinde aracın belirlenen hedef noktaya erişmesi için rota çizilir. Rota takip esnasında engellerden kaçmak için rotayı takip eden robotun hız verisinin çıktı olarak elde edilmesi sağlanır. Robotun başlangıç konumu "dometry/filtered" başlığında, çıktı olarak ise "cmd_vel" içinde kullanılır (Kağızman 2018). Bu düğüm içinde bulunan küresel ve yerel planlayıcılar kendi aralarında iletişime geçerek hedef noktaya erişim için çeşitli yolların planlarının oluşturulması sağlanır. Küresel planlayıcı harita üzerinde uzak noktalara yol planlamasını oluştururken, yerel planlayıcı kısa mesafede engellerden kaçabilecek yol planlarının oluşturulmasını sağlar (Joseph and Cacace 2018). Yerel planlamada odometri mesajları (/odom) alınırken, robotun hareket hızı (/cmd_vel) çıktı olarak kullanılır. Robotun başlangıç noktasından hedefe noktaya gidebilmesi için dinamik pencere yaklaşımı yerel planlayıcı (DWA local planner) kullanılmıştır. Temel olarak robot hareketi ile bu hareketten doğan ivmelenme ve hızın kinematiğini anlamak için kullanılan bir pakettir (Kağızman 2018). Bu süreçte robot planlanmış yol üzerinde bir engel ile karşılaşırsa geri kazanım (rotate-recovery) paketi ile bağlantı sağlanarak yerel maliyet haritası (local costmap) ve küresel maliyet haritası (global costmap) ile beraber harita elde edilir. Böylece ortamda bulunan nesneler ile ilgili veriler elde edilir.

Move_base düğümü temel olarak geometry_msgs / PoseStamped mesaj iletilerini kullanır. Bu yapı aslında basit aktif sunucu (SimpleActionServer) uygulaması olarak adlandırılır. Basit aktif sunucu düğümüne hedef konumu gönderilir. Move_base düğümü, move_base_simple / goal ile adlandırılmış olan konu ile navigaston sürecine giriş yapar. Move_base düğümü hedef nokta olarak belirlenmiş konumu aldığında küresel planlayıcı, yerel planlayıcı, geri kazanım, yerel maliyet haritası (local costmap) ve küresel maliyet haritası (global costmap) gibi bileşenler ile bağlantı kurar. Bu süreç sonunda hız komutu mesajı yayınlayan geometry_msgs / Twist çıktı verisini oluşturur ve taban kontrolüne göndererek istenilen hedeflenen noktaya robot ulaşım sağlar (Joseph and Cacace 2018).

Bu çalışmada, Turtlebot 2 üzerinde bulunan kinect kamera laser tabanlı tarama yapmaktadır. AMCL (Adaptive Monte Carlo Localization) lazer tabanlı bir haritaya girer, lazer taraması yapar ve elde edilen mesaj veriler doğrultusunda çıkarım yaparak tahminlerde bulur. AMCL parametreleri parçacık filtresini (partical filter) başlatır. Başlangıçta başka bir parametre ayarlanmaz ise ilk durum parametre değerleri her zaman 0,0,0 olarak belirlenir. Robotun bu parametre değerleri ile harita içine yerleştirilmesini sağlayan AMCL, harita verilerini ve robot hareketlerini sağlamak amacı ile ROS servisi olarak çalışan map_server ile beraber lazer ve odometri çıktılarını kullanarak 2 boyutlu bir harita oluşturan Gmapping paketi kullanılmıştır. Robot üzerinde bulunan lazer tarayıcı vasıtası ile yayınlanan scan konusu kullanılarak bir harita çıkarılmasıdır. Elde edilen veri setinin LaserScan verisine dönüştürülmesiyle yatay scan çıktısı elde edilmelidir.

3.2.4.a. Hız ve ivmelenme (Velocity and accelartion)

Robutun hız ve ivme verileri, yerel planlayıcı ve yerel planlayıcı tarafından kullanılan dinamik pencere yaklaşımı (DWA) için temel verilerdir. Robot işletim sistemi navigasyon yığınının kullanmış olduğu yerel planlayıcı temel olarak odometri mesajlarını alır (/odom) ve çıktı olarak robotun hareketini sağlayan motor sensörlerine hız komutlarını (/cmd_vel) verir. Mobil taban, en yüksek ve düşük hız ile ivme parametreleri en temel veri olarak kullanır. Bu parametrelerin en uygun biçimde ayarlanması yerel

planlayıcının en doğru davranışı sergilemesi için gereklidir. Bu süreçte düzlemsel hareket ve ivme ile dönme hızı ve ivmesinin bilinmesi gereklidir.

Robotun en yüksek hıza çıkış değerini görmek için kullanım klavuzuna başvurulabilir. Turtlebot 2 robotumuz en yüksek hız olarak 0.65m/s sahiptir (Clearpath Robotics 2014). Hız kavram incelendiğinde, çevrimsel hız ve dönme hızı karşımıza çıkmaktadır. Çevrimsel hız, 3 boyutlu sistemde yatay ve dikey düzlem boyunca doğrusal bir biçimde hareket sağlar. Çevresel hız, robotun en yüksek hız değeri ile aynıdır ve metre/saniye cinsinden birimlendirilir. Dönme hızı, açısal olarak ifade edeceğimiz hız türüdür. Radyan/saniye cinsinden birimlendirilmiştir. Turtlebot 2 robotunun maksimum dönüş hızı 3.142 rad/s dir (Clearpath Robotics 2014). Maksimum çeviri ve dönme hızlarını, robotun sahip olduğu gerçek hızından düşük tutmak çalışmada daha güvenli sonuç vermesi için tercih edilmiştir.

Maksimum ivmeyi ölçmenin alternatif yolları mevcuttur. Zaman damgası ile oluşturulmuş odometri verileri elde edilir ve robotun sabit maksimum çeviri hızına ulaşmasının ne kadar sürdüğü görülür. Odomerti mesajlarından (nav msgs/Odometry message) alınan veriler ile pozisyon ve hız bilgileri kullanılarak ivmelenme hesabı yapılmıştır. Maksimum çevirme ve maksimum dönüş ivmelenmesi hesaplanma aşağıdaki (Zheng 2019), denklem 1 ve denklem 2'de görülmektedir.

$$a_{t\,,\,max} = \max dv / dt \approx v_{max} / t_t \tag{1}$$

$$a_{r\,,\,max} = \max dw / dt \approx v_{max} / t_r \tag{2}$$

Minumum hız paramatresinin ayarlanması yukarıda belirttildiği gibi belli bir formül ile yapılmaz. Robot kendini geri çekecek bir çalışma yapısına sahip olmasına rağmen işlemlerin çoğunu kendini ileri taşıyarak ve dönüş yaparak sağlamayı amaçlar. Mimumum dönme hızı maksimum hızın negatifi olarak ayarlanmaktadır ve böylece robot iki yönde hareketini sağlamaktadır. Ayrıca yerel planlayıcının kullandığı dinamik pencere yaklaşımında dönme hızının mutlak değeri alınır (Zheng 2019).

X hızı robotun bulunduğu düzlem üzerinde düz harekete paralel olan hızı anlamına gelmektedir ve çeviri hızı (translational velocity) ile aynıdır. Y hızı ise x hızına 90˚ lik bir açı ile dik yönde olan hızdır.

3.2.2.b. Küresel planlayıcı (global planner)

Yol planlama işleminin yapılması için 2 farklı yaklaşım kullanılmaktadır. Bunlar; küresel ve yerel planlamadır. Küresel planlayıcı bulunduğu ortam hakkında tüm bilgiye sahip olduğunu var sayar. Sahip olduğu tüm bu veri kümesiyle robotun bulunduğu çevrede nesneleri tanımlar ve robot tarafından engel olarak algılanacak bu nesnelerin pozisyonunu, boyutlarını, koordinatlarını ve şekillerinin tam olarak belirlenmesini sağlar. Ortam ile ilgili tüm verilerin elde bulunması ile küresel planlama algoritması çevrimdışı çalışır. Bu esnada hareket tam anlamıyla başlamasa bile başlangıç ve hedef nokta arasında yol planlanmış olur. Bu planlanmış yol bilgisi hareket esnasında referans olarak kullanılır (Buniyamin *et al.* 2011).

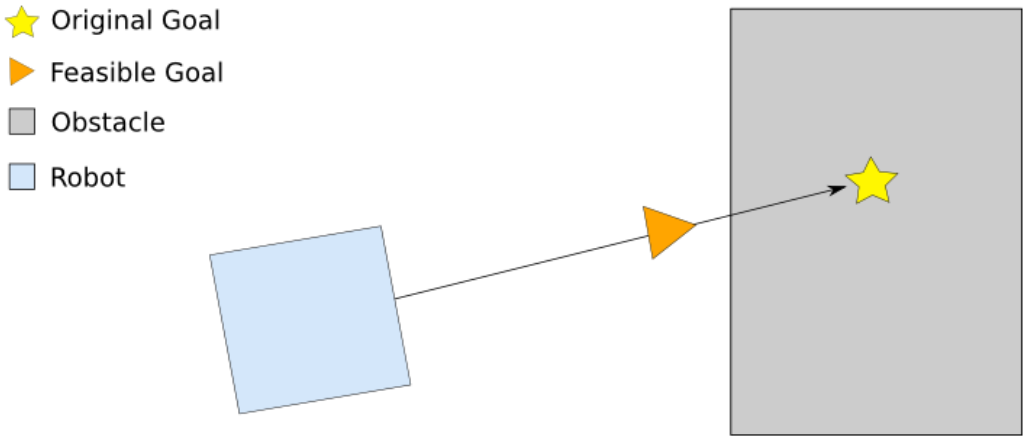
Küresel yol planlaması esnasında sonuç, daha uzun bir sürede elde edilir. Bunun temel sebebi, dinamik olarak yol planlaması yapılmıyor oluşudur. Ayrıca ortam ve ortamda bulunan nesnelerin belirlenmiş olması çizilen yolun daha kaliteli olarak belirlenmesini sağlamıştır (Suvaydan 2011).

Küresel yol planlamada ortaya çıkan bazı noktalar aşağıda belirtilmiştir.

- Daha uzun bir sürede tamamlanan bir süreç
- Tepkisel olarak daha yavaş
- Harita tabanlı çalışma mantığı
- Çevre ve çalışma ortamı hakkında bütün verilere sahip olma
- Hedef noktaya varmak için oluşturulan uygun bir yol planı
- Görüşmeye dayalı, koordineli çalışan bir sistem.

Küresel ve yerel planlayıcı olarak kullanacağımız ROS üzerinde, Nav_core:: BaseGlobalPlanner paketi içinde bulunan 3 adet planlayıcı mevcuttur. Bunlar; corrot_planner, navfn ve global_planner'dır.

Carrot_planner : Şekil 3.17'de gözüktüğü üzere küresel planlayıcılar arasında en basit olanıdır. Planlayıcı gideceği hedef noktasını kullanıcıdan alır. Alınan bu hedefin üzerinde bir engel olup olmadığını kontrol eder ve hedef nokta bir engel üzerinde konumlandırılmış ise robot, bir engelle karşılaşıncaya kadar hedef noktaya gitmek isteyecektir. Daha sonraki süreçte hedef noktayı bir yerel planlayıcıya devreder. Bu şekilde robot hedef nokta erişilmez olsa bile mümkün olduğu kadar hedef noktaya yakın bir yere konumlandırma yapacaktır.



Şekil 3.17. Küresel yol planlama (Open Source Robotics Foundation 2018)

Navfn ve global_planner : Izgara (grid) tabanlı bir navigasyon fonksiyonu olarak kullanılan navfn ve global_planner, küresel bir planlama algoritmasında kullanılan verimli ve doğal bir işleve sahiptir. Küresel dinamik pencere yaklaşımı sabit engellerden kaçmak için küresel ve yerel planlama fonksiyonlarını bir araya getirir. Bu işlem, hedef noktaya dalga yayım tekniği kullanılarak hesaplama yapılmasını sağlar. Klasik hareket planlama algoritmaları ile çalıştırılan navigasyon işleminde engelleri aşmada oluşturulan yörüngenin tüm kullanılan ızgaraları planlama sürecinde, başarılı olmaması sebebiyle dezavantaja sahip olur. Dinamik pencere yaklaşımı bu sorunu ortadan kaldırır ve seçilen

hareket komutlarıyla engellerden kaçmak için oluşturulan yörünge boyunca engeller ile arasında mesafeyi korumaktadır. Küresel dinamik pencere fonksiyonu hareket komutu seçildiğinde navfn yeniden hesaplar ve bilinmeyen ve değişken ortamda çalışmasına izin verir (Brock and Khatib 1999).

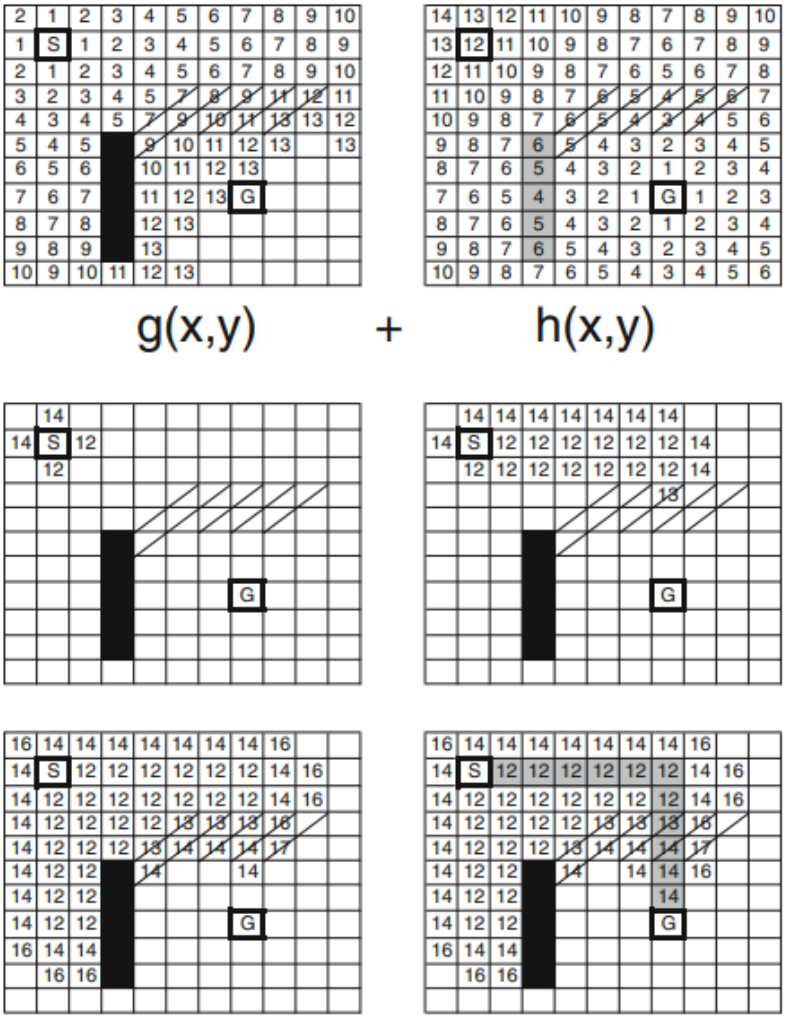
Navfn, başlangıç ve hedef noktası arasındaki küresel yolu hesaplarken en düşük maliyetle hesaplamak için Dijkstra's algoritmasını kullanılır. Bu çalışmada en kısa yol planlaması yapılırken Dijkstra's algoritması kullanılmıştır. Global_planner ise navfn'e oranla daha çok gelişmiş seçenekler ve daha esnek bir yapı kullanmaktadır. Bu seçenekler; A* algoritması için destek, geçişli ikinci dereceden yaklaşım ve geçişli ızgara yoludur (Zheng 2019).

A* algoritması : İlk olarak 1968 yılında kullanılmaya başlanan bu algoritma, yapılan çalışmalarda kullanılarak ve bu süreç boyunca kullanıma bağlı olarak çeşitli iyileştirmelerden geçmiştir (Dunn 2015).

Günümüzde yapılan birçok uygulamada, başlangıç noktası belli olan robotun engellere çarpmadan hedef noktaya en kısa bir biçimde varmış olması sağlanmalıdır. A* algoritması bunlardan birdir ve en kısa yolu bulma problemini sayısal olarak çözmeyi amaçlamıştır. Bu algoritma başlangıç noktasından hedef noktaya en az maliyetli yolu belirler. Doğrudan doğruya hedef düğüm için bir maliyet hesabı yapılır (Kshirsagar and Shukla 2010). A* algoritması yapı olarak sezgisel bir algoritmadır. Bunun temel sebebi ise iki nokta arasındaki mesafe hesaplamada belirlenen fonksiyondan gelmektedir.

$$f(n) = g(n) + h(n) \tag{3}$$

denklemde f(n) hesaplama yapan sezgisel bir fonksiyondur, g(n) başlangıç noktasından bitiş noktasına kadar yapılan yolun maliyet hesabıdır ve h(n) hareket esnasındaki mevcut konum ile hedef nokta varmak için yapılan tahmini bir mesafedir. Bu süreçte tahmin yoluyla hedefe varıyor olmak sezgisel bir fonksiyon olduğunu göstermektedir (Şeker 2009). Şekil 3.18'de A* algoritmasının sezgisel çalışma yapısı görülmektedir.



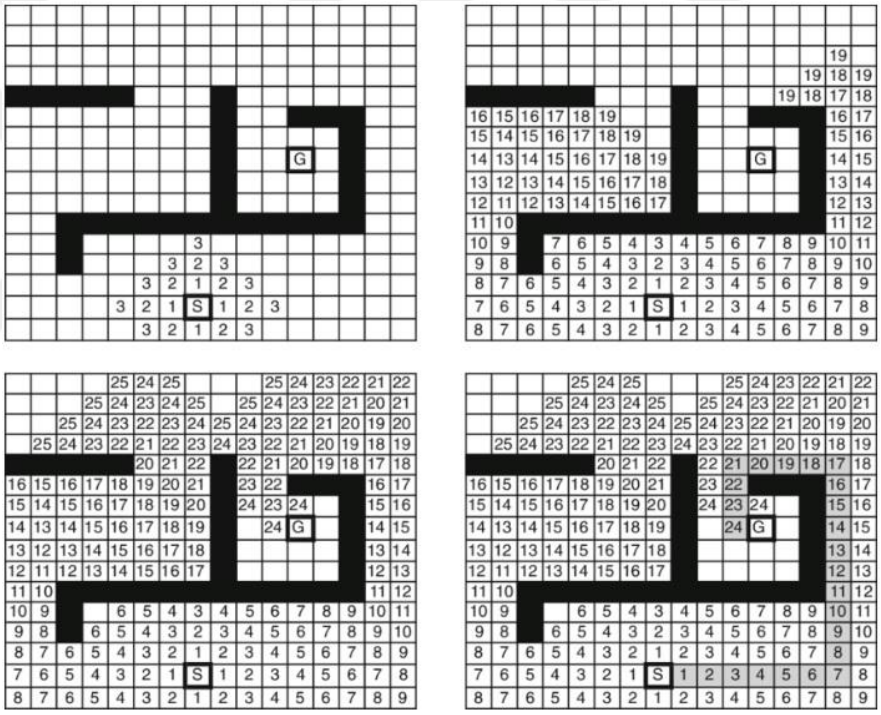
Şekil 3.18. A* algoritması örnek görünümü (Ben-Ari and Mondada 2017)

Algoritmanın çalışma prensibi aşağıdaki gibi tanımlanabilir.

- Her basamakta en düşük değerli düğümü alır ve bu düğümü kuyruktan çıkarır.
- Gidilen bu düğüm esas alınarak bu düğüme komşu olan tüm düğümlerin maliyeti güncellenir ve bu değer f(n) hesap yapılan sezgisel fonksiyon içinde tutulur.
- Hedef düğüme erişinceye kadar (önceliği en öne gelene kadar) veya kuyrukta hiçbir düğüm kalmayana kadar bu işlem tekrar eder.

Dijkstra's algoritması : Dijkstra algoritması 1956 yılında Alman bilgisayar bilimci Edsger Dijkstra tarafından tasarlandı. En kısa yol bulma algoritmaları arasında en çok kullanılan algoritmalardan biridir.

Şekil 3 .19'da görüldüğü gibi bu algoritma daha çok yönlerdirme işlemlerinde ve grafik tabanlı algoritmalarda en kısa yolu bulmak için kullanılır. Çalışmada robot doğrudan ızgaralara ayrılmış düzlem üzerinde başlangıç noktasında hedef noktaya kadar maliyet hesaplaması yaparak gidebileceği en kısa yolu oluşturmayı amaçlamıştır.



Şekil 3.19. Dijkstra's algoritması komşu ilişkisi (Ben-Ari and Mondada 2017)

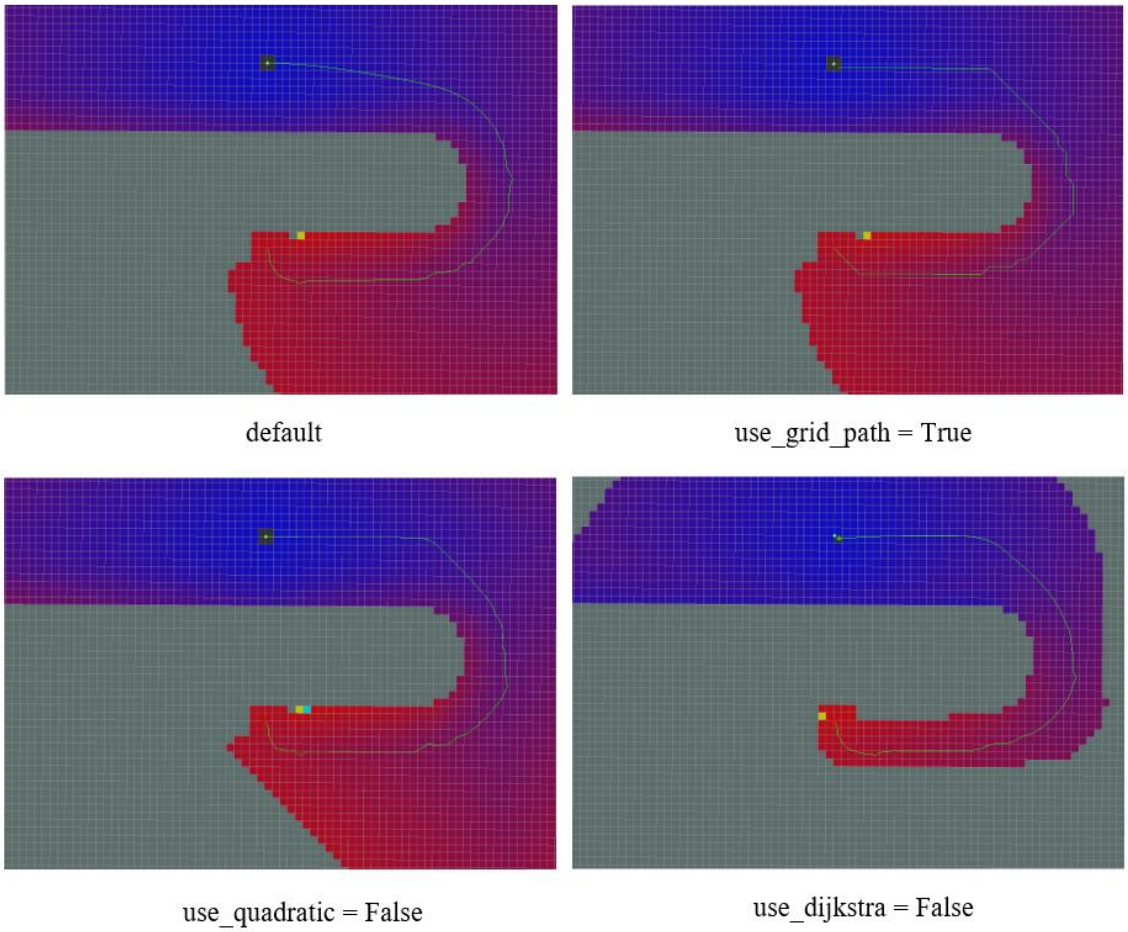
Algoritmanın çalışma prensibi aşağıdaki gibi tanımlanabilir.

