Bina İçi Ortamlarda Arama ve Kurtarma Görevleri İçin Seyrüsefer

Çağrı Mete Şenler

YÜKSEK LİSANS TEZİ

Elektrik - Elektronik Mühendisliği Anabilim Dalı

Nisan 2018

Navigation for Search and Rescue Missions in Indoor Environments

Çağrı Mete Şenler

MASTER OF SCIENCE THESIS

Department of Electrical - Electronics Engineering

April 2018

Bina İçi Ortamlarda Arama ve Kurtarma Görevleri İçin Seyrüsefer

Çağrı Mete Şenler

Eskişehir Osmangazi Üniversitesi
Fen Bilimleri Enstitüsü
Lisansüstü Yönetmeliği Uyarınca
Elektrik - Elektronik Mühendisliği Anabilim Dalı
Kontrol ve Kumanda Sistemleri Bilim Dalında
YÜKSEK LİSANS TEZİ
Olarak Hazırlanmıştır

Danışman: Prof.Dr. Osman PARLAKTUNA

Nisan 2018

ONAY

Elektrik - Elektronik Mühendisliği Anabilim Dalı YÜKSEK LİSANS öğrencisi Çağrı Mete Şenler'ın YÜKSEK LİSANS tezi olarak hazırladığı "**Bina İçi Ortamlarda Arama ve Kurtarma Görevleri İçin Seyrüsefer**" başlıklı bu çalışma, jürimizce lisansüstü yönetmeliğin ilgili maddeleri uyarınca değerlendirilerek oy birliği ile kabul edilmiştir.

Danışman : Prof.Dr. Osman PARLAKTUNA

Yüksek Lisans Tez Savunma Jürisi:

Üye : Prof.Dr. Osman PARLAKTUNA

Üye : Dr.Öğr.Üyesi Burak KALECİ

Üye : Dr.Öğr.Üyesi Hakan KORUL

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ..... tarih ve ..... sayılı kararıyla onaylanmıştır.

Prof.Dr. Hürriyet ERŞAHAN
Enstitü Müdürü

ETİK BEYAN

Eskişehir Osmangazi Üniversitesi Fen Bilimleri Enstitüsü tez yazım kılavuzuna göre, Prof.Dr. Osman PARLAKTUNA danışmanlığında hazırlamış olduğum "**Bina İçi Ortamlarda Arama ve Kurtarma Görevleri İçin Seyrüsefer**" başlıklı tezimin özgün bir çalışma olduğunu; tez çalışmamın tüm aşamalarında bilimsel etik ilke ve kurallara uygun davrandığımı; tezimde verdiğim bilgileri, verileri akademik ve bilimsel etik ilke ve kurallara uygun olarak elde ettiğimi; tez çalışmamda yararlandığım eserlerin tümüne atıf yaptığımı ve kaynak gösterdiğimi ve bilgi, belge ve sonuçları bilimsel etik ilke ve kurallara göre sunduğumu beyan ederim.

Çağrı Mete Şenler

ÖZET

Bu tez çalışmasında, bir deprem sonrası afet ortamı benzeri alanlarda arama-kurtarma görevlerinde kullanılabilecek bir veya birden çok gezgin robottan oluşan sistemler için bir bina içi seyrüsefer yaklaşımı tasarlanmıştır. Tasarlanan yöntemde, harici bir keşif erkininden, ortamın doluluk ızgarası olarak ifade edilen bir metrik haritasının elde edildiği varsayılmaktadır. Metrik haritadan, düğümler ve düğümleri birbirine bağlayan ayrıtlardan oluşan bir çizge ile belirtilen topolojik harita elde edilmektedir. Topolojik haritadaki düğümler, metrik haritadan spektral kümeleme algoritmasıyla elde edilirken, ayrıtlar en küçük kapsayan ağaç algoritması kullanılarak oluşturulmaktadır. Yol planı ise, elde edilen topolojik haritadaki düğümler ve ayrıtlar üzerine Dijkstra en kısa yol algoritmasının uygulanmasıyla oluşturulmaktadır. Bunlara ek olarak, afet ortamının engebeli bir zeminden oluşabileceği göz önüne alınarak, rgb-d kamerası tabanlı bir rampa algılama ve yolnoktası belirleme algoritması oluşturulmuştur. Ayrıca, çok robotlu bir sistemin kullanılması durumunda, robotların birbirleriyle çapışmalarının önlenmesi için bir çarpışma protokolü tasarlanmıştır.