- Başlangıç noktası, kaynak düğüm olarak seçilir ve maliyet değeri 0 olarak belirlenir.
- Başlangıç noktasındaki düğüme komsu olan tüm komşu düğümler kontrol edilir.
- Her komşu için kümülatif düğüm maliyetleri hesaplanır ve geçici olarak tutulur.
- Geçici düğümler kontrol edilir.

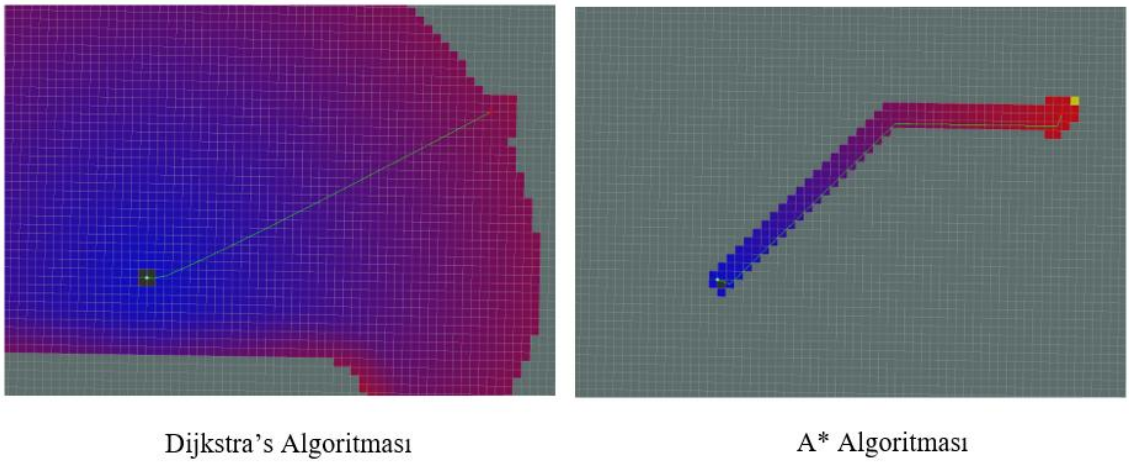
  - Kümülatif maliyeti en düşük olan düğüm seçilir ve kalıcı düğüm olarak tutulur. Bu düğüm son maliyet olarak düşünülür.
  - Eğer düğüme ulaşmak için birden çok düğüm kullanılıyorsa maliyet olarak en düşük yol seçilir.
- Tüm düğümleri kalıcı hale getirmek için 2,3,4 adımları tekrarlanır ve en kısa yol hesaplanmış olur.

Çalışmada global_planner paketi kullanımı tercih edilmiştir. Bu paketin temel parametreleri mevcuttur ve Şekil 3.20, 3.21 ve 3.22'de parametreler ve etkileri belirtilmiştir. Global_planner paketi navigasyon uygulamaları için hızlı çözüm sunar. Nav_core sınıfı içinde bulunan nav_core:: BaseGlobalPlanner paketi arabirimine uyar. Navfn paketine oranla daha esnek bir yapıya sahiptir. Global_planner paketi parametrelerini görmek ROS içinde bulunan rqt dinamik ayarlar programı kullanılır. Çalıştırmak için ROSrun rqt_reconfigure rqt_reconfigure komutu yazılır. Parametreler ve başlangıç değerleri aşağıdaki gibidir.

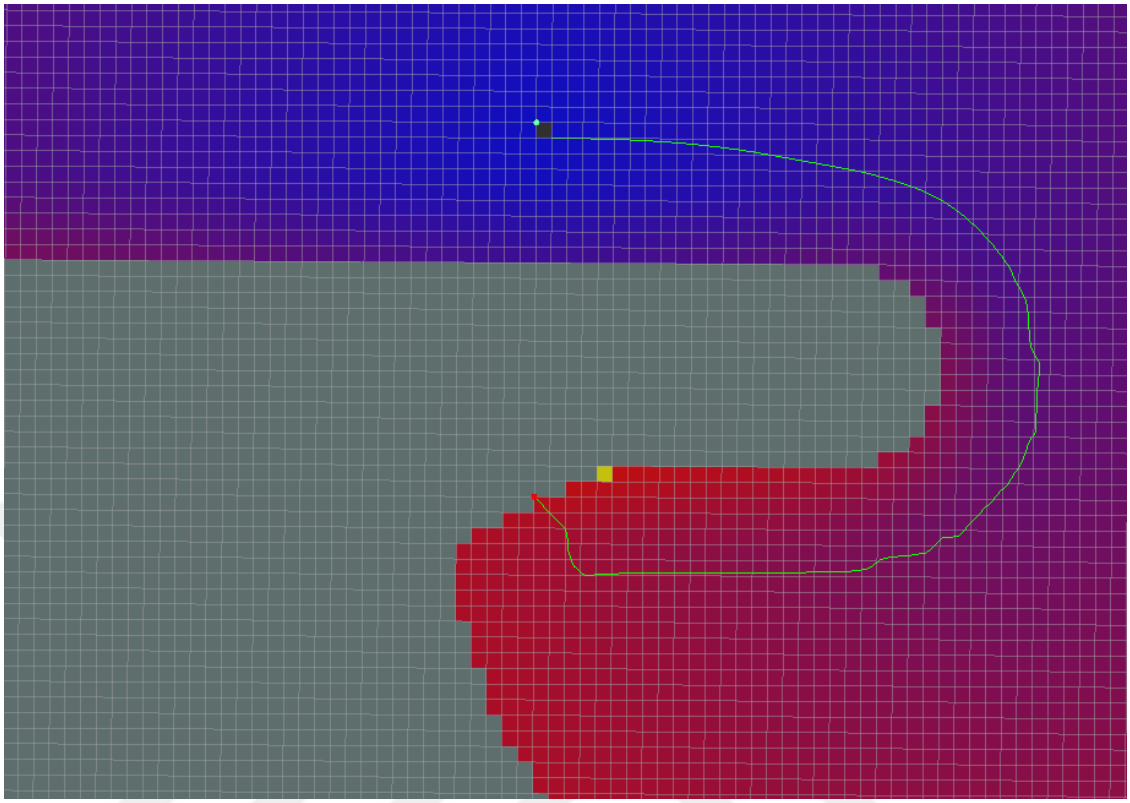
- Allow_unknown – doğru (true)
- Use_dijkstra – doğru (true)
- Use_quadratic – doğru (true)
- Use_grid_path – doğru (true)
- Old_navfn_behavior – yanlış (false)
- Visualize_potential – yanlış (false), visualize_potential parametresini kullanmak Rviz'de haritayı görselleştirmek için yardımcı olacaktır.



Şekil 3.20. Global planner parametreri (Open Source Robotocs Foundation 2018)



Şekil 3.21. Dijkstra's algoritması ve A* algoritması görünümü (Open Source Robotocs Foundation 2018)



Şekil 3.22. Visualize potential parametresinin yanlış olma durumu (Open Source Robotocs Foundation 2018)

3.2.2.c. Yerel planlayıcı (local planner)

Yol planlama işleminin yapılması için 2 farklı yaklaşım kullanılmaktadır. Bunlar; küresel ve yerel planlamadır. Yukarıda küresel planlama süreci ile ilgili işlemler anlatılmıştır.Yerel planlama ise yol planlaması için kullanılan veri kümesininin yerel kaynaklar ile gerçek zamanlı yapılmasıdır. Yerel planlama esnasında elde olan tek veri seti robotun başlangıç bilgisi koordinatlarıdır, hedef nokta ve ortam hakkında hiçbir bilgiye sahip değildir. Yol planlaması esnasında rota planlaması yapılır. Bu süreçte rota üzerinde bulunan engellere tepki vererek engellerden kaçmak için planlama yapılır. (Buniyamin *et al.* 2011).

Yerel planlamanın genel özelliklerinden bahsedecek olursak (Suvaydan 2011);

- Tepkisel bir yapıya sahip olmasıyla hızlı bir sürede işlem tamamlanır.
- Ortam içinde konumlanmış robota yakın nesneler tanımlanır, en hızlı biçimde engeller analiz edilerek planlama yapılır ve hedef noktaya ulaşım sağlanır.
- Hızlı bir biçimde rota planlaması yapılır.
- Sensör tabanlı bir sistemdir.
- Ortamdan gelen verilerin tamamlanmamış bilgi olduğunu kabul eder.

Çalışmada lokal planlama yapılırken nav_core::BaseLocalPlanner paketi içinde bulunan dwa_local_planner kullanılmıştır.

Dinamik pencere yaklaşımı (DWA): Dinamik pencere yaklaşımı, kinematik ve dinamik kısıtlamalar dikkate alınarak eş zamanlı hareket kabiliyetine sahip hareket motorlarının engellerden kaçmasını sağlamak için kullanılan bir yöntemdir. Kinematik kısıtlamalar göz önünde bulundurularak doğrudan doğruya robotun hız uzayında arama yapılır. Bu arama alanı direk olarak robotun eriştiği doğrusal (v) hız ve dönüş (w) hızının bir arada tutulduğu çevirme ve dönüş (v,w) hızlarının veri kümesidir (Dieter *et al.* 1997). Düzlem üzerinde sensörlerden gelen veri kümesi hesaplanarak ortamdaki engellerden kaçınmak için kullanılır. DWA ilk olarak robotun düzlem üzerindeki doğrusal ve açısal hızlarını kullanarak belirli bir zaman aralığında doğrusal ve açısal ivmelenmesini hesaplanarak hız dönüşüm verileri elde edilir.

Tüm bu hız kümeleri arasında, robotun mevcut pozisyon bilgisi, var olan hızı ve robotun ivmelenme özellikleri göz önüne alır ve robotun herhangi bir engel olarak tanımlanan nesneye çarpmadan önce durmasına imkan sağlamayan veriler elenerek durmasını sağlayan veriler elde tutulur (Brock and Khatib 1999). Bu hızlara kabul edilebilir hızlar denir. Arama alanında budama işleminin yapılması algoritmanın ilk basamağıdır. İkinci adımda, hızı maksimize eden amaç fonksiyonu kalan hızlar içinden seçilen hız çiftelerini işler. Bu işleyiş sürecini ele alacak olursak, bunlar tarama uzayı ve optimizasyondur.

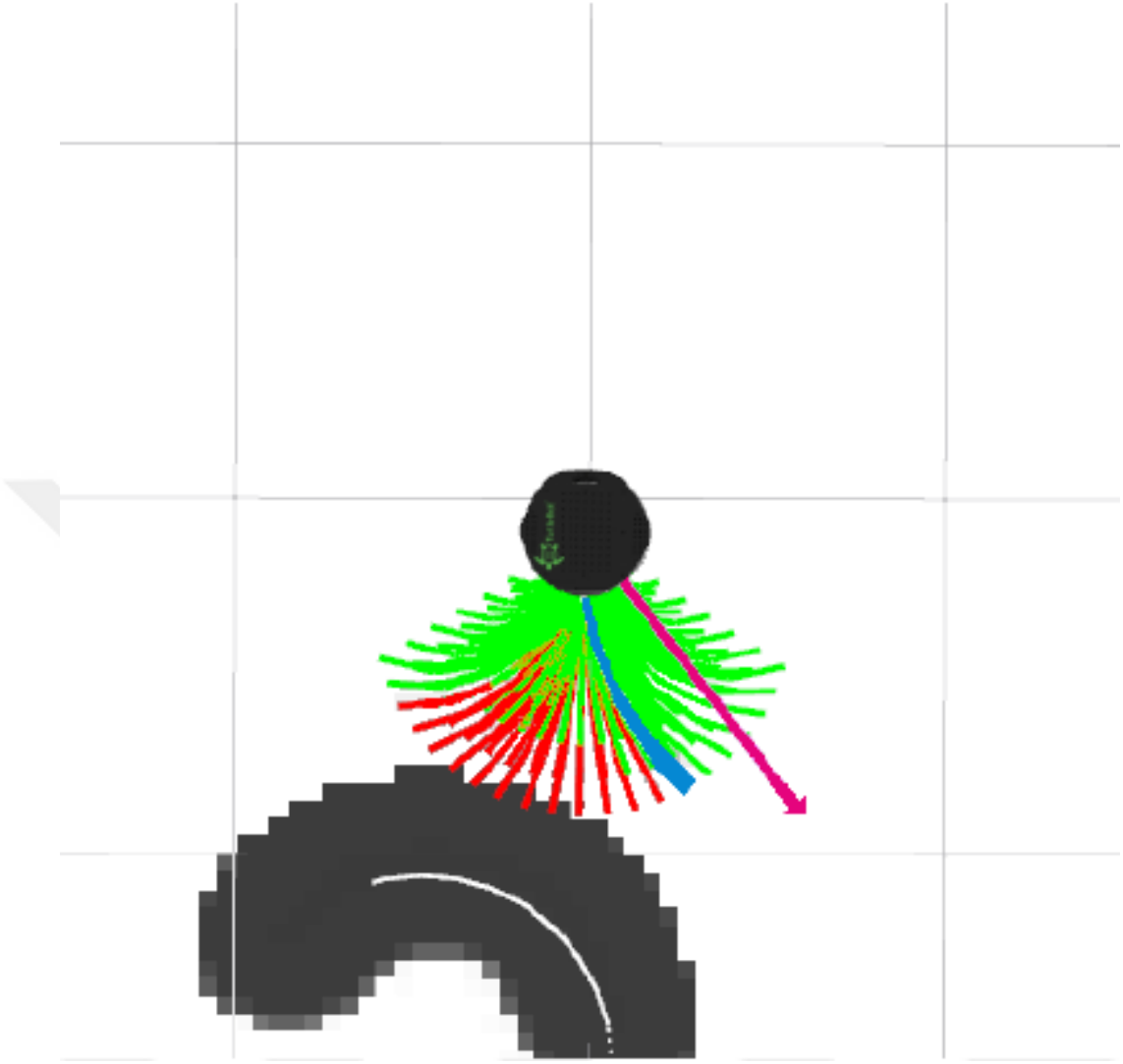
Tarama uzayı olası hızlarının azaltılması 3 aşamada gerçekleşir. Dairesel yörüngelerde, sadece çevirme ve dönme (v,w) hızlarının veri kümesi tarafından belirlenen benzer

olmayan yörüngeler dikkate alınır. Sonuç olarak iki boyutlu hız tarama uzayı elde edilir. Kabul edilir hız ile sadece robot için güvenli rotaları dikkate alır, Robot çarpacağı nesneye yaklaşmadan önce durabilmesini sağlayan hız (V,w) değerlerini elde tutar. Dinamik pencere ile robotun belirli sınırlar içinde tutulan ivmelenme göze alınarak çok kısa sürede istenilen hızlara ulaşmayı sağlar.

Optimizasyon ile hızların en üst seviyeye ulaşmasını sağlamaktır. Tarama uzayından gelen hız çiftleri (V,w) denklem 4'de görülen amaç fonksiyonu tarafından hesaplanır.

$$G(V,w) = \sigma[\alpha * yön(V,w) + \beta\_mesafe(V,w) + \_hız(V,w)] \quad (4)$$

Hedef yönü, hefen noktasına robotun hareket etmesinin ölçütlendirilmesidir. Robotun doğrudan hedef noktaya hareket etmesi en üst seviyededir. Mesafe, farklı yörüngeler arasında engelin hareketini engellemeyecek en uygun pozisyonun belirlendiği rotadır. Hız ise robotu desteklenen düzeyde ilerleme hızıdır. Bu fonksiyon düzlem üzerindeki üç bileşenin ağırlıklı toplamıdır. Böylece engelden kaçmak için gerekli boşluğu sağlar. Fonksiyonların [0:1] değerleri arasında kalması için düzleştirme işlemi (σ) kullanılır. Yukarıda uygulanan tüm bu işlemler sonucu dinamik pencere yaklaşımının robot üzerindeki etkisi Şekil 3. 22'de görülmektedir.



Şekil 3.22. Rota Belirlenmesi

Şekil 3.22'de hedef nokta rotası (pembe), uygun olarak kullanılabilecek hız kümesi verileri (yeşil), uygun olmadığı belli olan çevrimsel ve dönüşsel hız çifti (kırmızı), dinamik pencere yaklaşımı ile hesaplanan en uygun hız verisi (mavi) ile gösterilmiştir (Özdemir 2016).

Ayrıca DWA planlayıcısı kendisine engel olarak belirlenen nesnenin bilgisini sağlayan maliyet haritasına (costmap) doğrudan bağlıdır. Bu sebepten maliyet haritasının hesaplanmasında kullanılan bütün parametrelerin ayarlanması, DWA yerel planlayıcının en uygun biçimde çalışması için büyük önem taşır.

Bu parametrelerden ilk olanı ileri simülasyondur (forward simulation). Yerel planlayıcı robotun tarama uzayındaki hız çiftlerinin değerlerini alır ve bu hız verilerini temsil eden yörüngeler incelenir ve engelle kesişen daireler çarpmaya sebep olacağından elenir. Hız değerleri sim_time(s) parametresi tarafından kontrol edilir (Zheng 2019). Temel amacı elde tutulan hız verilerinin hareketi için izin verilen bir süreymiş gibi düşünülebilir. Bu parametre değerinin büyük olması robotun hesaplama sürecini uzatatır ve tahmin edilen rota daha uzun sürede çizilir.

Diğer bir parametremiz ise yörünge skorlamasıdır (trajectory scoring). En uygun hız çiftlerini bulmak için amaç fonksiyonu çalışır. Bu fonksiyonun değişkenleri üç bileşene dayanmaktadır. Engellerden uzak durma, hedef noktaya ulaşma süreci ve ileriye götürecek hızdır. Robot işletim sisteminde (ROS) hesaplama denklem 5'de görüldüğü gibidir.

Maliyet = [yol_mesafe_öngörüsü * (yörüngenin bitiş noktasından yola uzaklığı) + hedef_mesafe_öngörüsü * (yörüngenin yerel hedeften bitiş noktasına olan uzaklığı) + occdist_ölçeği * (yörünge boyunca maksimum engel maliyeti)] (5)

Yapılan bu hesaplama ile birlikte en düşük maliyet belirlenmiş olur. Yol_mesafe_öngörüsü (path_distance_bias) yerel planlayıcının küresel planlayıcıya ne kadar yakın olması gerektiği ağırlığıdır. Hedef_mesafe_öngörüsü (goal_distance_bias) belirlenen yörüngelerden hangisi olursa olsun robotun yerel hedefe ulaşma girişim ağırlığıdır. Occdist_scale ise engellerden kaçması için gereken ağırlıktır.

Son iki parametre ise hedef nokta mesafe toleransı (goal distance tolerance) ve salınım sıfırlamadır (oscilation reset).

3.2.2.d. Harita maliyeti (costmap)

Navigasyon sürecinde robotun hareket bilgileri, engellerden kaçmak için elde edilen hız veri setlerinden en uygun olanını seçme işlemleri yapılır. Harita maliyeti (costmap), engeller ile ilgili veri setini elinde bulunduran yapıdır (Çakmak 2014). İki tür harita maliyeti hesaplanır. Tüm ortam içinde bulunan nesneler haritasını çıkaran küresel harita maliyeti (global_costmap) diğeri ise, küresel harita maliyeti ile belirlenmiş engellerin tespit edemediği ve harita üzerinde belli olmayan engellerin yerel maliyet hesabıdır (local costmap).Ayrıca robotun sensörlerinden gelen veriler gerçek zamanlı olarak işlenir. Maliyet hesabı yapılırken kullanılan her iki yöntemde renk, engel boyutu ve eşik değeri gibi temel yapısal ayarlar yapılabileceği gibi farklı yapısal ayarların yapılmasını sağlayacak dosyalar da mevcuttur. Bu dosyalar base_local_planner_params.yaml, costmap_common_params.yaml, global_costmap.yaml ve local_costmap_yaml dosyalarıdır (Goebel 2012).

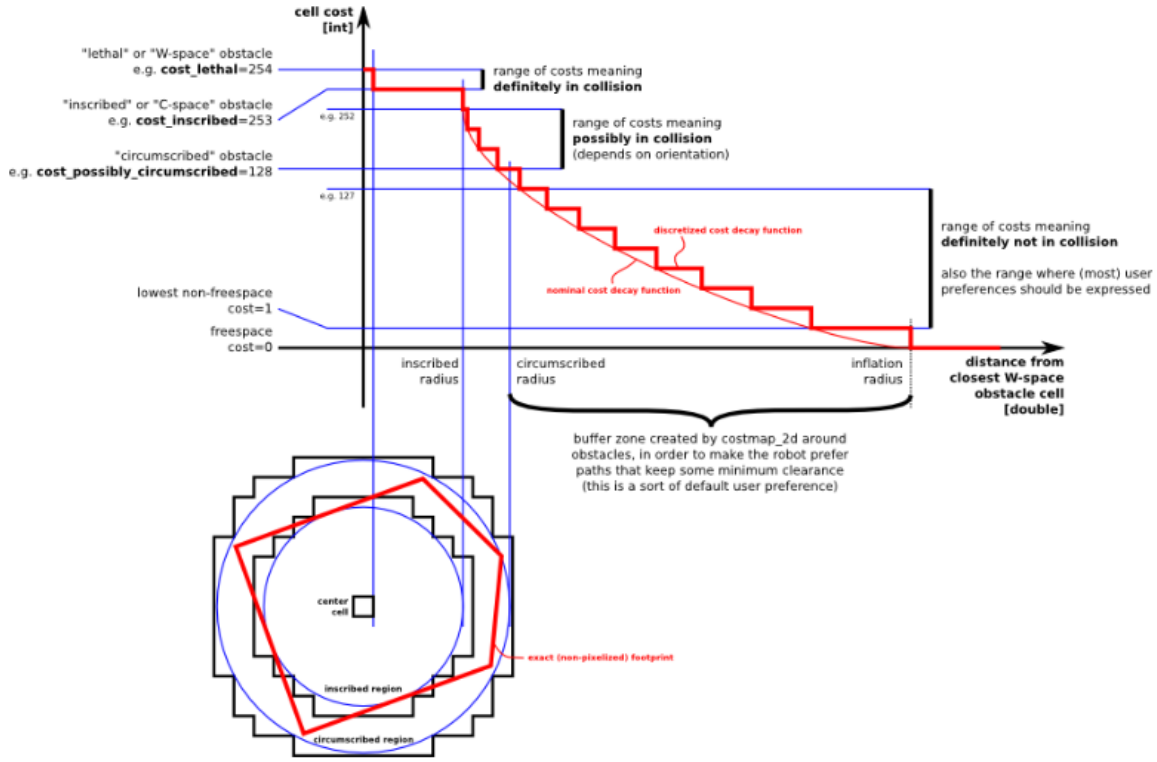
Küresel ve yerel maliyet haritası temel yapısal ayar parametrelerini inceleyecek olursak; obstacle_range parametresi maliyet hesabında kullanılan engellerin eşik değerlerini belirlemek için kullanılır. Maliyet haritası çizilirken engel olabilmesi mümkün cisimlerin tespiti için mesafe sensöründen verileri alır. Raytracle_range, sensör verilerinin okunması için önünde bırakması gereken mesafeyi belirtmektedir. Ayak izi (footprint), kullanılan robot fiziksel olarak dairesel ise robotun sınırlandırılmış yarıçapı ve ayak iz değerlerini hesaplar. ROS'ta ayak izi parametresi denklem 6'da görüldüğü gibi iki boyutlu dizi biçimindedir ve ilk kordinatı tekrarlamaya ihtiyaç duymaz. Robotun merkezinin 0.0, 0.0 değeri olduğu varsayılır ve robot dönüş hareketleri hem saat yönünde hem de saat yönünün tersine dönüşleri destekler.

$$[[x0,y0],[x1,y1],[x2,y2],\ldots.[xn,yn]] \quad (6)$$

Observation_sources parametresi, düzlem üzerinde bulunan boşluklar ile ayrılan maliyet haritasına iletilecek olan sönsör bilgilerinin bir listesini verir. Herbir sensör bir sonraki satırda tanımlanmıştır. Örnek vermek gerekirse observation_source: laser_scan_sensor

point_cloud_sensor burada parametre olarak belirlenen laser scan_sensor verileri gözlem kaynağına parametre olarak tanımlanmıştır. Aynı zamanda gözlem kaynağı için parametre olarak kullanılan laser_scan_sensor'de kendi parametre değerlerine sahiptir. laser_scan_sensor: { sensor_frame: frame_name, data_type: LaserScan, topic: topic_name, marking: true, cleaning: true } (Gill 2018).

Enflasyon parametresinden bahsedecek olursak, bu katman 0 ile 255 arası değer alan hücrelerden meydana gelmektedir. Enflasyon hesabı yapılırken kullanılan parametreler; enflasyon yarıçapı ve maliyet ölçütlendirme faktörüdür. Enflasyon yapıçapı başlangıç noktası yani sıfır maliyet noktasının ne kadar uzakta olduğunu kontrol eder. Maliyet ölçütlendirme faktörü enflasyon hücreleri arasında ters orantı mevcuttur. Bu parametrenin daha büyük değerler alması çürüme eğrisini Şekil 3.23'te gösterildiği gibi daha dik bir değer noktasına taşır.



Şekil 3.23. Enflasyon çürüme eğrisi (Open Source Robotics Foundation 2018).

Yukarıda bahsedildiği üzere maliyet haritası çıkarılırken parametrelerin en uygun biçimde ayarlanması, sadece dinamik pencere yaklaşımı yönteminin uygulanması için değil aynı zamanda yerel planlayıcının başarılı bir biçimde sonuç vermesi adına çok önemlidir. ROS'ta harita maliyet hesaplanırken 3 katmana ayrılır. Bunlar; sabit harita katmanı (static map layer), engel harita katmanı (obstacle map layer), enflasyon katmanıdır (inflation layer) (Zheng 2019).

Sabit harita katmanı, navigasyon yığınına verilen sabit SLAM haritasını doğrudan yorumlama yeteneğine sahiptir. Engel harita katmanı, iki ve üç boyutlu (voxel layer) engelleri içerir. Enflasyon katmanı, iki boyutlu maliyet haritası hücresi için şişirilen engellerin hesaplandığı kısımdır.

Küresel maliyet hesaplama: /map mesaj iletisi yapan "map_server" küresel maliyet haritasının hesaplanmasında kullanılır. Küresel maliyet hesaplamaya ait paremetreler belirlenir. Global_costmap.params.yaml yapılandırma dosyasında kullanılacak parametreler şu kelide belirtilmiştir; global_frame parametresi, referans gösterilmesi gereken koordinat çerçeversini belirlemek için kullanılır, update_frequency, hz cinsinden maliyet haritasının anlık olarak çalışma hızını temsil eder ve hz cinsinden değer aralığı belirlenir ve static_map parametresi, maliyet haritası map_server tarafından belirlenmiş bir haritanın maliyet haritası ile birlikte başlatılıp başlatılmayacağını belirlenir (Gill 2018). Yapılandırma dosyasındaki parametre değişkenleri ayarlandığında robot için sınır olacak duvarlar ve ortam içindeki nesneler tespit edilip çizilmiş olur.

3.2.4. AMCL (Adaptive monte carlo localization)

AMCL robotun bilinen 2 boyutlu bir harita içinde olasıklıksal olarak yerleştirme yapılmasını sağlayan bir ROS paketidir (Zheng 2019). Bilinen haritaya karşı konumunu izlemek için parçacık filtresini kullanır. Her örnek robotun pozisyon bilgisini temsil eden bir konum ve yönlerdirme verisini depolar. Burada bulunanan mevcut parçacıklar başlangıçta rastgele örneklenir. Robot hareket ettikçe parçacıklar mevcut durumu göz

önüne alır ve bayesian tahmin metodunu kullanarak robotun hareketine göre yeniden örneklendirme işlemini yapar (Macenski 2018).

Her defasında parçacıkların yeniden oluşturulması sürecinde rastgele belirlenen parçacıklar da yeni oluşum kümesine dahil olmaktadır. Bu esnada ortaya çıkan iki temel konu mevcuttur. İlk olarak her defasında bir araya gelen parçacık kümesine eklenen rasgele parçacık sayısının tespiti ve diğer konu ise hangi model kullanılarak rasgele parçacıkların dağıtılması gerektiğidir (Çakmak 2014). Rastgele parçacık sayısının belli olması çözüm olacağı gibi kestirme performansı göz önüne alınarak rastgele parçacıkların veri kümesine eklenmesi çözüm odaklı bir yaklaşımdır. Bu parçacıklardan önem faktörü kullanılarak bu soruna çözüm bulma amaçlanmıştır. Rastgele parçacıkların dağıtılması için kullanılacak duyarga ve poz uzayı ile ağırlığı veri kümesini kullanan tekdüze (uniform) dağılımları mevcuttur. Bu anlamda AMCL algoritma yapısı şekil 3.24'de belirtildiği gibidir.

Algoritma: AMCL

Girdi: X_{t-1}, u_t, z_t, m

$static\ w_{slow}, w_{fast}$
$X_t = X'_t = 0$
$for\ m = 1\ to\ M\ do$
  $x_t^{[m]} = hareket\_modeli(u_t, x_{t-1}^{[m]})$
  $w_t^{[m]} = olcum\_modeli(z_t, x_t^{[m]}, m)$
  $X'_t = X'_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
  $w_{ort} = w_{ort} + \frac{1}{M} x_t^{[m]}$
$endfor$
$w_{slow} = w_{slow} + a_{slow}(w_{ort} - w_{slow})$
$w_{fast} = w_{fast} + a_{fast}(w_{ort} - w_{fast})$
$for\ m = 1\ to\ M\ do$
  $do\ \max\left\{0.0, 1.0 - \frac{w_{fast}}{w_{slow}}\right\}\ olasılığıyla$
    $X_t\ ye\ \ rastgele\ poz\ ekle$
  $else$
    $x_t^{[i]}\ yi\ X_t\ ye\ ekle$
  $enddo$
$endfor$
$return\ X'$

Çıktı: X'

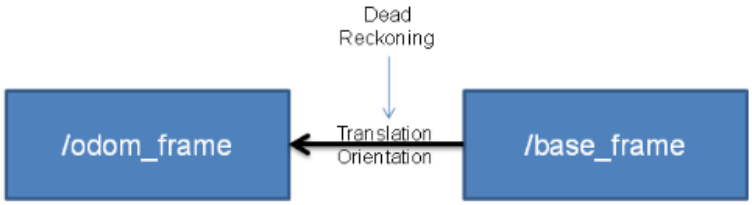
Şekil 3.24. AMCL kod bloğu (Çakmak 2014)

AMCL lazer tabanlı haritalama, lazer tarama ve dönüşüm mesajlarını alır ve çıktı olarak pozisyon tahmini yapmaya çalışır. Başlangıç değerleri sağlanarak parçacık filtresi ilk değer atamaları yapılır. İlk değer ataması yapılmadığında başlangıç değerleri otomatik olarak verilir. Parçacık filtresinin kullanacağı bu değerler (0,0,0) olarak belirlenir.

ROS paketi içinde bulunan AMCL paketinde bulunan dört ayrı pakete katılır bu paketler şu şekildedir. /scan mesajından oluşan "sensor_msgs/laserScan", tf/message tipinde belirtilmiş olan /tf mesajına, /initialpose mesajı içinde bulunan "geometry_msgs/PoseWithCovarianceStamped" ve /map mesajının kullanıldığı "nav_msgs/OccupancyGrid"dir. Buradan çıkan sonuçlar /amcl_pose mesajının yayınladığı "geometry_msgs/PoseWithCovariance Stamped" ve / particlecloud tarafından yayınlanan "geometry_msgs/Pose Array" mesajları yayınlanır. Bu süreçte tüm bu mesajların girdi olarak tanımlanması için bir haritaya ihtiyaç duyulmaktadır. Böylece harita üzerinde robotun pozisyon bilgileri belirlenir. Başlangıç pozisyonu olarak otomatik olarak belirlenen 0,0 pozisyonu olabileceği gibi, RVIZ görselleştirme programı ile işaretçi kullanılarak başlangıç noktası da belirlenir. Bu süreçte dönüşüm (tf) mesajı yayınlanmaktadır. gMapping üzerinden /odam mesajı alınarak gmapping üzerinden /map'e iletilir (Çakmak 2014).

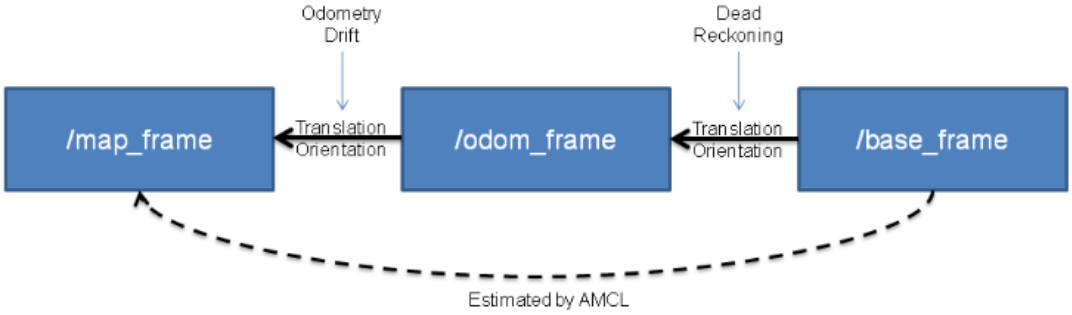
AMCL lazer taramasından gelen verileri odometri (~odom_frame_id) verilerine dönüştürür. Bu sebepten dolayı lazer tarama veri kümelerinin odometri için yayınlanan mesajlarını almak için dönüşüm (tf) yapısından bir yol bulması gerekmektektedir. İlk olarak alınan lazer taramasından sonra, AMCL taban çerçeve ve odom çerçevesi arasında dönüşüm arar ve sürekli işlenebilecek bir döngü oluşur ve ölü hesaplama meydana gelir. Bu sebepten

dolayı Şekil 3.25'de görüldüğü gibi AMCL taban çerçevesi hareketi ile lazer verilerini işleyemez.



Şekil 3.25. Odometri yerleştirme (Kang 2019)

Aşağıda Şekil 3.26'da görüldüğü gibi odometri ve AMCL kullanılarak yerleştirme arasındaki fark gözükmektedir. Genel çerçeve (global_frame_id) kullanılarak temel çerçevenin (base_frame_id) dönüşümleri tahmin edilir. Fakat sadece odometri çerçevesi (odom_frame_id) arasındaki dönüşüm mesajları yayınlanır. Burada asıl amaç ölü hesaplamayı kullanarak oluşan dönüşüm için hesapları ortaya çıkarmaktır.



Şekil 3.26. AMCL Haritası (Kang 2019)

3.2.5. gMapping

gMapping, ROS üzerinde çalışan gerçek zamanlı konum belirleme ve haritama algoritmasıdır. Ortam haritası çıkarıldıktan sonra bu harita verisi kullanılmaktadır. Robot

üzerinde bulunan sensör bilgilerinin bu harita üzerine aktarılması, mesafe ölçümü, engellere çarpmadan hareket etmesini gmapping sağlamktadır. gMapping esasında "slam_gmapping" düğümünü kullanır ve burada üretilen harita üzerinde çalışmalarını yürütür. gMapping, lazer mesafe ölçümü yapılan ve bu verileri kendi bünyesinde bulunduran "sensor_msgs/LaserScan" tipinde "/scan" mesaj türüne abone olmaktadır. Ayrıca "tf/tfMessage" tipinde "/tf" mesaj türüne abone olarak düğümlerin kendi aralarında oluşan iletişim sağlanmaktadır. Bu düğümler lazer, odometri ve base_link mesajlarına ait veri kümesini tutmaktadır ve gMapping düğümünün çalışabilir olması için gereklidir. Böylece lazer mesafe ölçüm veri seti kullanılarak robotun harita üzerinde konumunun tespiti sağlanır.

3.2.6. Bulanık mantık (Fuzzy logic)

Sistem odasında otonom hareket eden robot ile aldığımız sıcaklık verileri her kabinetin içinde bulunan NTI sıcaklık ve nem sensörlerinden gelen sıcaklık verileri ile ilişkilendirilmiştir. Bu ilişki bulanık mantık modeli ile oluşturulmuş, değerler bulanıklaştırılmış ve kurallar doğrultusunda çıkış parametresi olarak ızgaraların hangi derecede açılması gerektiğine karar verilmesi istenmiştir. Böylece kabinet içinde bulunan sunucu, depolama ve network cihazlarının uygun sıcaklıkta çalışmasının sağlanması amaçlanmıştır.

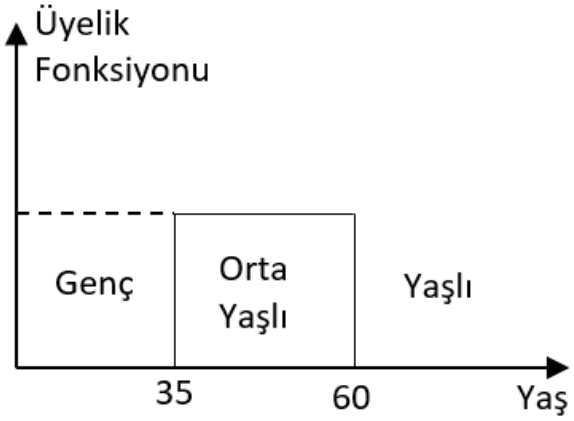
3.3.6.a. Bulanık mantık modelleme

Bulanık mantığın temel olarak kullanılma sebebi bir sürecin meydana gelme olasılığından daha çok sürecin oluşum derecesi ile ilgili bir ifadeyi tanımlayabilmektir. Bu kapsamda bulanık mantık olasılık konusunun devamı niteliğinde görülebilir. Fakat esasında ikisi çok farklı kavramlardır. Olasılık, bir işlemin olup olamayacağına bakarken bulanık mantık ise işlemin hangi aşamada olduğunu ve ortaya çıkan sonucun hangi aşamada meydana geleceğini ortaya koyar. Bulanık mantık kavramsal olarak çok detay ve yüksek ölçekte matematik içermesine karşın uygulama süreci oldukça basittir (Saday 2019).

Klasik yöntemlerde kullanılanılan mantık (0-1) kesin bir yargı içermektedir. Bu kapsamda bir eleman bir kümenin ya elemanıdır yada değildir. Bu yapıda oluşan kümelere crisp adı verilir. Klasik mantık kümesi bir A kümesini tanımlarken aynı zamanda bu kümenin değili olan kümeye sahiptir. Fakat bulanık mantık kümesi için ifade edilen değerler (0 ve 1) aralığını sınır kabul eder.

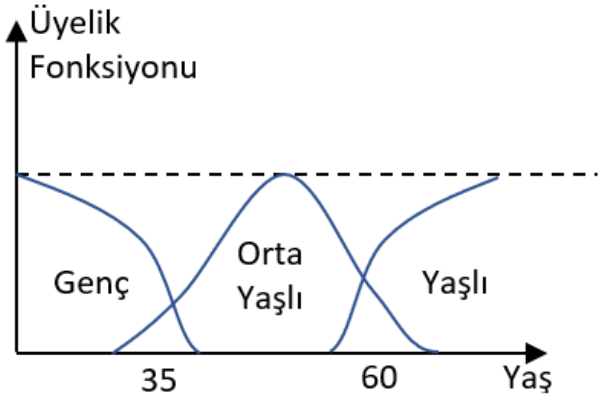
3.3.6.b. Bulanık kümeler

Bulanık küme kavramını tam olarak küme elemanı olabilmeyi derecelendireceği bir genelleştirmedir (Saday 2019). Bulanık mantık oluşturulurken temel olarak bulanık bir küme oluşturmak esas alınmıştır. Bir küme elemanı içinde bulunduğu kümeye ait ya da ait değilken, bu küme elemanı belirli bir oranda kümenin kismi bir elemanı olabilir. Bulanık mantıkta değerler kümesi kesin bir biçimde ifade edilemediği için matematiksel olarak ifade edilmeleri kolay değildir. Matematiksel ifadeler sadece sınır değerleri üzerinde işlem yapılmasına izin vermektedir. Bulanık mantık yöntemi doğrudan doğruya sayıların komşusal ilişkisini inceler. Karar sürecinde durumu ifade etmek için kullanılan sayı, kararın kabülünün gerçekleşmesini sağlar. Fakat bu karara sebep olan sayıya yakın olan sayılar karar sürecinin bir parçası olarak kabul edilecektir.Örnek olarak inceleyecek olursak Şekil 3.27'de görüldüğü gibi 0-35 yaş aralığı genç bireyleri, 35-60 orta yaşlı ve 60 yaş üzeri kişiler ise yaşlı sınıfını temsil etmektedir. Klasik küme teorisinde 36 yaşındaki bir birey orta yaşlı sayılırken 33 yaşındaki bir birey genç sayılmaktadır (Özek and Sinecen 2004).



Şekil 3.27. Klasik küme teorisi

Bulanık mantık açısından inceleyecek olursak, Şekil 3.28'de görüldüğü gibi 35 yaşındaki bir birey belirli ölçülerde hem genç hem de yaşlı sayılmaktadır.Bulanık mantıkta klasik mantığı nazaran kesin bir kanı yoktur. Günlük yaşantımızda kullandığımız daha esnek bir yaklaşım söz konusudur (Özek and Sinecen 2004).



Şekil 3.1. Bulanık küme teorisi

"A, R∈ (-∞, +∞)'da küme elemanı ise, μ_A (x) için üyelik fonksiyonu $R \rightarrow [0,1]$ aralığı oluşmaktadır. A kümesi $A = [a_1, a_3]$ aralığında yer alıyorsa genel ifade ile μ_A (x)" (Saday 2019) için üyelik fonksiyonu denklem (7)'deki formülü ile gösterilebilir.

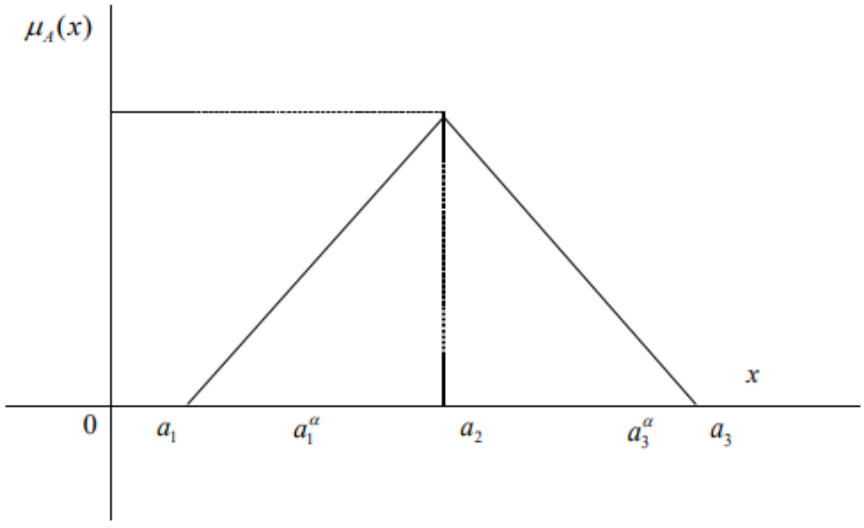
$$\mu_A(x) = \begin{cases} 0, x < a_1 \\ 1, \ a_1 \leq x \ \leq \ a_3 \\ 0 \ x > \ a_3 \end{cases} \quad (7)$$

Çeşitli üyelik fonksiyonları mevcuttur. Bunlar üçgensel, yamuk ve üstel gibi üyelik fonksiyonlarıdır.

Bu çalışmada üçgensel üyelik fonksiyonu ile bulanık hale getirilmiştir. $\mu_A(x)$ için oluşturulan üçgensel üyelik fonksiyonu denklem (8)'deki formül ile ifade edilmiştir.

$$\mu_A\,(x) = \begin{cases} 0, \ x < \ a_1 \\ \frac{x-a_1}{a_2-a_1}, a_1 \leq x \ \leq \ a_2 \\ \frac{a_3-x}{a_3-a_2}, a_2 \leq x \ \leq \ a_3 \\ 0, x \ > \ a_3 \end{cases} \quad (8)$$

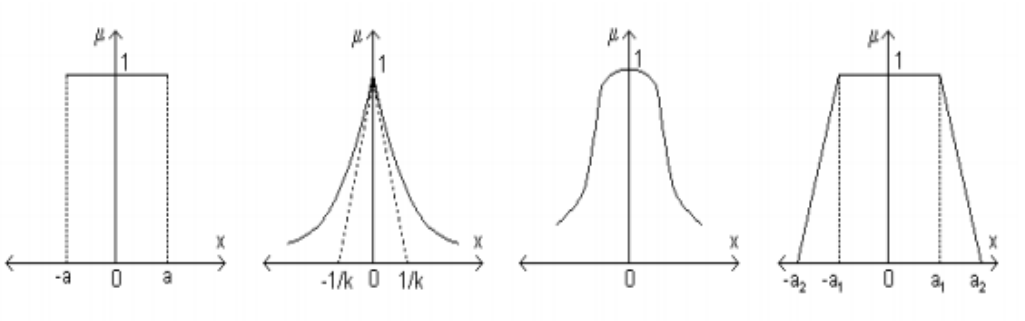
Yukarıda belirtilen formülde kümenin, A = (a_1 , a_2, a_3) olması gerekmektedir. Burada bulunan a_2 elemanı belirtilmiş olan normal değere sahip üyelik olarak tanımlanabilir. Bulanık mantıkta ise bir α katyasına bağlı olarak a_1 ve a_3 elemanlarının arasında bulunan a_2 elemanına yakın bulunan değerlere anlam atfederek a_2 temsil edilebileceği Şekil 3.29'da komuşuluk ifadesi gösterilmiştir.