Geliştirilen algoritmalar ROS (Robot Operating System - Robot İşletim Sistemi) kullanılarak, Gazebo benzetim ortamı üzerinde test edilmiştir. Benzetim ortamında platform olarak lazer mesafe algılayıcısı ve rgb-d (renk ve derinlik) kamerası ile donatılmış Pioneer P3-AT robotu kullanılmıştır. Ortam olarak ESOGÜ Elektrik-Elektronik Mühendisliği Laboratuvarı binasının ilk katı modellenmiş ve testlerde kullanılmıştır.

Anahtar Kelimeler: ROS, Gazebo, topolojik harita, Dijkstra, spektral kümeleme, rgb-d kamera, seyrüsefer, yol planlama, gezgin robot

SUMMARY

In this work, a single or multi-robot navigation system for search and rescue missions in indoor disaster environments is presented. It is assumed that a metric map consisting of an occupancy grid and a navigation target are received from an external exploration node that is assumed to be working in conjunction with the designed navigation node. After metric map is received, a topological map representation of the environment is created using the metric map. The created topological map is actually a graph which consist of nodes and arcs that connect these nodes. In order to construct the nodes of the topological map, spectral clustering method is utilized. For the purpose of creating arcs between nodes, minimum spanning tree algorithm is applied to the obtained nodes. For path planning purposes, Dijkstra's shortest path algorithm is used with the acquired topological map. In addition, since a disaster environment may have uneven ground surfaces, a rgb-d based ramp detection and waypoint extraction algorithm is proposed. Furthermore, for the case of multi-robot navigation, a collision avoidance algorithm is developed in order to prevent the robots from colliding with each other.

Designed algorithms are tested using ROS (Robot Operating System) with Gazebo simulation environment. Pioneer P3-AT robot equipped with a laser range scanner and a rgb-d camera is used as the test platform. First floor of ESOGU Electrical-Electronics Laboratory building is modelled in Gazebo simulation environment and used for testing purposes.

Keywords: ROS, Gazebo, topological map, Dijkstra, spectral clustering, rgb-d camera, navigation, path planning, mobile robot

TEŞEKKÜR

Yüksek lisans dönemim ve tez çalışmam boyunca, bana danışmanlık yapan, bilgisi ve deneyimiyle beni yönlendiren, yardım ve desteğini esirgemeyen değerli hocam ve danışmanım Prof.Dr.Osman PARLAKTUNA'ya en içten teşşekkürlerimi sunarım.

Hem tez çalışmam hem de derslerim ve projelerim kapsamında beni fikirleri ve yönlendirmeleriyle destekleyen Dr.Öğr.Üyesi Helin DUTAĞACI'na teşekkür ederim. Ayrıca, tez konusundaki çalışmalarım sırasında bana yardımcı olan ve çalıştığımız proje boyunca birlikte emek verdiğimiz Dr.Öğr.Üyesi Burak KALECİ'ye teşekkür ederim.

Yüksek lisans tez çalışmamın özellikle son zamanlarında hem lojistik olarak hem de değerli arkadaşlığı ile bana destek olan Halit ÇAKIR'a teşekkür ederim.