Şekil 3.29. Sayıların komuşuluğu (Saday 2019).

3.3.6.c. Üyelik fonksiyonları

Bulanık mantık uygulamaların üyelik fonsiyonlarında bulunan kurallar içinde sebep sonuç terimleri arasındaki ilişki yapılandırılmıştır. Üyelik ağırlığı ise bulanık mantık içinde tanımlanmış değer kümesine bakarak bu değerlerin daha güvenilebilir olduğunu öngörmektedir. Şekil 3.30'da çeşitli üyelik fonksiyonları ve onların derece hesaplamaları gösterilmiştir.



μ(x)=0 $-\infty<x<-a$ **μ(x)=1** $-a\leq x\leq a$ **μ(x)=0** $a\leq x\leq\infty$

μ(x)=e^{kx} $-\infty<x<0$ **μ(x)=e^{-kx}** $0\leq x\leq\infty$ **(k>1)**

μ(x)=e^{-kx}

μ(x)=0 $-\infty<x\leq -a_2$

μ(x)= $\frac{a_2-x}{a_2-a_1}$ $-a_2\leq x\leq -a_1$

μ(x)= 1 $-a_2\leq x\leq a_1$

μ(x)= $\frac{a_2-x}{a_2-a_1}$ $a_1\leq x\leq a_2$

μ(x)= 0 $a_2\leq x\leq\infty$

Şekil 3.30. Üyelik fonsiyonları ve derece hesaplamaları (Saday 2019).

4. ARAŞTIRMA BULGALARI ve TARTIŞMA

Bu tez çalışması kapsamında veri merkezleri içinde otonom hareket eden robot ile ortam sıcaklığı ölçütlendirilerek, sabit noktalarda konumlandırılmış sıcaklık ve nem sensörlerinden farklı olarak homejen bir ölçüm yapılması amaçlanmış bu doğrultuda sistem odasında bulunan cihazların en ideal sıcaklıkta çalışmalarını sağlamak için kabinet önünde bulunması gereken ızgaraların hangi derecede açık olmasını sağlanmıştır. Şekil 4.1'de bu çalışma sürecinin akış diyagramı belirtilmiştir.

Şekil 4.1. Süreç akış diyagramı

Sistem odamızda konumlandırdığımız robot, ilk olarak tanımadığı ortam gezdirilerek tanıtılarak ortam haritası çıkartılmıştır. Daha sonra otonom hareket için kullanacağı bu harita üzerinde gideceği rota tespit edilmiştir. Arduino kart üzerinde yerleştirdiğimiz BME 280 hassas sıcaklık ölçümü yapan sensör kabinetlerin ön tarafında bulunan yükseltilmiş taban üzerinde hareket ederek rotası boyunca 1 saniyelik aralıklarla toplam

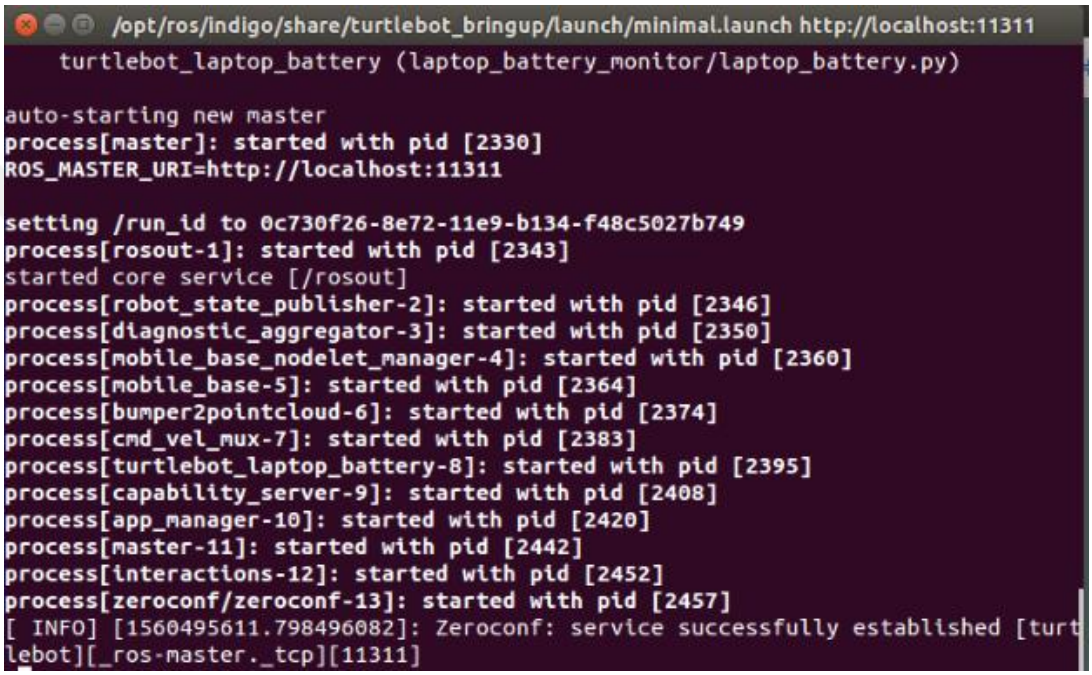
20 defa hassas ölçüm yapmaktadır.Aynı zaman süresinde kabinet içi sıcaklığı ölçümleri NTI cihazları tarafından yapılmaktadır. Elde edilen sıcaklık veri seti bulanık mantık modelleme ile anlamsal bir ilişki sonucu ızgaraların konumunun açısal olarak açık ve kapalı bir biçimde konumlandırması yapılarak kabinet içi sıcaklık değerlerin en ideal sıcaklık değerlerini yakalayarak veri merkezi cihazlarının tam performanslı çalışması amaçlanmıştır. Bu Sistem üzerinde çalışan senaryoya göre bulunan bulgular kısaca açıklanmıştır.

4.1. Veri Merkezinin Otonom Robot ile Haritalanması

Bu çalışmada otonom hareket eden robot olarak Turtlebot 2 seçilmiştir. Turtlebot 2 robotunu çalıştırmak için kullanılan en uygun işletim sistemi Linux işletim sisteminin 14.06 kararlı sürümüdür. Aynı zamanda bu işletim sistemi ile birlikte uyumlu çalışan robot işletim sistemi platformu ise ROS indigo versiyonu seçilmiştir. İlk olarak yapılan işlem Şekil 4.2'de belirtilmiş veri merkezi içinde konumlandırılan Turtlebot 2 robotuna ortam tanıtılmıştır.

Şekil 4.2. Atatürk Üniversitesi veri merkezi genel görünüşü

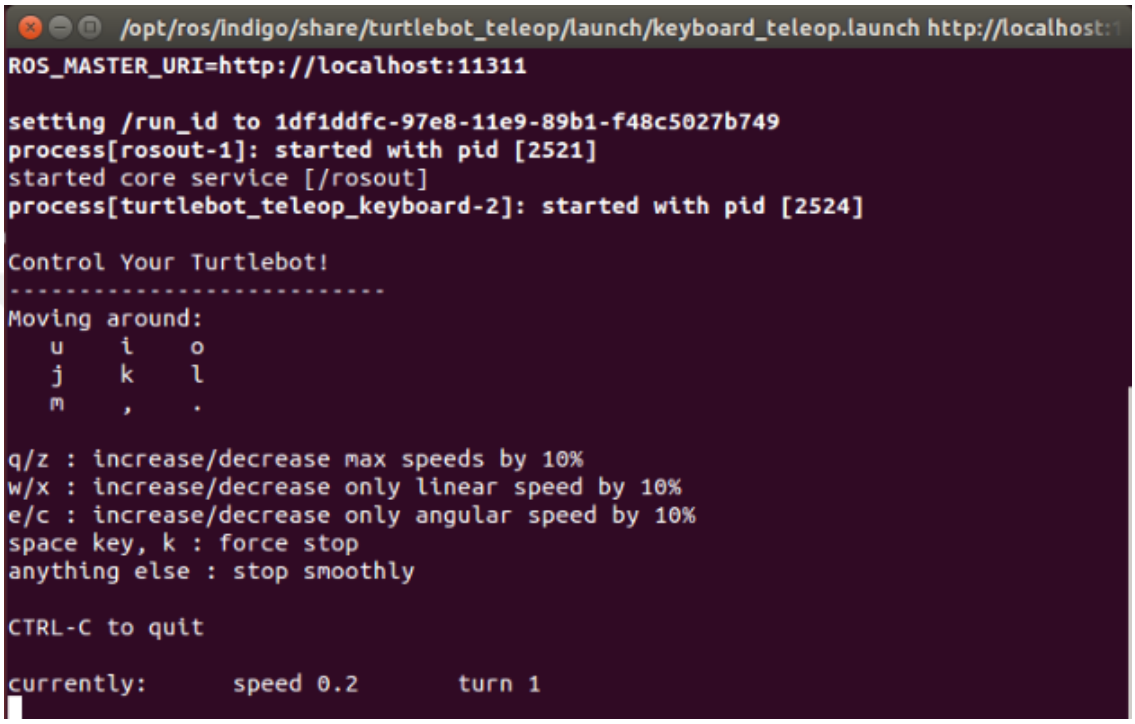
Bu işlem gerçekleştirilirken öncelikle ROS işletim sistemi ile robotun donanımsal bileşenlerini konuşturmak amacı ile terminale "ROSlaunch Turtlebot_bringup minimal.launch" komutu yazılarak işletim sistemi ile robot konuşturulmuştur. Doğru bir biçimde iletişime geçmesi durumunda Şekil 4.3'de gösterildiği gibi bir terminal ekranı karşımıza çıkmaktadır.



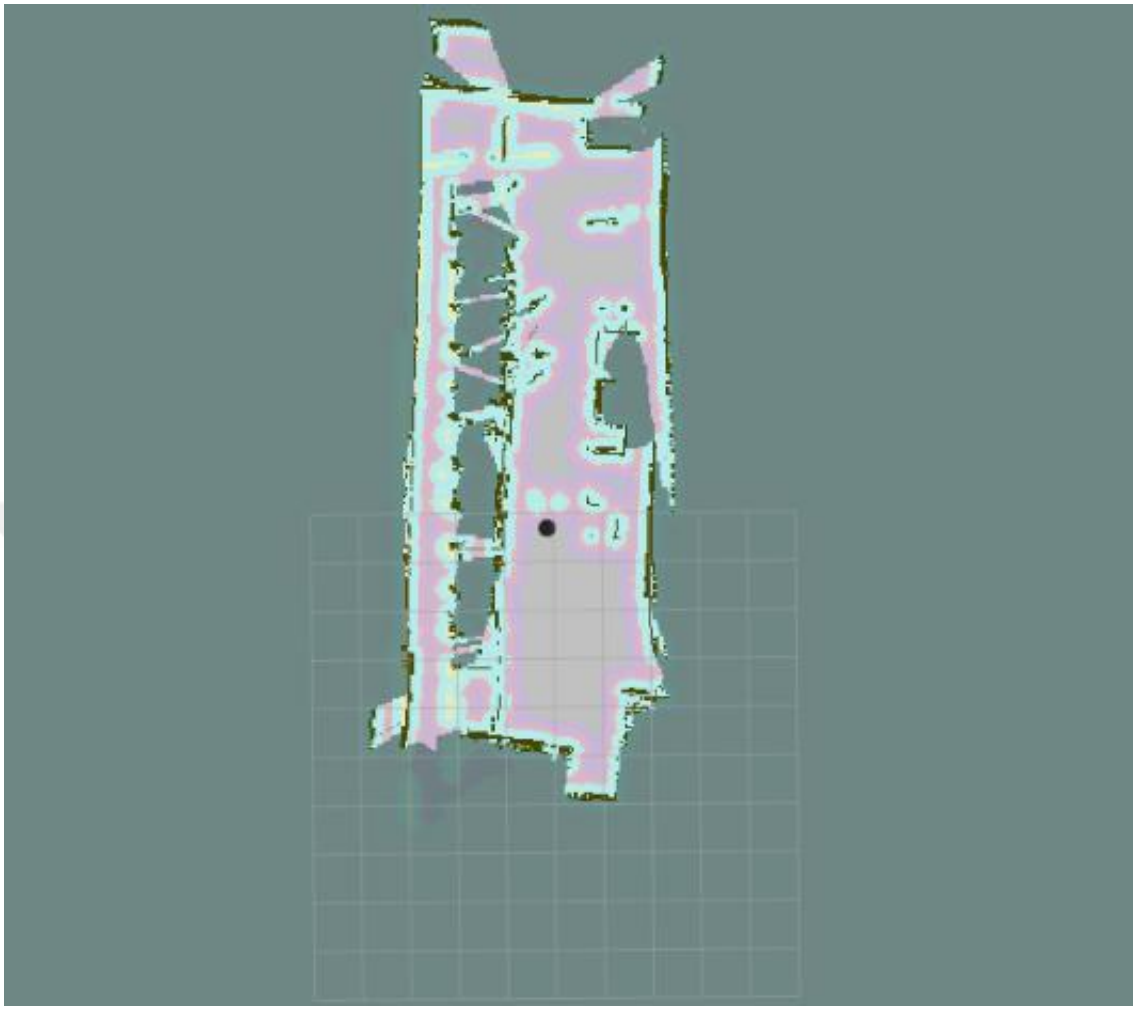
Şekil 4.3. Robot donanımı ve ROS'un iletişimi

Gerçek zamanlı haritalama yapmada "gMapping" nodu kullanılır. Bu node Robot işletim sistemi içinde bulunan navigasyon paketi içinde bulunan bir noddur ve harekete geçirmek için "ROSlaunch Turtlebot_navigation gmapping_demo.launch" komutu yazılır. Haritalama işlemi yapılırken çıkarılan ortam haritası 2 boyutlu görselleştirme programı rviz üzeri yansıtılır. Bu işlemi gerçekleştirmek üzere yazılacak komut satırı "rviz_launchers" paketi içinde bulunan "view_navigation" nodudur ve "ROSlaunch Turtlebot_navigation gmapping_demo.launch" işlem gerçekleştirilir. Böylece robotumuz gerçek zamanlı olarak ortam haritası çıkarmaya hazır hale getirilmiştir. ROS içinde bulunan "Turtlebot_teleop" hareket paketi içinde bulunan "keyboard_teleop" nodu çalıştırılarak Şekil 4.4'de belirtilen hareket, hız ve dönüş kontrol tuşları kullanılarak

robotumuzun gerçek zamanlı harita çıkarması sağlanır. Ortaya çıkan harita robotun otonom hareketini sağlayacak harita verisi Şekil 4.5'de görüldüğü gibi oluşturulur ve .yaml dosya uzantısı ile tutulur.



Şekil 4.4. Hareket, hız ve dönüş kontrolü sağlayan tuş takımı



Şekil 4.5. 2 Boyutlu ortam haritası

Elde edilen harita verileri AMCL yaklaşımı kullanılarak lazer tabanlı haritalama, lazer tarama ve dönüşüm mesajlarını alır ve çıktı olarak pozisyon tahmini yapmaya çalışır. Hali hazırda aktif olarak iletişim halinde olan robot donanımı ve ROS işletim sistemi üzerinden "Turtlebot_navigation" paketi içinde bulunan "acml_demo" nodu kullanılır. Önceden çıkarttığımız harita bilgisini "map_file:=/home/ROS/verimerkezi.yaml" dosyasını çağırarak harita verileri Turtlebot 2 üzerinde bulunan Kinect kameranın anlayacağı lazer tabanlı haritalama veri setine dönüştürülerek odometri verileri alındı (odom received) mesajına çevrilir. 2 Boyutlu görselleştirme programı RVİZ üzerinde haritayı göstermek için "Turtlebot_rviz_launchers" paketi içinde bulunan "view_navigation" nodu çalıştırılır. Böylece sıcaklık verilerinin bir rota üzerinde alınması için yazılan kod çalıştırılarak ortam içinde homojen bir ölçüm alınması sağlanmıştır.

4.2. Ortam Sıcaklığının Ölçülmesi

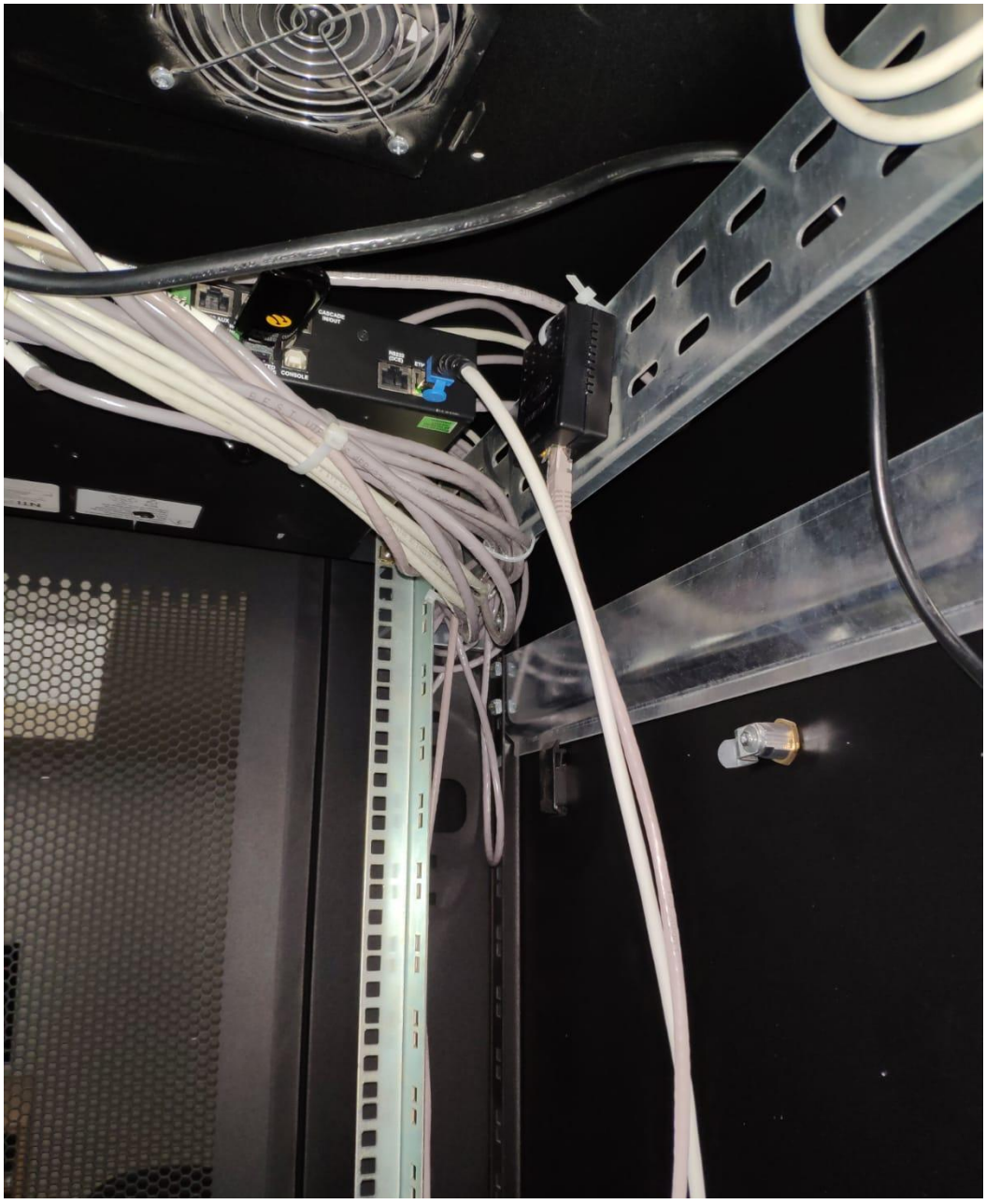
Veri merkezi ortamının hassas sıcaklık ölçümleri BME 280 sıcaklık ve nem sensörü ile yapılmıştır.Hareket rotası olarak giriş kapasının önünden iklimlendirme cihazlarının önüne kadar belirlenmiştir. Ölçümün otonom bir robot ile hareketli bir biçimde yapılmasının sebebi belli noktalarda konumlandırılan ölçüm cihazlarının farklı sonuç vermesidir. Şekil 4.6'da görüldüğü üzere iki farklı sıcaklık ölçümüne sahip iklimlendirme cihazı ve başka bir sensör farklı değerleri göstermektedir.

Şekil 4.6. İki farklı sensör verisinden gelen ortam sıcaklık ölçümü

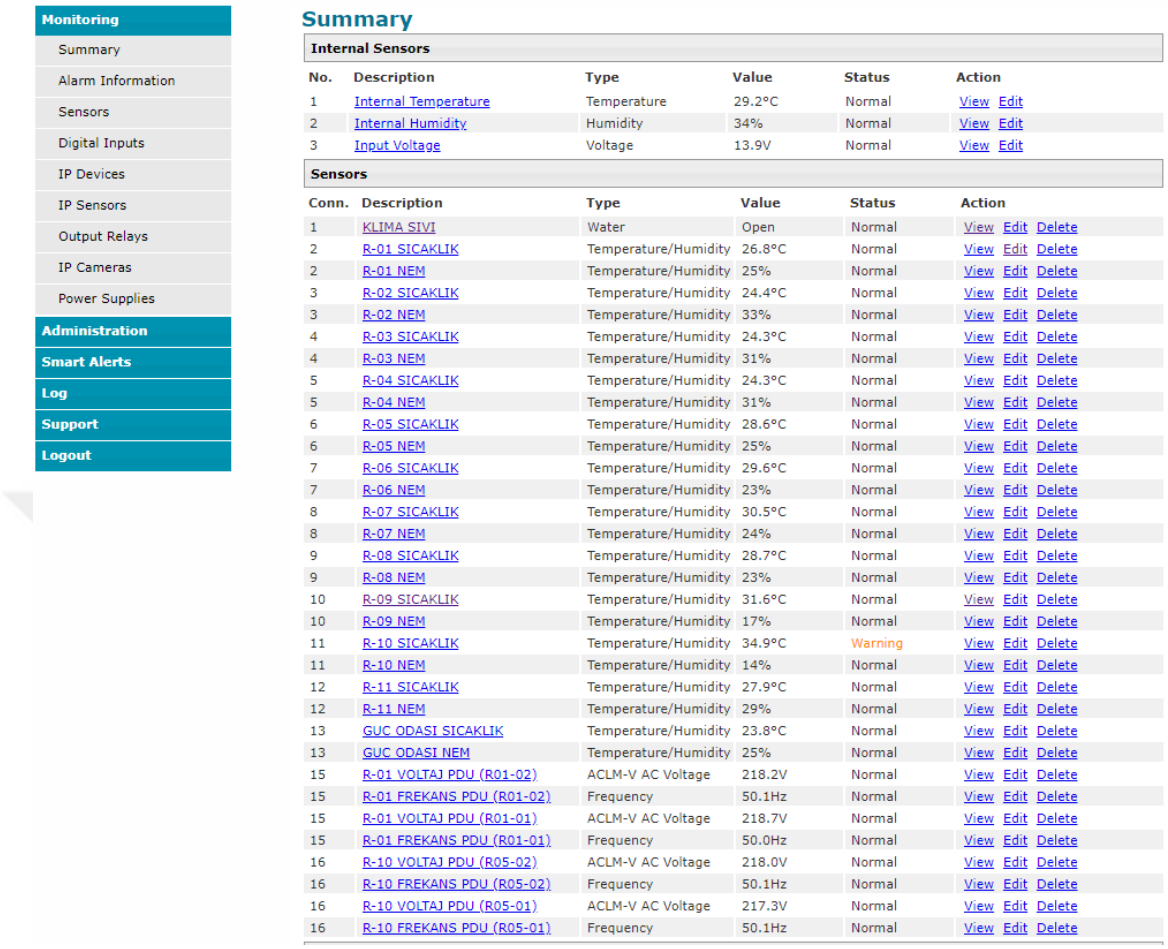
Sıcaklık ölçümü yapılırken ULAKNET verilerine ait anlık kullanıcı sayıları incelenerek en yoğun çalışma saat aralığı olan sabah 8 ile akşam 8 saatleri arasında her saat başı ölçüm yapılarak genel bir ortam sıcaklığı değeri yakalanmaya çalışılmıştır. Ölçüm robotun başlangıç pozisyonundan başlayarak hedef pozisyonu arasında geçen zamanda aralığında her 1 saniye aralıklarla tespit edilerek genel ortalaması alınarak o saat için ortam sıcaklığının tespit edilmesi amaçlanmıştır.

4.3. Kabinet İçi Sıcaklığın Ölçülmesi

Veri merkezi içinde bulunan her bir kabinet içine Şekil 4.7'de gösterildiği gibi konumlandırılan IP protokülü ile iletişim kurulan NTI E-STS sıcaklık ve nem sensörü kabinet içindeki sıcaklık verilerini alarak 1 dakika arayla Şekil 4.7'de görüldüğü gibi web arayüz ekranına göndermektedir.



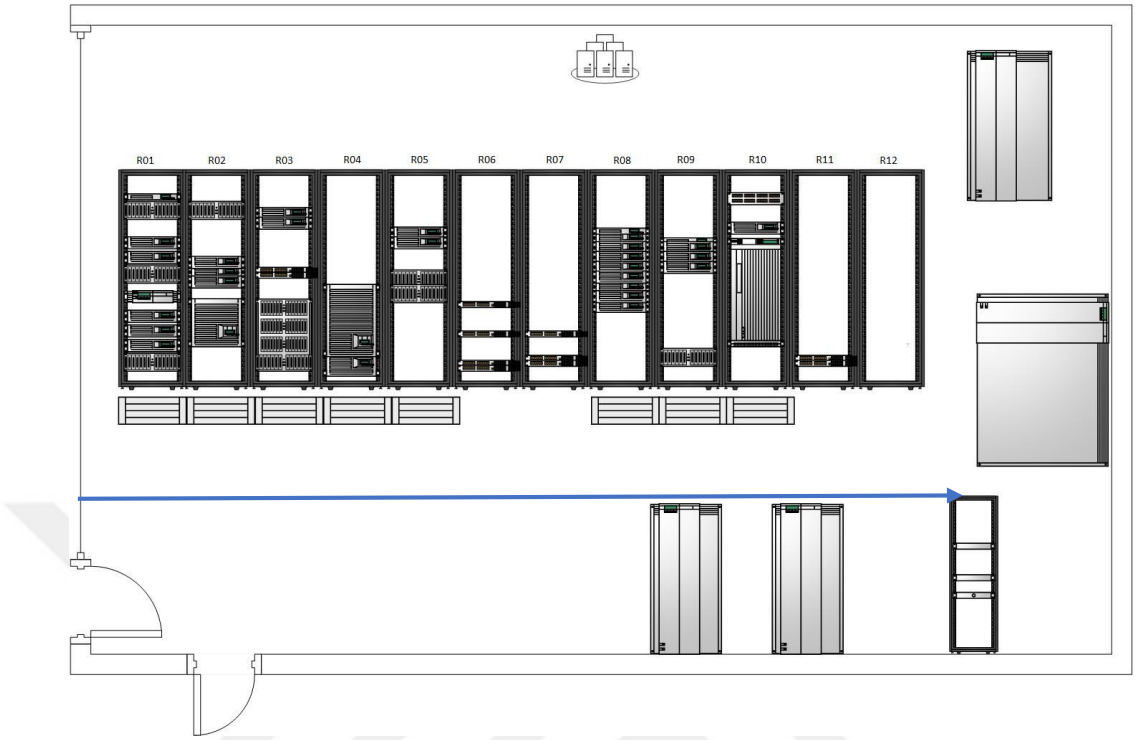
Şekil 4.7. NTI E-STS cihazı kabinet içi konumu



Şekil 4.8. NTI E-STS Sıcaklık ve Nem Sensörü Web Arayüzü

4.4. Sıcaklık Veri Setleri

Ortam sıcaklığı için veri seti oluşturulurken hareketli robotumuz kabinet önünde bulunan yükseltilmiş tavan üzerinde Şekil 4.9'de gösterildiği doğrultuda ilk kabinet önünden hareketine başlayarak bütün kabinetler önünden geçerek iklimlendirme cihazına yakın bir konumda hareketini tamamlamıştır. Belirlenen bu rotayı yaklaşık olarak 20 saniyede tamamlamasından dolayı 1 saniye aralıklarla sıcaklık ölçümü yapılmıştır ve 20 adet sıcaklık ölçüm değeri elde edilmiştir. Her saat başı yapılan sıcaklık ölçümü için elde edilen tüm sıcaklık değerlerinin ortalaması alınarak sıcaklık hesaplanmış ve Çizelge 4.1 ve Çizelge 4.2'de gösterilmiştir.



Şekil 4.9. Veri merkezi kabinet görünümü ve Turtlebot 2 ile sıcaklık ölçüm rotası

Çizelge 4.1. Saat başı sıcaklık ölçüm değerleri 1

08:00 Sıcaklık Ölçümü	09:00 Sıcaklık Ölçümü	10:00 Sıcaklık Ölçümü	11:00 Sıcaklık Ölçümü	12:00 Sıcaklık Ölçümü	13:00 Sıcaklık Ölçümü
24.92 ℃	24.63 ℃	24.63 ℃	24.67 ℃	25.02 ℃	24.37 ℃
24.90 ℃	24.63 ℃	24.61 ℃	24.63 ℃	25.00 ℃	24.33 ℃
24.88 ℃	24.62 ℃	24.56 ℃	24.60 ℃	24.97 ℃	24.29 ℃
24.80 ℃	24.59 ℃	24.35 ℃	24.58 ℃	24.91 ℃	24.23 ℃
24.78 ℃	24.53 ℃	24.30 ℃	24.52 ℃	24.86 ℃	24.17 ℃
24.67 ℃	24.50 ℃	24.19 ℃	24.49 ℃	24.81 ℃	24.14 ℃
24.62 ℃	24.43 ℃	24.10 ℃	24.38 ℃	24.74 ℃	24.05 ℃
24.58 ℃	24.36 ℃	24.01 ℃	24.30 ℃	24.66 ℃	23.99 ℃
24.56 ℃	24.32 ℃	23.94 ℃	24.24 ℃	24.58 ℃	23.93 ℃
24.54 ℃	24.24 ℃	23.84 ℃	24.17 ℃	24.54 ℃	23.89 ℃
24.53 ℃	24.17 ℃	23.75 ℃	24.09 ℃	24.47 ℃	23.86 ℃
24.51 ℃	24.12 ℃	23.64 ℃	24.05 ℃	24.34 ℃	23.84 ℃
24.40 ℃	24.04 ℃	23.58 ℃	24.01 ℃	24.19 ℃	23.76 ℃
24.37 ℃	24.00 ℃	23.47 ℃	23.90 ℃	24.20 ℃	23.71 ℃
24.32 ℃	23.94 ℃	23.44 ℃	23.82 ℃	24.20 ℃	23.62 ℃
24.30 ℃	23.87 ℃	23.37 ℃	23.74 ℃	24.16 ℃	23.59 ℃
24.26 ℃	23.76 ℃	23.34 ℃	23.71 ℃	24.10 ℃	23.54 ℃
24.23 ℃	23.73 ℃	23.28 ℃	23.68 ℃	24.08 ℃	23.56 ℃
24.20 ℃	23.66 ℃	23.26 ℃	23.64 ℃	24.05 ℃	23.54 ℃
24.16 ℃	23.63 ℃	23.24 ℃	23.59 ℃	24.02 ℃	23.54 ℃
Ortalama Sıcaklık Değerleri ℃					
24.53 ℃	24.19 ℃	23.85 ℃	24.14 ℃	24.50 ℃	23.89 ℃

Çizelge 4.2. Saat başı sıcaklık ölçüm değerleri 2

14:00 Sıcaklık Ölçümü	15:00 Sıcaklık Ölçümü	16:00 Sıcaklık Ölçümü	17:00 Sıcaklık Ölçümü	18:00 Sıcaklık Ölçümü	19:00 Sıcaklık Ölçümü	20:00 Sıcaklık Ölçümü
24.37 ℃	24.31 ℃	23.87 ℃	23.32 ℃	23.40 ℃	23.63 ℃	23.37 ℃
24.33 ℃	24.31 ℃	23.86 ℃	23.29 ℃	23.39 ℃	23.62 ℃	23.36 ℃
24.29 ℃	24.30 ℃	23.78 ℃	23.28 ℃	23.39 ℃	23.60 ℃	23.34 ℃
24.23 ℃	24.29 ℃	23.72 ℃	23.23 ℃	23.38 ℃	23.58 ℃	23.29 ℃
24.17 ℃	24.27 ℃	23.69 ℃	23.18 ℃	23.29 ℃	23.52 ℃	23.25 ℃
24.14 ℃	24.24 ℃	23.64 ℃	23.09 ℃	23.23 ℃	23.43 ℃	23.23 ℃
24.05 ℃	24.18 ℃	23.53 ℃	23.07 ℃	23.22 ℃	23.39 ℃	23.18 ℃
23.99 ℃	24.12 ℃	23.48 ℃	23.02 ℃	23.20 ℃	23.35 ℃	23.11 ℃
23.93 ℃	24.09 ℃	23.41 ℃	22.95 ℃	23.15 ℃	23.32 ℃	23.05 ℃
23.89 ℃	24.04 ℃	23.37 ℃	22.88 ℃	23.08 ℃	23.27 ℃	22.98 ℃
23.86 ℃	23.96 ℃	23.34 ℃	22.82 ℃	23.02 ℃	23.23 ℃	22.97 ℃
23.84 ℃	23.94 ℃	23.33 ℃	22.77 ℃	22.99 ℃	23.14 ℃	22.96 ℃
23.76 ℃	23.93 ℃	23.31 ℃	22.72 ℃	22.96 ℃	23.14 ℃	22.92 ℃
23.71 ℃	23.89 ℃	23.21 ℃	22.67 ℃	22.93 ℃	23.14 ℃	22.85 ℃
23.62 ℃	23.91 ℃	23.17 ℃	22.63 ℃	22.91 ℃	23.12 ℃	22.80 ℃
23.59 ℃	23.83 ℃	23.14 ℃	22.61 ℃	22.87 ℃	23.10 ℃	22.78 ℃
23.56 ℃	23.77 ℃	23.11 ℃	22.56 ℃	22.84 ℃	23.07 ℃	22.77 ℃
23.54 ℃	23.77 ℃	23.06 ℃	22.52 ℃	22.82 ℃	23.06 ℃	22.72 ℃
23.53 ℃	23.76 ℃	23.01 ℃	22.51 ℃	22.80 ℃	23.04 ℃	22.69 ℃
23.52 ℃	23.73 ℃	22.97 ℃	22.48 ℃	22.79 ℃	23.03 ℃	22.66 ℃
Ortalama Sıcaklık Değerleri ℃						
23.89 ℃	24.02 ℃	23.46 ℃	22.85 ℃	23.09 ℃	23.38 ℃	23.01 ℃

Bu veri setine ek olarak her saat başı kabinet içi sıcaklıkları NTI Web arayüzünden ortam sıcaklığı ile eş zamanlı olarak alınan değerler aşağıdaki Çizelge 4.3 ve Çizelge 4.4'de gösterilmiştir.

Çizelge 4.3. Kabinet içi sıcaklık değerleri 1

	08:00 Sıcaklık Ölçümü	09:00 Sıcaklık Ölçümü	10:00 Sıcaklık Ölçümü	11:00 Sıcaklık Ölçümü	12:00 Sıcaklık Ölçümü	13:00 Sıcaklık Ölçümü
Kabinet 1	26.5	27.2	27.0	27.0	27.2	27.1
Kabinet 2	24.5	25.1	24.9	25.0	24.9	24.9
Kabinet 3	24.1	24.7	24.1	24.3	24.1	24.0
Kabinet 4	24.3	25.1	24.6	24.7	24.6	24.4
Kabinet 5	28.2	29.4	28.9	28.8	29.2	29.0
Kabinet 6	29.6	29.5	29.2	29.0	29.2	29.1
Kabinet 7	30.3	30.6	30.3	30.3	30.3	30.3
Kabinet 8	29.2	31.8	31.8	32.1	31.9	31.9
Kabinet 9	31.4	33.4	33.8	34.0	34.0	34.1
Kabinet 10	35.0	35.2	35.2	35.3	35.2	35.4
Kabinet 11	28.1	28.3	28.3	28.4	28.2	28.5

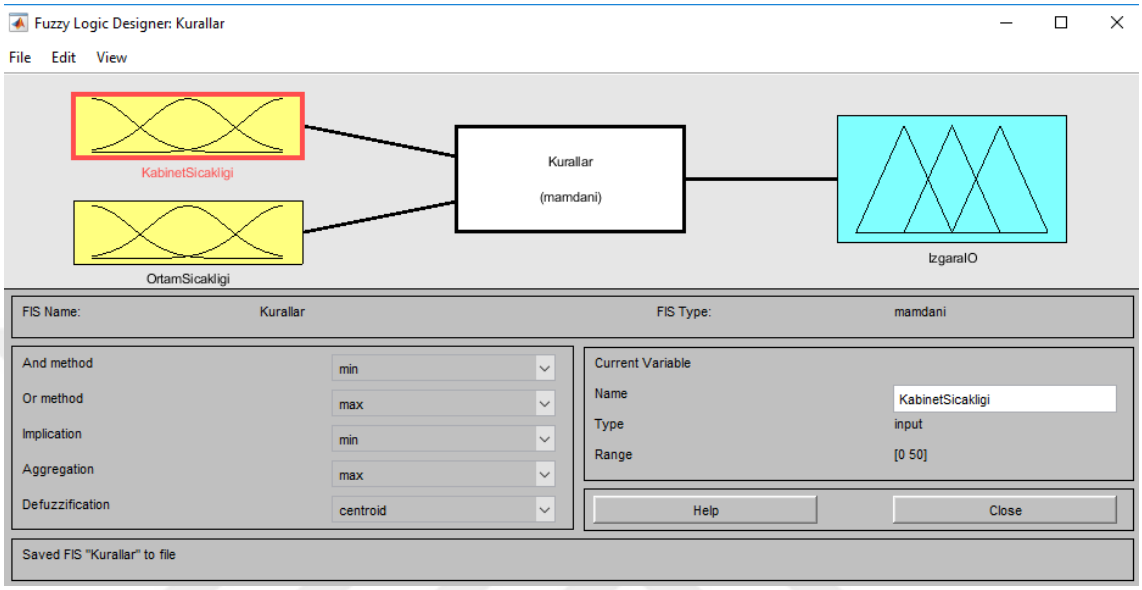
Çizelge 4.4. Kabinet içi sıcaklık değerleri 2

	14:00 Sıcaklık Ölçümü	15:00 Sıcaklık Ölçümü	16:00 Sıcaklık Ölçümü	17:00 Sıcaklık Ölçümü	18:00 Sıcaklık Ölçümü	19:00 Sıcaklık Ölçümü	20:00 Sıcaklık Ölçümü
Kabinet 1	27.3	28.4	27.9	27.7	27.6	27.4	27.2
Kabinet 2	25.2	26.0	25.6	25.4	25.6	25.6	25.6
Kabinet 3	24.3	24.8	24.9	24.7	24.7	24.5	24.5
Kabinet 4	24.7	24.9	24.7	24.5	24.5	24.3	24.5
Kabinet 5	29.2	29.4	29.2	29.2	29.1	29.0	28.5
Kabinet 6	29.3	29.5	30.3	30.0	30.0	29.9	29.7
Kabinet 7	30.5	30.8	31.0	30.8	30.8	30.8	30.6
Kabinet 8	32.2	32.3	29.0	29.1	29.2	28.9	29.8
Kabinet 9	34.4	34.7	32.1	31.7	31.9	31.8	31.9
Kabinet 10	35.6	35.6	35.1	34.9	35.2	35.1	35.2
Kabinet 11	28.8	28.8	27.8	27.6	28.0	27.9	27.9

4.5. Bulanık Mantık ile Modelleme

Bulanık sistem tasarlanırken ilk olarak Şekil 4.10'da görüldüğü gibi giriş ve çıkış parametreleri belirlenmiştir. Giriş parametrelerinin ilki kabinet içi sıcaklıkları olarak tanımlanmıştır. Bir diğer giriş parametresi ise otonom robot ile belirlenmiş ortam sıcaklığı değerleridir. Uzman görüşleri doğrultusunda kurallar belirlenmiştir ve çıkış parametresi olarak ızgaraların açıklık değeri açısal olarak belirtilmiştir. Tüm bu süreç grafiksel çoklu

alan simülasyon ve model tabanlı tasarım programı olan Matlab 2018b'nin fuzzy modülü ile gerçekleştirilmiştir.



Şekil 4.10. Giriş ve çıkış parametrelerinin gösterimi

Belirlenen bu parametrelerin bulanıklaştırılması gerekmektedir. Ortam sıcaklığı ve veri merkezinin en uygun çalışma sıcaklık değer aralığı The American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE) tarafından 2008 yılında paylaşılan sıcaklık alt sınırı 18℃ ve sıcaklık üst sınırı 27℃ olan referans değerleri alınmıştır. Ayrıca 20℃ ve 21.6 ℃ iyi bir değer aralığı iken 22.7 ℃ ve 24 ℃ derece aralığı veri merkezi için en ideal çalışma sıcaklığıdır (The Severn Group 2017). Ayrıca veri merkezlerinde kullanılan depolama alanları SSD (Solid state disk) diskler ile değiştirildiği taktirde sunucular -20℃ ve 40℃ gibi sıra dışı sıcaklıklarda çalışmaktadır (). Bu süreçte sıcaklık alt ve üst değerleri belirlemede Bu çalışmada verilen bazı değerler dış ortam sıcaklığının etkisi göz önüne alınarak verilmiştir. Elde edilen bu sıcaklık verileri ile bulanık küme parametrelerinin en büyük ve en küçük değer aralığı Çizelge 4.5'de görüldüğü gibi belirlenmiştir.

Çizelge 4.5. Giriş çıkış parametreleri

	Bulanık Değer Adı	**Çok Soğuk**	**Soğuk**	**Normal**	**Sıcak**	**Çok Sıcak**
Giriş	Kabinet Sıcaklığı	(-10) - 18	10 - 24	22 - 32	27 - 38	35 - 60
	Bulanık Değer Adı	**Çok Soğuk**	**Soğuk**	**Normal**	**Sıcak**	**Çok Sıcak**
Giriş	Ortam Sıcaklığı	(-10) - 10	0 - 20	10 - 30	20 - 40	30 - 50
	Bulanık Değer Adı	**0° - 45° - 90°**				
Çıkış	Izgara Açısı					

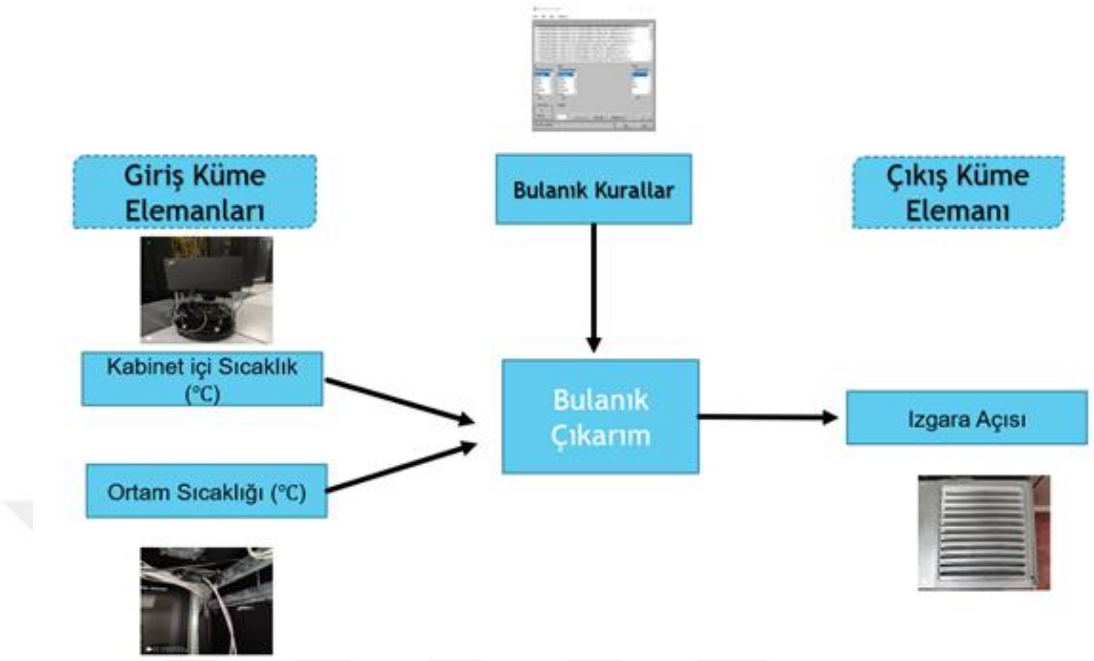
4.5.1 Bulanık uzman sisteminin tasarlanma aşamaları

Bulanık uzman sistemi tasarlanırken öncelikli olarak konusunda yetkin kişi veya kişilerden destek almak gerekmektedir.

Bu çalışmada tasarlanan bulanık mantık modeli için gerekli olan işlemler aşağıda belirtildiği gibidir:

1. Giriş ve çıkış parametrelerinin belirtilmesi
2. Giriş parametreleri olarak belirlenen, kabinet içi ve ortam sıcaklıklarının uzman görüşüne ve literatür çalışmasına göre bulanıklaştırılması. Üst ve Alt sıcaklık limit değerlerin tespit edilmesi
3. Üçgen üyelik fonsiyonun kullanılması ile bulanık kümelerin matematiksel formülleri bulunur.
4. Bulanık kuralların belirlenmesi. Kurallar Eğer Kabinet İçi Sıcaklık=Sıcak ve Ortam Sıcaklığı=Normal O halde Izgara Açısı=0.593 derece şeklinde oluşturulur.
5. Mandami çıkarımı çalıştırılır ve sistem çıkarımda bulunur.
6. Elde edilen sonuçlar durulaştırılır.