Son olarak, bu noktaya gelmemde büyük emekleri olan, karşılaştığım zorluklar karşısında bana her zaman yol gösteren ve arkamda duran, hiçbir yardımlarını esirgemeyen, başta çok sevdiğim ve saygı duyduğum annem Ayşe Gevher ŞENLER, babam Aydın Deniz ŞENLER ve biricik kardeşim Ece Asena ŞENLER olmak üzere, bütün aileme teşekkür ederim.

İÇİNDEKİLER

Sayfa

ÖZET . . . . . **vi**

SUMMARY . . . . . **vii**

TEŞEKKÜR . . . . . **viii**

İÇİNDEKİLER . . . . . **ix**

ŞEKİLLER DİZİNİ . . . . . **x**

1. GİRİŞ VE AMAÇ **1**

2. LİTERATÜR ARAŞTIRMASI **4**

3. MATERYAL VE YÖNTEM **11**
3.1. Robot Kontrol Çatısı 11
3.2. Benzetim Ortamı 12

4. BULGULAR VE TARTIŞMA **13**
4.1. Topolojik Harita Kullanılarak Dijkstra ile Seyrüsefer 13
4.1.1. Normalleştirilmiş Kesim Problemi 16
4.1.2. Topolojik Haritanın Oluşturulması 19
4.2. Vektör Alan Histogramı (VAH) 24
4.3. Çok Robotlu Seyrüsefer için Çarpışmadan Kaçınma 26
4.4. Rampa Algılama ve Hareket Düğümü Belirleme 35
4.4.1. İniş/Çıkış (Eğimli) Rampalarında Yolnoktası Oluşturma 47
4.4.2. Düz Rampalarda Yolnoktası Oluşturma 52
4.4.3. Yolnoktası Karar Verme Algoritması 55

5. SONUÇ VE ÖNERİLER **62**

KAYNAKLAR DİZİNİ **87**

ŞEKİLLER DİZİNİ

Şekil		Sayfa
1.1	Sistem Mimarisi	2
4.1	Çizge kesim örneği (Duong vd., 2014)	17
4.2	Metrik Harita	19
4.3	İlginlik Matrisi	19
4.4	D Matrisi	20
4.5	Örnek özvektörler	20
4.6	Uyarlamalı düğüm oluşturma örneği	22
4.7	Uyarlamalı yöntem ile elde edilen kümeleme sonuçları	25
4.8	Uyarlamalı yöntem ile elde edilen topolojik harita	25
4.9	Robotların aynı odaya girmek istediği an	28
4.10	Robotların aynı koridora çıkmak istediği an	28
4.11	Durum 1 Örneği	29
4.12	Durum 2 Örneği	29
4.13	Durum 1 Örneği için görsel sonuç	30
4.14	Durum 2 Örneği için görsel sonuç	30
4.15	Durum 3 Örneği	31
4.16	Durum 3 Örneği için görsel sonuç	31
4.17	Durum 4 Örneği	32
4.18	Durum 4 Örneği için görsel sonuç	32
4.19	Durum 5 Örneği	33
4.20	Durum 5 Örneği için görsel sonuç	33
4.21	Durum 6 Örneği	34
4.22	Durum 6 Örneği için görsel sonuç	34
4.23	Durum 7 Örnek 1	34
4.24	Durum 7 Örnek 2	35
4.25	Durum 7 Örnek 1 için görsel sonuç	35
4.26	Durum 7 Örnek 2 için görsel sonuç	35
4.27	Örnek Afet Ortamı Haritası	36
4.28	Voksel Izgarası Örneği (Wang vd., 2012)	43
4.29	Voksel Izgarası Uygulanmış Nokta Bulutu Örneği	43
4.30	Kamera Koordinat Sistemi - Robot Koordinat Sistemi Dönüşümü	44
4.31	Örnek Düzlem Bölütleme Gösterimi	45
4.32	Robotun Bakış Açısına Göre Rampaların Sınıflandırılması	46

ŞEKİLLER DİZİNİ (devam)