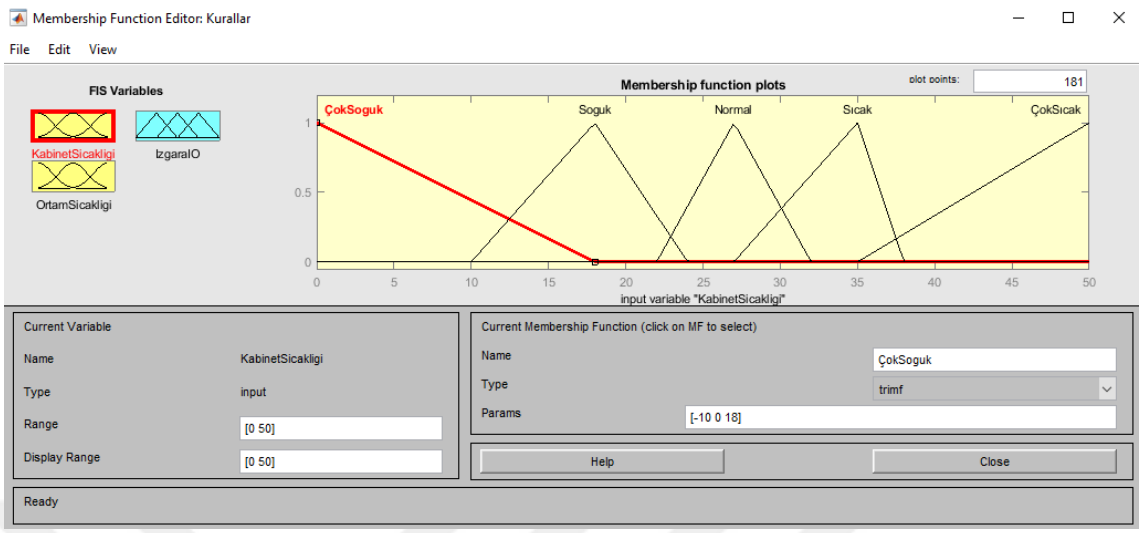
Şekil 4.11'de tasarlanan bulanık uzman sisteminin genel yapısı gösterilmektedir



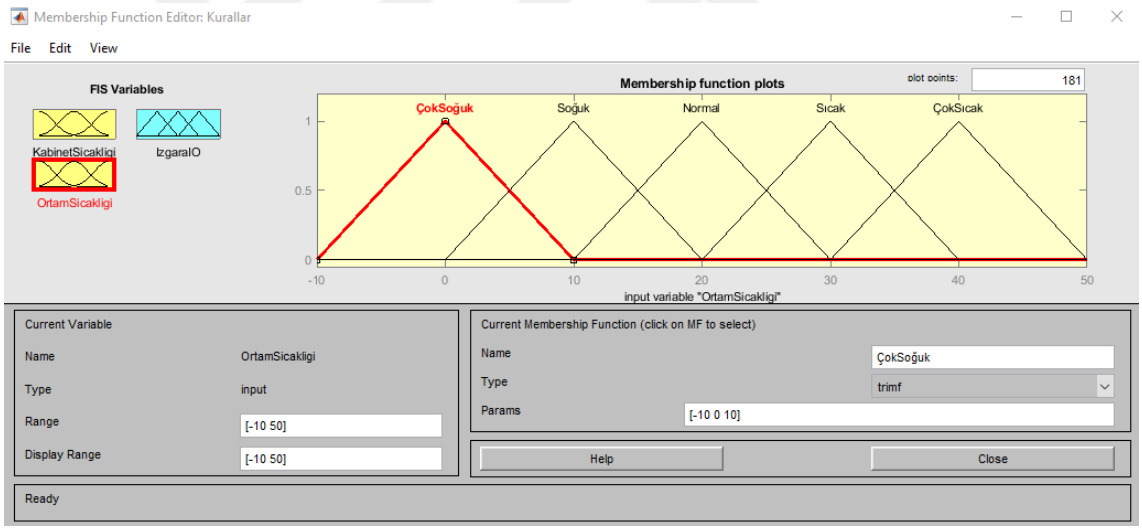
Şekil 4.11. Izgara durumunu tahmin eden bulanık sistem yapısı

4.5.1.a Giriş fonksiyonları

Giriş fonksiyonları Çizelge 4.5 de belirtildiği gibi, kabinet içi sıcaklıklar için -18 - +10°C aralığı için çok soğuk, 10 - 24°C aralığı için soğuk, 22 – 32°C aralığı normal, 27 – 38°C aralığı sıcak, 35 – 60°C aralığı çok sıcak olarak tanımlanmıştır. Aynı şekilde ortam sıcaklıkları, -10 - +10°C aralığı için çok soğuk, 10 - 20°C aralığı için soğuk, 20 – 30°C aralığı normal, 30 – 40°C aralığı sıcak, 40 – 50°C aralığı çok sıcak olarak tanımlanmıştır. Bu değerler fonsiyon değişkenleri olarak tanımlanmıştır. Giriş ve çıkış değerlerinin kolay yorumlanması açışından uzman görüşü alınarak üçgen fonksiyon seçimi yapılmıştır. Bu çalışmada fonksiyon aralıkları parametre olarak belirlendikten sonra Şekil 4.12 ve Şekil 4.13'te gözüktüğü gibi kabinet içi ve ortam sıcaklığına ait grafikler ortaya çıkmıştır.



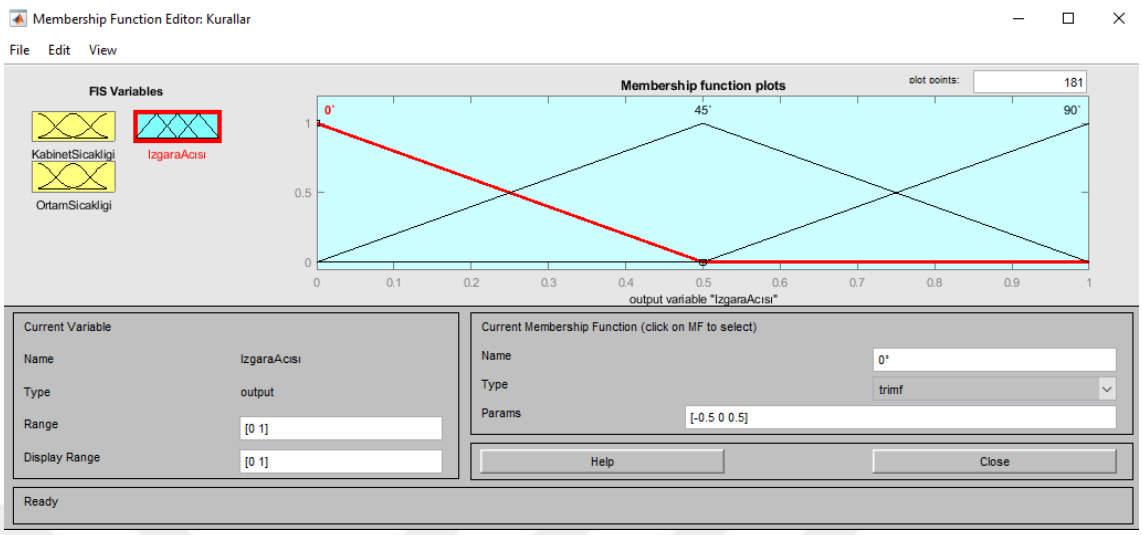
Şekil 4.12. Kabinet içi sıcaklık üyelik fonksiyonu



Şekil 4.13. Ortam sıcaklık üyelik fonksiyonu

4.5.1.b. Çıkış fonksiyonu

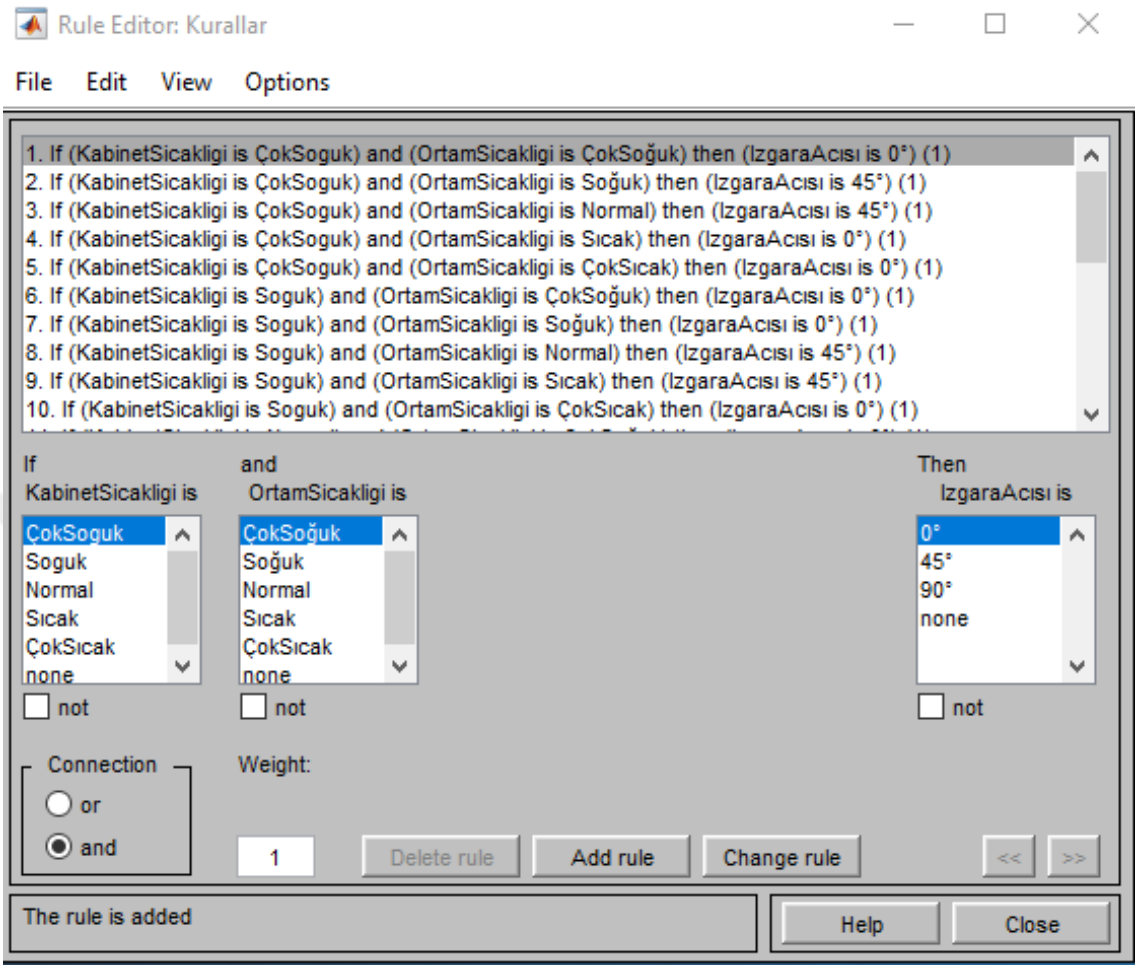
Çıkış fonsiyonu olarak sistem için tasarlanan 0 derece tam kapalı, 45 derece yarı açık ve 90 derece tam açık olarak çıkış parametresi değerleri belirlendikten sonra fonksiyon üyüleri olarak parametre girişi yapılmıştır. Şekil 4.14'de ızgara açısı üyelik fonksiyonu gösterilmiştir.



Şekil 4.14. Izgara açısı üyelik fonksiyonu

4.5.1.c. Bulanık kurallar

Kurallar tanımlanırken fuzzy toolbox'ında bulunan Rule Editör kullanılmıştır. Kurallar uzman görüşüne dayanılarak tanımlanmıştır. Kurallar oluşturulurken eğer – o halde ifadesi kullanılmıştır. Giriş değerleri verilir bulanık çıkarım vasitası ile sonuç elde edilir. Eğer kabinet içi sıcaklık=giriş parametre değeri ve ortam sıcaklığı=giriş parametre değeri O halde ızgara açısı=çıkış parametre değeri olarak oluşturulmuştur. Kabinet içi sıcaklık 5 farklı şart ile belirtilmiş aynı şekilde ortam sıcaklıklarıda 5 farklı şekilde belirtilmiştir. Böylece her iki üyelik fonksiyonundan gelen 5x5=25 adet kural meydana gelmiştir. Kuralların bir kısmı Şekil 4.15'te gösterildiği gibidir.



Şekil 4.15. Bazı kurallların gösterildiği matlab kurallar tablosu

Sistem üzerinde Mamdani bulanık çıkaram yöntemi kullanılarak sistem ile ilgili genel bir sonuca varılır. Son olarak elde edilen tüm bu değer kümesi durulaştırılır. Çok fazla durulaştırma yöntemi mevcut iken yaygın olarak kullanılan Centroid yöntemi tercih edilmiştir (Saday 2019).

4.6. Uygulamalar

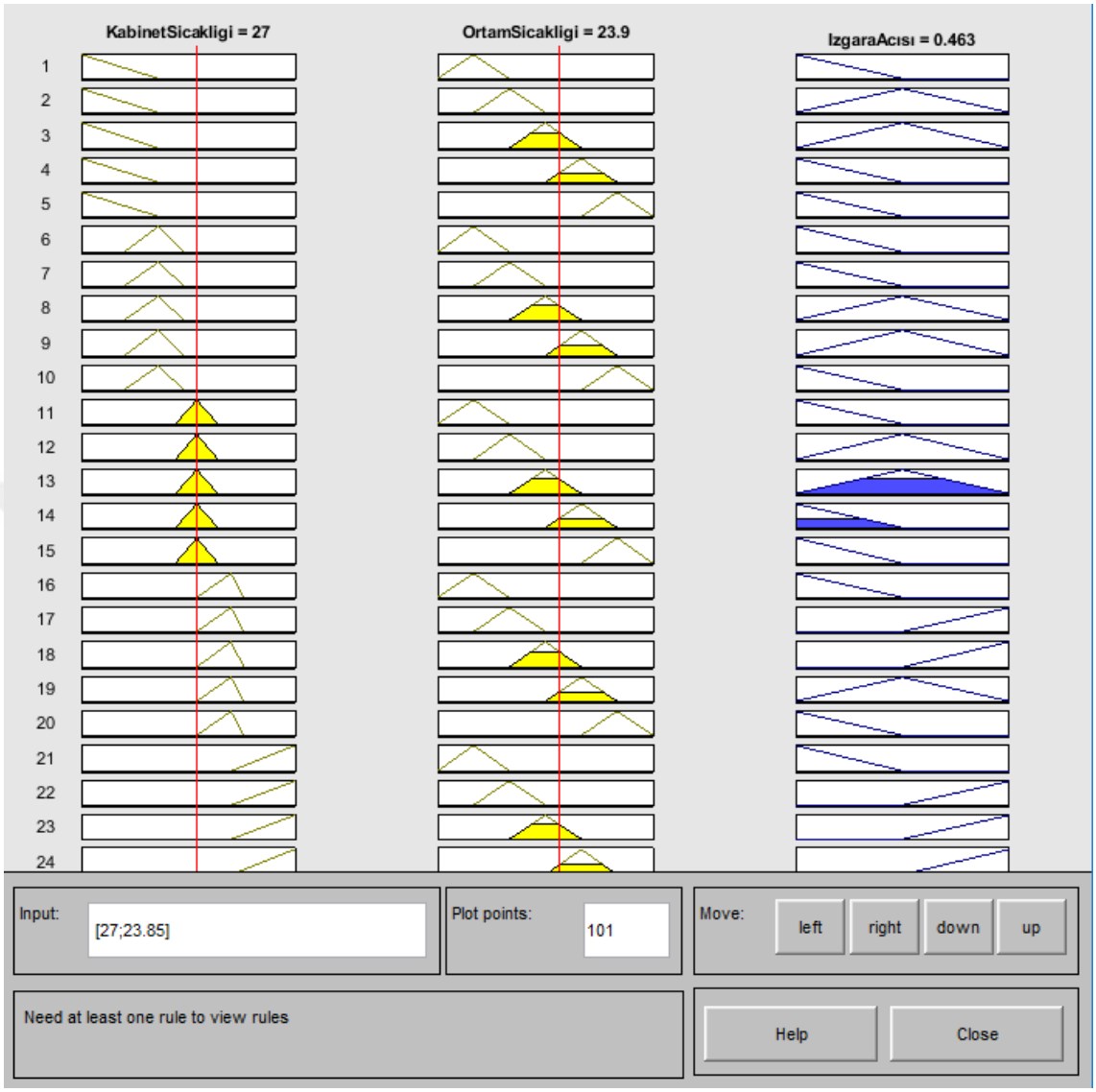
Bu çalışmada, her saat başı elde edilen her bir kabinet içi sıcaklığı ve ortam sıcaklığı verileri kullanılarak yükseltilmiş zemin ızgara açısı bulunması amaçlanmıştır. Bu uygulama yapılırken 3 farklı kabinet 3 farklı zaman aralığında incelenmiştir. Karşılaştırma yapılacak kabinet ve ortam sıcaklık verileri Çizelge 4.6'da gösterilmiştir.

Çizelge 4.6. Uygulama veri seti

	10:00 Kabinet Sıcaklığı	10:00 Ortam Sıcaklığı	15:00 Kabinet Sıcaklığı	15:00 Ortam Sıcaklığı	20:00 Kabinet Sıcaklığı	20:00 Ortam Sıcaklığı
Kabinet 1	27.0 ℃	23.85 ℃	28.4 ℃	24.02 ℃	27.2 ℃	23.01 ℃
Kabinet 8	31.8 ℃	23.85 ℃	32.3 ℃	24.02 ℃	29.8 ℃	23.01 ℃
Kabinet 10	35.2 ℃	23.85 ℃	35.2 ℃	24.02 ℃	35.2 ℃	23.01 ℃

4.6.1. Uygulama 1

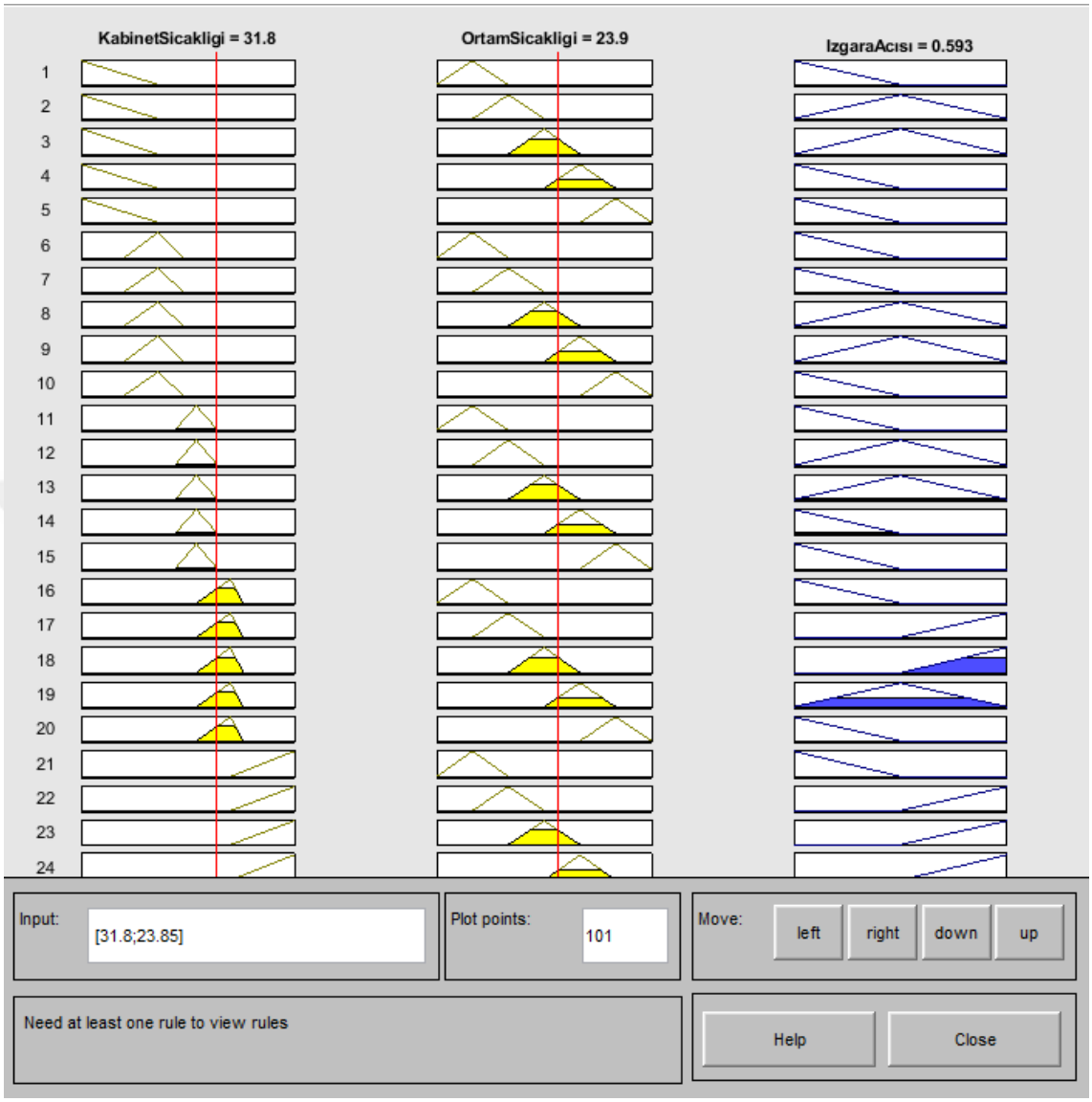
Kabinet 1 için saat 10:00'da yapılan ölçümlerde, giriş üyelik fonksiyon değerleri Çizelge 4.6'da belirtildiği üzere, kabinet sıcaklığı = 27.0℃, ortam sıcaklığı = 23.85℃ belirlenmiş olup, yükseltilmiş zemin üzerinde bulunan ızgaranın açısal değeri 0.463˚ olarak elde edilmektedir. Bu örnekte hazırlanmış olan ızgara açısal değeri Matlab üzerinde Şekil 4.16'da gösterilmektedir.