4.33 Egimli Rampa Üst-Alt Limitlerinin Belirlenmesi 47
4.34 Egimli Rampa Uçnoktalarının Belirlenmesi 48
4.35 Egimli Rampa Uçnoktalarının ve Yolnoktalarının Aykırı Nokta Düzeltme Algoritması ile Hizalanması 51
4.36 Çıkış Rampası Önü Yolnoktası 52
4.37 Düz Rampa Düzlemine Kümeleme Uygulanması 53
4.38 Kümelenmiş Düz Rampa Düzlemine Dışbükey Zarf Uygulanması ile Yolnoktası Elde Edilmesi 54
4.39 $rampaOnuYolnoktasiKontrolu$ Algoritmasıyla Rampa Çakışma Kontrolü . . 59
4.40 İniş rampasının dik gelmesine rağmen gidilebileceği durum 60

5.1 Test Ortamı 62
5.2 Çıkış Rampası Örneği 64
5.3 Düz Rampa Örneği 64
5.4 İniş Rampası Örneği 65
5.5 Tek robot ROS Düğümleri 65
5.6 Tek Robot Topolojik Harita - Kümeleme Adımları 1 66
5.7 Tek Robot Topolojik Harita - Kümeleme Adımları 2 67
5.8 Tek Robot Topolojik Harita - Kümeleme Adımları 3 68
5.9 Tek Robot Topolojik Harita - Düğüm Adımları 1 69
5.10 Tek Robot Topolojik Harita - Düğüm Adımları 2 70
5.11 Tek Robot Topolojik Harita - Düğüm Adımları 3 71
5.12 Tek robot testi sonucu elde edilen topolojik harita 73
5.13 Tek robot testi sonunda robotun izlediği yol 74
5.14 Çift robot testlerinde kullanılan gazebo test ortamı 74
5.15 Çift Robot ROS Düğümleri 75
5.16 İki Robot Topolojik Harita - Kümeleme Adımları 1 76
5.17 İki Robot Topolojik Harita - Kümeleme Adımları 2 77
5.18 İki Robot Topolojik Harita - Kümeleme Adımları 3 78
5.19 İki Robot Topolojik Harita - Kümeleme Adımları 4 79
5.20 İki Robot Topolojik Harita - Düğüm Adımları 1 80
5.21 İki Robot Topolojik Harita - Düğüm Adımları 2 81
5.22 İki Robot Topolojik Harita - Düğüm Adımları 3 82
5.23 İki Robot Topolojik Harita - Düğüm Adımları 4 83
5.24 İki robot testi ile oluşan topolojik harita 84
5.25 İki robot testi sonucunda robotların aldıkları yol 84