Şekil 4.16. Uygulama 1 için matlab çözüm görüntüsü

4.6.2. Uygulama 2

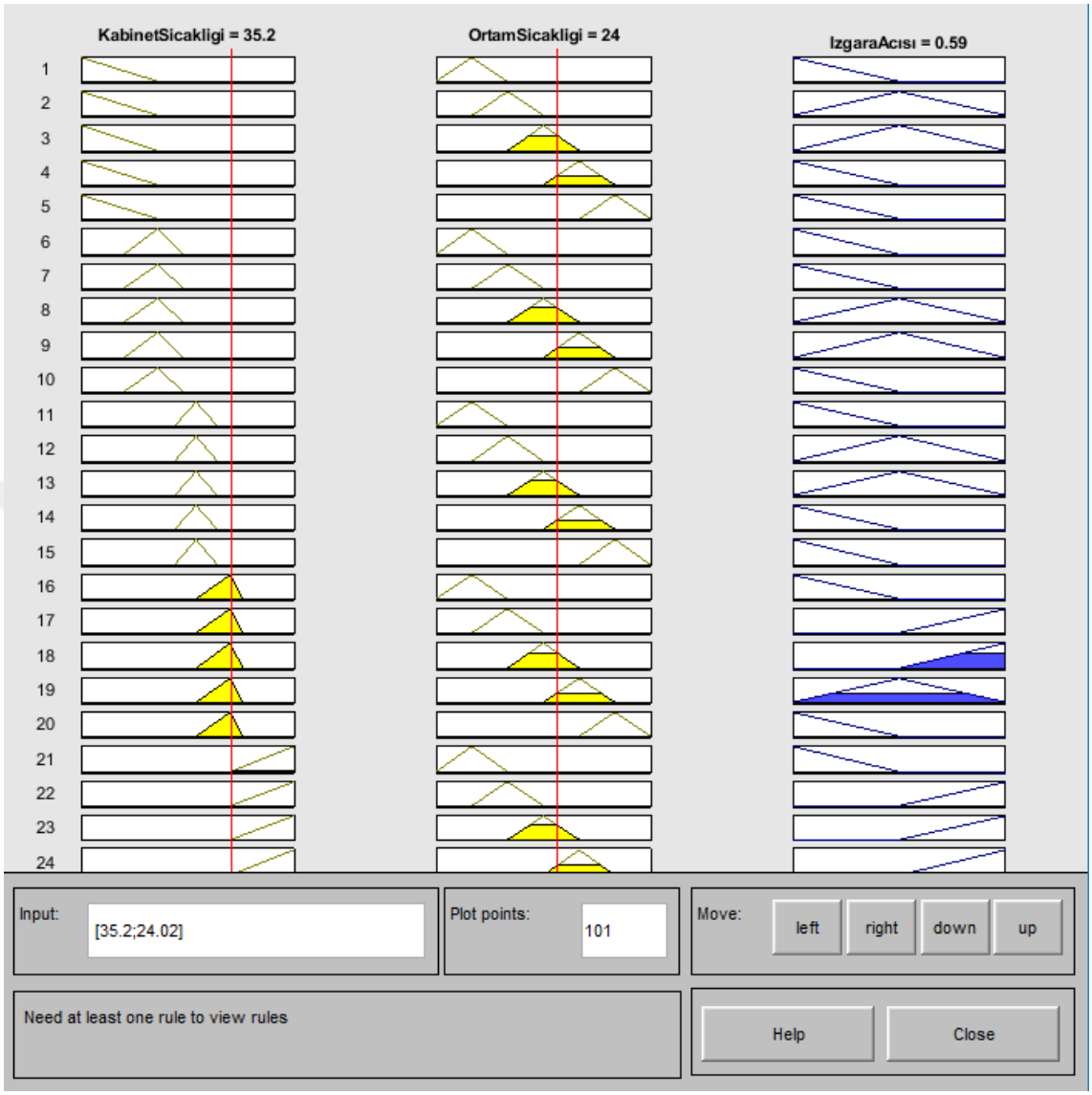
Kabinet 8 için saat 10:00'da yapılan ölçümlerde, giriş üyelik fonksiyon değerleri Çizelge 4.6'da belirtildiği üzere, kabinet sıcaklığı = 31.8℃, ortam sıcaklığı = 23.85℃ belirlenmiş olup, yükseltilmiş zemin üzerinde bulunan ızgaranın açısal değeri 0.593˚ olarak elde edilmektedir. Bu örnekte hazırlanmış olan ızgara açısal değeri Matlab üzerinde Şekil 4.17'de gösterilmektedir.



Şekil 4.17. Uygulama 2 için matlab çözüm görüntüsü

4.6.3. Uygulama 3

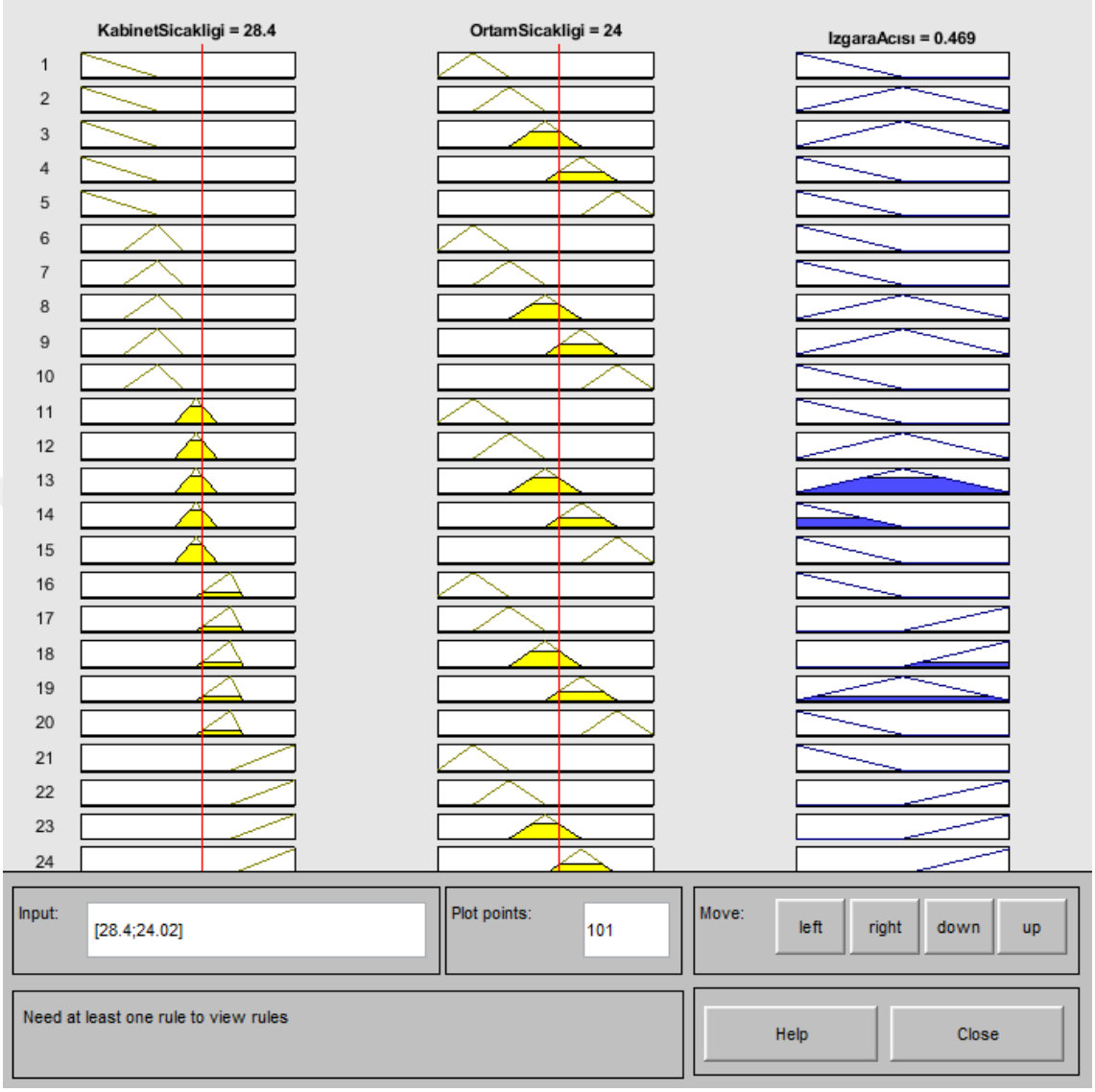
Kabinet 10 için saat 10:00'da yapılan ölçümlerde, giriş üyelik fonksiyon değerleri Çizelge 4.6'da belirtildiği üzere, kabinet sıcaklığı = 35.2℃, ortam sıcaklığı = 23.85℃ belirlenmiş olup, yükseltilmiş zemin üzerinde bulunan ızgaranın açısal değeri 0.59˚ olarak elde edilmektedir. Bu örnekte hazırlanmış olan ızgara açısal değeri Matlab üzerinde Şekil 4.18'de gösterilmektedir.



Şekil 4.18. Uygulama 3 için matlab çözüm görüntüsü

4.6.4. Uygulama 4

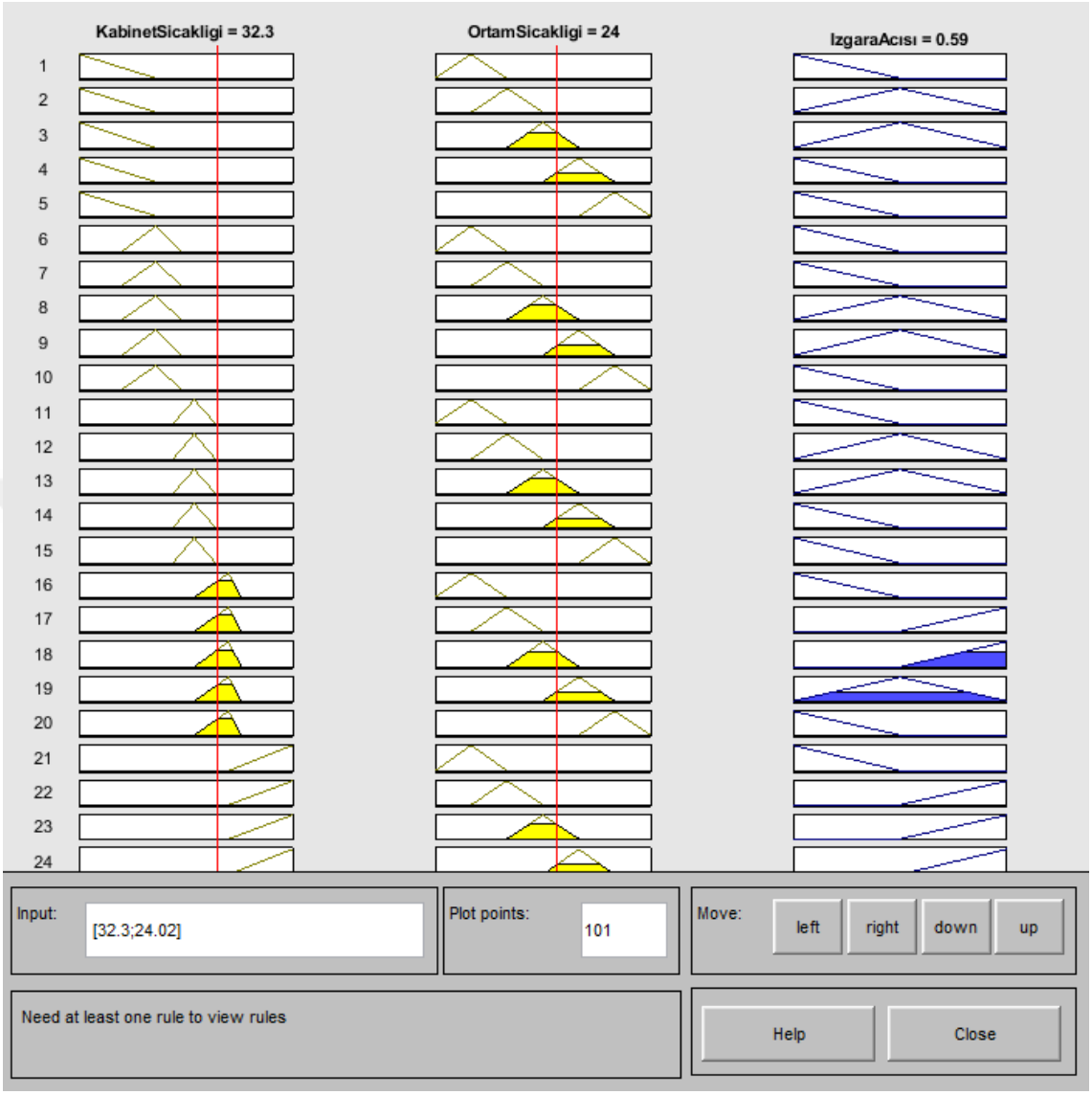
Kabinet 1 için saat 15:00'da yapılan ölçümlerde, giriş üyelik fonksiyon değerleri Çizelge 4.6'da belirtildiği üzere, kabinet sıcaklığı = 28.4℃, ortam sıcaklığı = 24.02℃ belirlenmiş olup, yükseltilmiş zemin üzerinde bulunan ızgaranın açısal değeri 0.469˚ olarak elde edilmektedir. Bu örnekte hazırlanmış olan ızgara açısal değeri Matlab üzerinde Şekil 4.19'da gösterilmektedir.



Şekil 4.19. Uygulama 4 için matlab çözüm görüntüsü

4.6.5. Uygulama 5

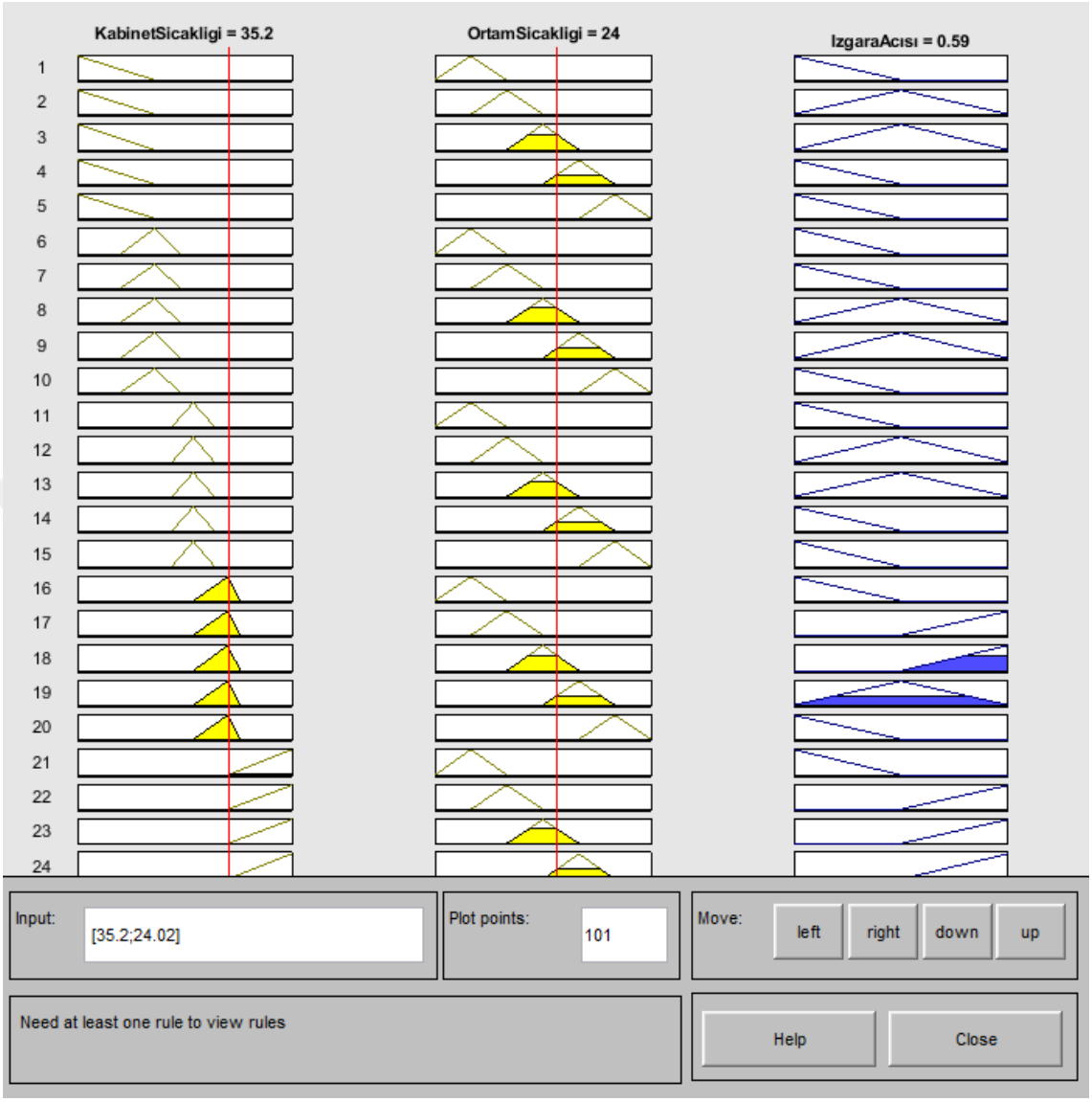
Kabinet 8 için saat 15:00'da yapılan ölçümlerde, giriş üyelik fonksiyon değerleri Çizelge 4.6'da belirtildiği üzere, kabinet sıcaklığı = 32.3℃, ortam sıcaklığı = 24.02℃ belirlenmiş olup, yükseltilmiş zemin üzerinde bulunan ızgaranın açısal değeri 0.59˚ olarak elde edilmektedir. Bu örnekte hazırlanmış olan ızgara açısal değeri Matlab üzerinde Şekil 4.20'de gösterilmektedir.



Şekil 4.20. Uygulama 5 için matlab çözüm görüntüsü

4.6.6. Uygulama 6

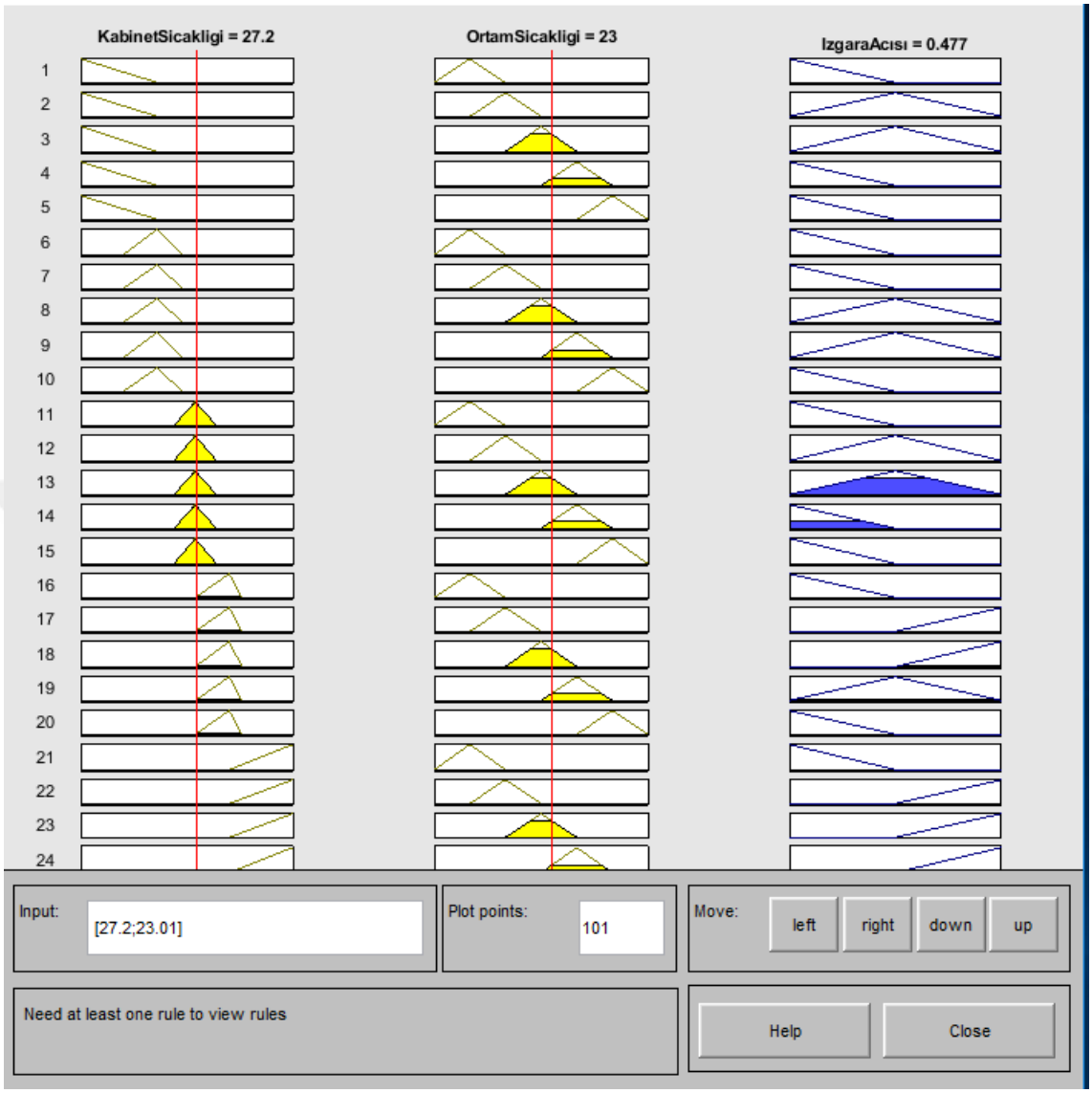
Kabinet 10 için saat 15:00'da yapılan ölçümlerde, giriş üyelik fonksiyon değerleri Çizelge 4.6'da belirtildiği üzere, kabinet sıcaklığı = 35.2℃, ortam sıcaklığı = 24.02℃ belirlenmiş olup, yükseltilmiş zemin üzerinde bulunan ızgaranın açısal değeri 0.59˚ olarak elde edilmektedir. Bu örnekte hazırlanmış olan ızgara açısal değeri Matlab üzerinde Şekil 4.21'de gösterilmektedir.