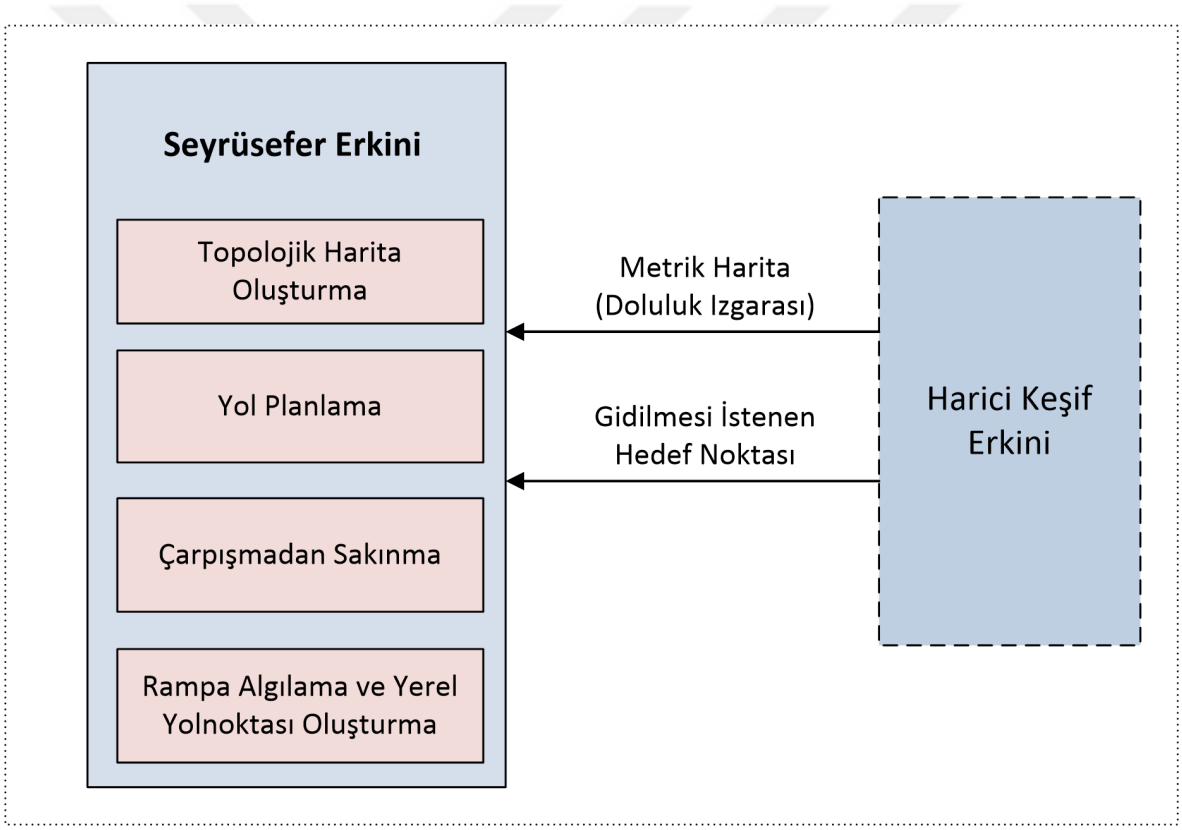
1. GİRİŞ VE AMAÇ

Arama-kurtarma, herhangi bir doğal veya insan kaynaklı afet esnasında acil yardıma ihtiyacı olan kimselerin yerini tespit etme, gerekirse ilk yardım uygulama ve daha kapsamlı yardım alabilecekleri güvenli bir yere nakletme faaliyetleri olarak tanımlanmaktadır. Türkiye, deprem, yangın, su baskını ve çeşitli nedenlerle oluşan patlamalar gibi felaketlerin sıklıkla yaşandığı bir ülke olduğundan, arama-kurtarma kapsamındaki görevler, insan ve diğer canlıların hayatının kurtarılabilmesi açısından büyük önem taşımaktadır. Deprem gibi afetler yaşandıktan sonra bina içerisinde devrilmiş kolonlar, çökmüş zeminler ve her yere dağılmış moloz yığınları bulunabilmektedir. Bununla birlikte, yapının içindeki elektrik, su ve gaz altyapıları hasar görmüş durumda olabileceğinden, bu gibi afet ortamları arama-kurtarma görevini yerine getiren insanlar ve arama-kurtarma köpekleri gibi hayvanlar için ayrı birer tehlike arz etmektedirler.

(Shah ve Choset, 2004)'de belirtildiği üzere, arama-kurtarma ekipleri, genellikle deprem gibi afetlerden sonraki 48 saatlik zaman diliminin kazazedelerin yerini tespit etmek ve kurtarmak için kritik önemde olduğuna değinirler. 48 saatten sonra afet ortamına maruz kalma, yemek, su ve tıbbi yardımın yetersizliği gibi nedenlerden ötürü ölüm oranı artış göstermektedir. Ancak Türkiye(1999), Tayvan(1999), New York(2001), İran(2003) gibi can kaybının yüksek olduğu afetlerde, arama-kurtarma için kullanılan kaynakların afet ortamındaki yıkımın büyüklüğü karşısında 48 saat içerisinde herkesi kurtarmaya yetmesinin mümkün olmadığı görülmüştür. Bu nedenlerle bina içi afet ortamlarında robotların kullanılması, can kayıplarını azaltabilecek bir çözüm olarak karşımıza çıkmaktadır. Robotların kullanılmasıyla, arama-kurtarma görevlilerinin maruz kaldığı hayati tehlikeler azalırken, hem insanların erişemeyeceği dar alanlara ulaşmak, hem de arama-kurtarma görevlerinin daha verimli, güvenilir ve hızlı olarak yapılabilmesi mümkün olabilmektedir.