Şekil 4.21. Uygulama 6 için matlab çözüm görüntüsü

4.6.7. Uygulama 7

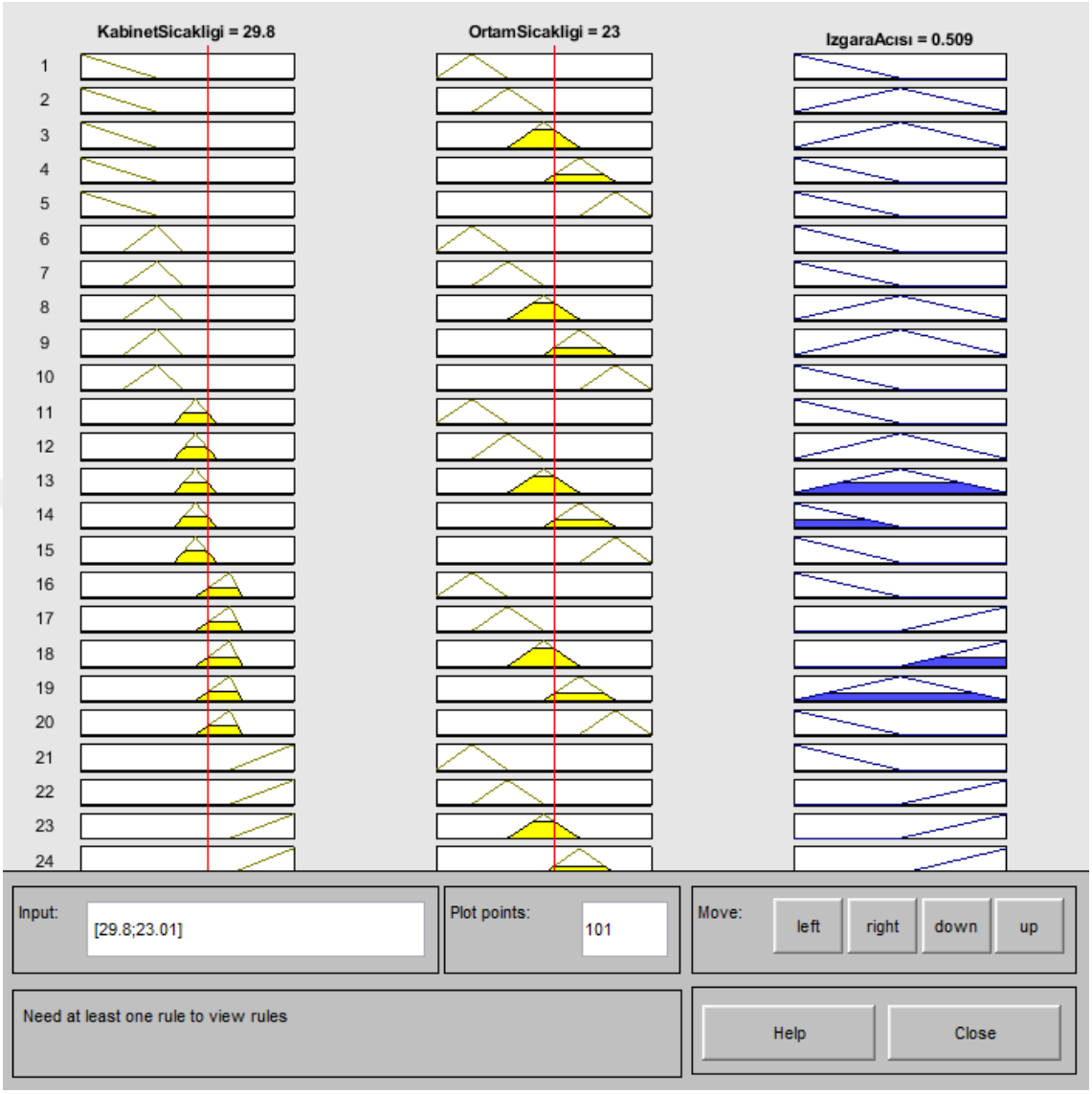
Kabinet 1 için saat 20:00'da yapılan ölçümlerde, giriş üyelik fonksiyon değerleri Çizelge 4.6'da belirtildiği üzere, kabinet sıcaklığı = 27.2℃, ortam sıcaklığı = 23.01℃ belirlenmiş olup, yükseltilmiş zemin üzerinde bulunan ızgaranın açısal değeri 0.477˚ olarak elde edilmektedir. Bu örnekte hazırlanmış olan ızgara açısal değeri Matlab üzerinde Şekil 4.22'de gösterilmektedir.



Şekil 4.22. Uygulama 7 için matlab çözüm görüntüsü

4.6.8. Uygulama 8

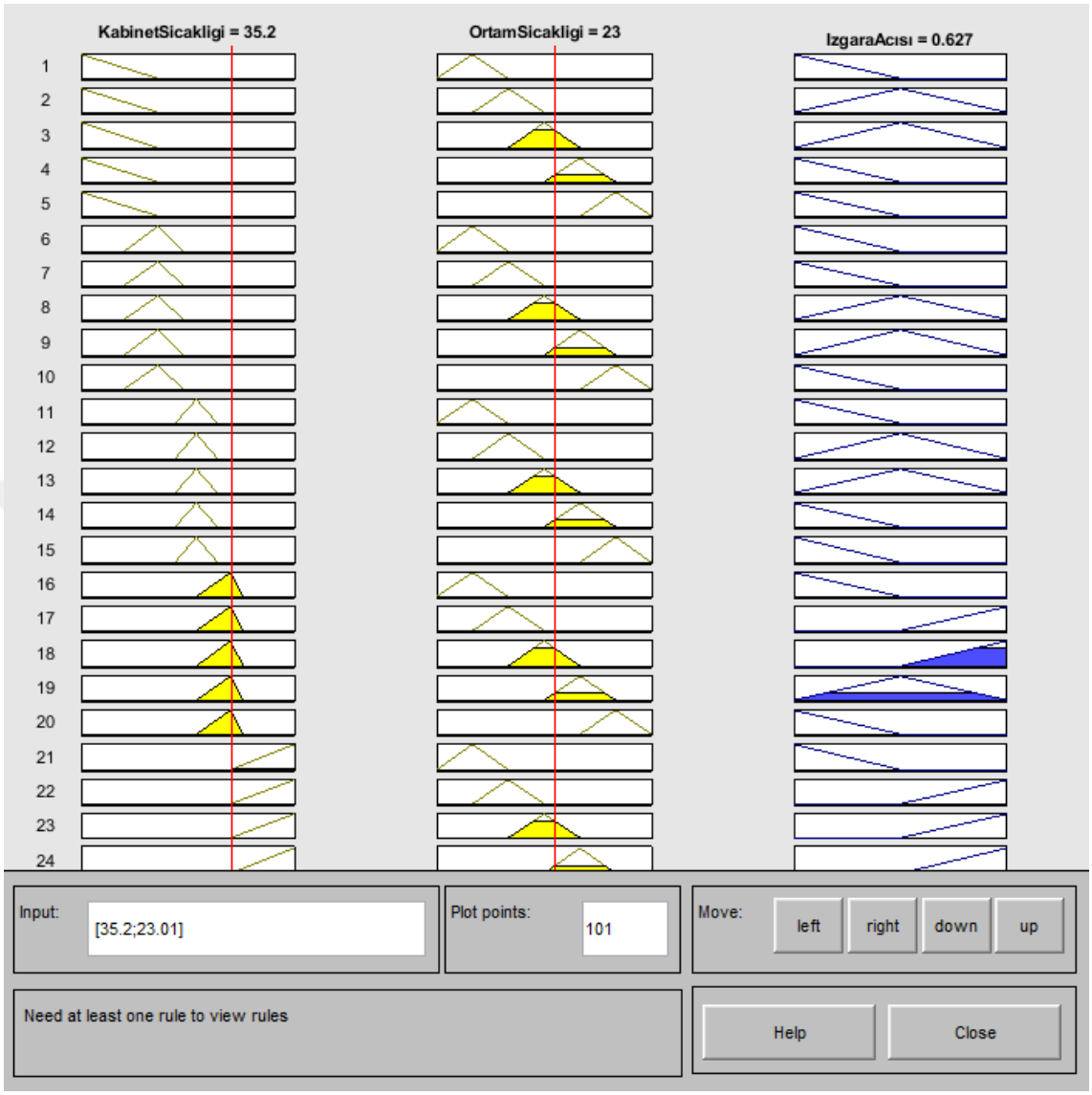
Kabinet 8 için saat 20:00'da yapılan ölçümlerde, giriş üyelik fonksiyon değerleri Çizelge 4.6'da belirtildiği üzere, kabinet sıcaklığı = 29.8℃, ortam sıcaklığı = 23.01℃ belirlenmiş olup, yükseltilmiş zemin üzerinde bulunan ızgaranın açısal değeri 0.509˚ olarak elde edilmektedir. Bu örnekte hazırlanmış olan ızgara açısal değeri Matlab üzerinde Şekil 4.23'de gösterilmektedir.



Şekil 4.23. Uygulama 8 için matlab çözüm görüntüsü

4.6.9. Uygulama 9

Kabinet 10 için saat 20:00'da yapılan ölçümlerde, giriş üyelik fonksiyon değerleri Çizelge 4.6'da belirtildiği üzere, kabinet sıcaklığı = 35.2℃, ortam sıcaklığı = 23.01℃ belirlenmiş olup, yükseltilmiş zemin üzerinde bulunan ızgaranın açısal değeri 0.627˚ olarak elde edilmektedir. Bu örnekte hazırlanmış olan ızgara açısal değeri Matlab üzerinde Şekil 4.24'de gösterilmektedir.



Şekil 4.24. Uygulama 9 için matlab çözüm görüntüsü

5. SONUÇ

Bu çalışma kapsamında otonom hareket eden mobil robot kullanılarak veri merkezi ortam sıcaklığı homojen bir biçimde tespit edilmiştir. Buradan elde edilen bulgular kabinet içi sıcaklıklarıyla eş zamanlı mukayese edilerek kabinet önünde konumlandırılmış ızgaraların kontrolü ile sıcaklık değerlerinin cihazların uygun bir biçimde çalışabileceği sıcaklıkları yakalaması amaçlanmıştır.

Gün içerisinde 08.00 ile 20:00 arasında yapılan sıcaklık ölçümleri incelendiğinde ortam sıcaklık değerinin saat 15:00 civarında 24.02℃ olduğu, saat 20:00 civarında ise 23.01 olduğu gözlemlenmiştir. Bu sıcaklık ölçümleri ele alındığında sunuculara gelen isteklerin saat 15:00 civarında mesai saatleri göz önüne alınarak sıcaklık seviyesinin arttığı görülmüştür. Bu doğrultuda sunuculara gelen isteklerin artmasının, kabinet içi sıcaklıklarının artmasını doğrudan etkilediği görülerek, sunucuların daha performanslı çalışması ortam sıcaklığının artmasına sebebiyet verdiği söylenebilir.

Aynı şartlar altında ızgara kapalıyken 0˚, ara değer olarak 45˚ ve açık iken 90˚ değeri belirlenmiştir. Ölçümler sonucunda ızgaraların mevcut açısal durumunun sıcaklık ölçümlerine doğrudan etki ettiği görülmüştür. Izgaranın tam olarak kapalı olması yani 0˚ ile tam açık olması yani 90˚ olması arasında yaklaşık olarak 4-5℃ sıcaklık farkının olduğu gözlemlenmiştir. Izgaraların belirli bir derece açık olması kabinet içi sıcaklığının optimum düzeyde tutulması açısından fayda sağlayacağı söylenebilir.

Yine aynı şartlar altında eş zamanlı olarak ortam sıcaklığı NTI cihazının yapmış olduğu ortam sıcaklığı ölçümü 29,2℃, Siemens marka sensör ile 26℃, Emerson marka iklimlendirme kliması ile 23℃ ve son olarak robot ile 23.38 derece ölçülmüştür. Sabit bir noktaya konumlandırılmış sıcaklık ve nem sensörlerinin doğru bir ölçüm yapılmasına olanak sağlamamaktadır. Gezgin robot ile hareketini gerçekleştirdiği rota boyunca farklı noktalarda ölçülen sıcaklık değerlerinin diğer sensörler gelen verilerden farklı olduğu görülmüştür. Fakat hasas iklimlendirme ünitelerinin gösterdiği sıcaklık değerine

benzerlik gösterdiği görülmüştür. Bu durumda robottan elde edilen sıcaklık değerlerinin diğer cihazlara kıyasla daha güvenilir sonuçlar sunduğu söylenebilir.

Son olarak sistem odasındaki izolasyon ile ortam sıcaklığının dış ortam sıcaklığına bağlı değişiminin kabul edilebilir derecede olduğu görülmüştür. Bu sebeple dış ortam sıcaklığının ortam sıcaklığına etki etmediği varsayılmıştır.

Çalışma kapsamında kullanılan Turtlebot 2 ‘nin boyutları ve kendi üzerinde bir işletim sisteminin bulunmamasından dolayı ölçümler yapılırken bir takım problemlerle karşılaşılmıştır. İşletim sistemi bulunmadığından robotun üzerine taşınabilir bilgisayar konumlandırılması gerekmektedir. Bu bilgisayar robot hareketiyle düşebilmektedir. İlerleyen süreçte TurtleBot 3’ün kullanımı bu problemleri ortadan kaldırarak daha verimli bir ortam oluşturmaya katkı sağlayacaktır. Ayrıca lazer sensörü yerine lidar sensörünün kullanılması harita oluşturma işleminde daha kesin sonuçlar alınabilmesine katkı sağlayacaktır.

KAYNAKLAR

Ackerman, E., 2012. https://spectrum.ieee.org/automaton/robotics/diy/turtlebot-2-prototypes-unveiled-at-roscon-2 (30/04/2019)

Akın, G., ve Ketenci, S. 2010. Sistem Odaları İklimlendirme Sistemleri. https://web.itu.edu.tr/akingok/doktora/iklimlendirme/Sistem_odasi_iklimlendirme_sistemleri.pdf (14/06/2019)

Alagöz, F., ve Çavdar, D., 2013. Yeşil Veri Merkezlerinde Enerji Verimliliği. 15. Akademik Bilişim Konferansı. Antalya.

Ali, A. 2018. Smart Fire Monitoring System Based On Internet of Things,Yüksek Lisans Tezi), Fen Bilimleri Enstitüsü, Kayseri.

Arduino, 2019. What is Arduino?. https://www.arduino.cc/en/Guide/Introduction (6/05/2019)

AWS RoboMaker, 2019. AWS RoboMaker Developer Guide. Seattle: Amazon Web Services. https://docs.aws.amazon.com/robomaker/latest/dg/simulation-tools-rviz.html (4/04/2019)

Ben-Ari, M., and Mondada, F., 2017. Elements of Robotics. Springer,Cham.

Bosch Sensortec GmbH. 2019. Bosch Sensortec: https://www.bosch-sensortec.com/bst/products/all_products/bme280 (12/05/2019)

Brock, O., and Khatib, O., 1999. High-speed Navigation Using the Global Dynamic Window Approach. Detroit: IEEE.

Buniyamin, N., Sariff, N., Ngah, W. J., and Mohamad, Z., 2011. Robot Global Path Planning Overview and A Variation of Ant Colony System Algorithm. British Columbia: Internatıonal Journal of Mathematıcs and Computers in Sımulatıon.

Ceyhan, A., 2019. Iot ile Bulanık Mantık Temelli Kestirimci Bakım Çözümüne Yönelik Sistem Tasarımı ve Uygulaması, Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü, Kayseri.

Claessens, R., Müller, Y., and Schnieders, B., 2013. Graph-based Simultaneous Localization and Mapping on the TurtleBot Platform. https://airesearch.de/written/Graph-Based-Simultaneous-Localization-and-Mapping-on-the-TurtleBot-Platform.pdf (22/04/2019)

Clearpath Robotics, 2014. Turtlebot 2. Ontario: Clearpath Robotics. https://www.clearpathrobotics.com/turtlebot-2-open-source-robot/ (2/05/2019)

Çakmak, F., 2014. Arama Kurtarma Amaçlı Diferansiyel Sürüşlü Tekerlekli Robot Tasarımı ve Gerçeklenmesi, Yüksek Lisans Tezi. Fen Bilimleri Enstitüsü, İstanbul.

Dattalo, A., 2018. ROS/Introduction. http://wiki.ros.org/ROS/Introduction (22/04/2019)

Dieter, F., Wolfram, B., and Sebastian, T., 1997. The Dynamic Window Approach to Collision Avoidance. IEEE Robotics and Automation Magazine.

Digitatek. 2018. Bilişim ve İletişim Teknolojileri Nedir?. https://www.digitatek.com/bilisim-ve-iletisim-teknolojileri-nedir-blog (19/04/2019)

Dunn, S., 2015. A*. https://github.com/libgdx/gdx-ai/wiki/A* (21/05/2019)

Elmas, Ç., 2003. Bulanık Mantık Denetleyiciler. Ankara: Seçkin Yayıncılık.

Ertuğrul, İ., 1996. Bulanık Mantık ve Bir Üretim Planlamasında Uygulama Örneği, Yüksek Lisans Tezi. Sosyal Bilimler Enstitüsü, Denizli.

Fairchild, C., and Harman, T. L. 2016., ROS Robotics By Example. Birmingham : Packt Publishing.

Fu, J., Wang, S., Lu, Y., Li, S., and Zeng, W. 2012., Kinect-Like Depth Denoising. South Korea: IEEE.

Gill, J. S., 2018. http://wiki.ros.org/navigation/Tutorials/RobotSetup#Costmap_Configuration_.28local_costmap.29_.26_.28global_costmap.29 (28/05/2019)

Goebel, P. R., 2012. ROS By Example A Do It Yourself Guide To The Robot Operating System . R. Patrick Goebel.

Harikrishnan, L., 2016. Using the low-level robot base controllers to drive the robot. http://wiki.ros.org/pr2_controllers/Tutorials/Using%20the%20robot%20base%20controllers%20to%20drive%20the%20robot (11/05/2019)

Joseph, L., and Cacace, J., 2018. Mastering ROS for Robotics Programming. Birmingham: Packt>.

Jusuf, F., 2016. Auto-Navigation For Robots. Implementation of ROS (Tez). Helsinki: Metropolia.

Kağızman, A., 2018. Otonom Araçlar İçin 2b Lazer Tarayıcı Kullanılarak Yeni 3b Lıdar Sistemi Elde Edilmesi ve Engel Tespiti, Yüksek Lisans Tezi. Fen Bilimleri Enstitüsü, İstanbul.

Kang, E., 2019. Amcl. http://wiki.ros.org/amcl (2/04/2019)

Kaya Sever, T., 2016. Whıte Space Cost Optımızatıon and Applıcatıon of Parametrıc Modelıng on Whıte Space Desıgn Improvement Process, Yüksek Lisans Tezi. Fen Bilimleri Enstitüsü, İstanbul.

Kshirsagar, S., and Shukla, K., 2010. A Study of Motion Planning Algorithms for Mobile Robots . International Journal on Computer Science and Engineering.

Lafcı, M., 2016. Dinamik Engellerin Bulunduğu Ortamda Gezgin Robot İçin Hareket Planlama, Yüksek Lisans Tezi. Fen Bilimleri Enstitüsü, Karabük.

Macenski, S., 2018. amcl. http://wiki.ros.org/amcl (12/06/2019)

Mansley, C., Connell, J., İşci, C., Lenchner, J., Kephart, J. O., McIntosh, S., and Schappert, M., 2011. Robotic Mapping and Monitoring of Data Centers. IEEE.

Mathworks, 2017. File Exchange. www.mathworks.com: https://www.mathworks.com/matlabcentral/fileexchange/64534-getting-started-with-robot-operating-system-ros (14/05/2019)

Moral, O., 2016. Yeni Nesil Veri Merkezi Tasarımı ve Kurulumu Aşamasındaki Gereksinimleri, Yüksek Lisans Tezi. Fen Bilimleri Enstitüsü, İstanbul.

Nagıp, G., and Gharıeb, W., 2004. Path Plannıng for A Mobıle Robot Usıng Genetıc Algorıthms. Accepted for Publication in International Conference on Electrical (s. 185-189). New York: IEEE.

Network Technologies Incorporated, 2019. Environment Monitoring System Temperature Sensors. www.networktechinc.com: http://www.networktechinc.com/temperature-sensor.html (23/06/2019)

Open Source Robotics Foundation, 2014. Why Gazebo?: http://gazebosim.org (3/05/2019)

Open Source Robotics Foundation, 2014. What is a TurtleBot. https://www.turtlebot.com/ (30/04/2019)

Open Source Robotics Foundation, 2018. costmap_2d: http://wiki.ros.org/costmap_2d (29/05/2019)

Open Source Robotics Foundation, 2019. Ubuntu install of ROS Indigo. http://wiki.ros.org/indigo/Installation/Ubuntu (6/05/2019)

Open Source Robotics Foundation, 2019. Carrot_planner. http://wiki.ros.org/carrot _planner (23/05/2019)

Open Source Robotocs Foundation, 2018. http://wiki.ros.org/global_planner: http://wiki.ros.org/global_planner (2/07/2019)

Özdemir, A., 2016. Mobil Robot Navigasyon Sistemi Geliştirmesi, Yüksek Lisans Tezi. Fen Bilimleri Enstitüsü, İstanbul.

Özek, A., and Sinecen, M., 2004. Klima Sistem Kontrolünün Bulanık Mantık ile Modellemesi. Mühendislik Bilimleri Dergisi, 353-358.

Özel, M., 2018. Robotik Biliminin Orta Okul 8. Sınıf Fen Bilimleri Dersine Entegrasyonu, Yüksek Lisans Tezi. Eğitim Bilimleri Enstitüsü, İstanbul.

Patterson, M. K., 2008. The Effect of Data Center Temperature on Energy Efficiency. Oregon: IEEE.

ROS.org., 2015. http://wiki.ros.org/ros_comm (23/04/2019)

Saday, F., 2019. Sığırlarda Bulanık Mantık Yaklaşımı İle Cinsiyet Belirleme Yöntemi Geliştirilmesi, Yüksek Lisans Tezi. Fen Bilimleri Enstitüsü, Konya.

Suvaydan, F., 2011. Mobil Robotlar İçin Yol Planlama Problemi Ve Karınca Kolonisi Algoritması İle Yol Planlama Problemlerinin Optimal Çözümü, Yüksek Lisans Tezi. Fen Bilimleri Enstitüsü, Düzce.

Şeker, Ş. E., 2009. A Yıldız Arama Algoritması. http://bilgisayarkavramlari. sadievrenseker.com/2009/03/02/a-yildiz-arama-algoritmasi-a-star-search-algorithm-a/ (1/04/2019)

Şen, Z., 2004. Mühendislikte Bulanık (FUZZY) Mantık İle Modelleme Prensipleri . İstanbul: Su Vakfı Yayınları.

Terun, B., 2011. Microsoft Kinect Nasıl Çalışır?.https://shiftdelete.net/microsoft-kinect-nasil-calisir-27482 (2/04/2019)

The Severn Group, 2017. What is the Proper Temperature for a Data Center?. https://www.theseverngroup.com/proper-temperature-for-a-data-center/ (26/06/2019)

Wan, J., Xiang, G., Kasahara, S., Zhang, Y., and Zhang, R., 2018. *Air Flow Measurement and Management for İmproving Cooling and Energy Efficiency in Raised-Floor Data Centers.* IEEE.

Wiring. Introduction. http://wiring.org.co/about.html (6/04/2019)

xTR Haber Merkezi. 2018. En Etkili Yıkıcı Teknolojiler: IoT, Yapay Zeka ve Robotik. https://www.xtrlarge.com/2018/11/23/yikici-teknoloji-iot-yapay-zeka-robotik/ (19/04/2019)

Yazıcı, A., 2016. Endüstri 4.0 ve Otonom Robotlar. http://www.emo.org.tr/ekler /91f2bb2a057879e_ek.pdf?dergi=1069 (17/04/2019)

Zheng, K., 2019. ROS Navigation Tuning Guide. New York: Cornell University.

ÖZGEÇMİŞ

1987 yılında Erzurum'da doğdu. İlk, orta ve lise öğrenimini Erzurum'da tamamladı. 2005–2010 yılları arasında Atılım Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği (İngilizce) Bölümünde lisans eğitimini tamamladı. 2011-2016 yılları arasında sırasıyla Barla Bilgisayar, Playture, Aşkale Çimento şirketlerinde çalıştı. 2016 yılında Atatürk Üniversitesi Bilgisayar Bilimleri Araştırma ve Uygulama Merkezinde Öğretim Görevlisi olarak göreve başladı. Halen Sistem Biriminde Sistem Yöneticisi olarak çalışmaktadır. Sunucu ve depolama ile sanallaştırma teknolojileri üzerine çalışmaktadır.