(Greer vd., 2002)'deki çalışmada belirtildiği üzere, 1995 Oklohoma Şehri Bombalaması ve Japonya'da gerçekleşen Kobe depremi gibi afetlerden sonra arama-kurtarma alanında robotların kullanılmasıyla ilgili çalışmalarda artış görülmüştür. 2001 yılındaki Dünya Ticaret Merkezi'nin yıkılışında ilk defa robotların arama-kurtarma görevi için kullanıldığı belirtilmektedir. Burada elde edilen deneyimler sonucunda robotların otonom hareket etmesinin ve afet ortamındaki zeminlerde ilerleyebilecek şekilde hareket kabiliyetlerinin geliştirilmesinin arama ve kurtarma görevinin verimi açısından önemli olduğu belirlenmiştir.

Bu tez çalışması, Eskişehir Osmangazi Üniversitesi Bilimsel Araştırma Projeleri Komisyonu tarafından desteklenen ”Deprem Sonrası Bina İçi Ortamlarda Kara ve Hava Robotları Kullanılarak Keşif ve Haritalama (Proje No:201415012(2013-253))” isimli araştırma projesi için yapılan çalışmalar kapsamında oluşturulmuştur. Tez çalışmasının amacı, bir bina içi arama ve kurtarma görevinin gerçekleştirilebilmesi için, bir veya birden çok gezgin yer robotunun yol planlamasının yapılarak seyrüseferinin gerçekleştirilmesinin sağlanmasıdır. Çalışma sırasında, kullanılan robotun 360 derece tarama açısına sahip bir lazer mesafe algılayıcısı ile bir rgb-d (renk ve derinlik) kamerasına sahip olduğu, buna ek olarak yol planlaması ve seyrüsefer görevlerini yerine getirebilecek hesaplama kapasitesinde donanımının bulunduğu varsayılmıştır. Çalışmanın kapsamının daha iyi anlaşılabilmesi için kullanılan mimari Şekil 1.1’de gösterilmektedir.



Şekil 1.1: Sistem Mimarisi

Şekil 1.1’den de anlaşılacağı üzere, bu tez çalışmasında gerçeklenen seyrüsefer erkinine her aşamada harici bir keşif erkininden ortamın o anda keşfedilen bölgelerini ifade eden ve doluluk ızgarasıyla gösterilen bir metrik harita ile robotun gitmesinin istendiği hedef konumu gelmektedir. Harici keşif erkininin çalışması bu tez çalışması kapsamında anlatılmayacak olup, detaylı bir açıklaması için (Kaleci, 2016)’deki çalışma incelenebilir .

Tez çalışmasının takip eden bölümlerinde, ilk olarak Bölüm 2'de afet ortamında seyrüsefer ile ilgili bazı çalışmaların incelendiği literatür taraması verilecektir. Bölüm 3'te kullanılan afet ortamından bahsedilecek ve çalışma sırasında faydalanılan benzetim ortamı ile robot kontrol çatısı(framework) gibi programlara değinilecektir. Bölüm 4'te ilk olarak harici erkinden gelen veriler kullanılarak bir topolojik harita oluşturma yaklaşımından bahsedilecektir. İkinci aşamada birden fazla robotun kullanılması durumunda robotların birbirlerine çarpışmasının önlenmesi için tasarlanan bir çarpışma protokolüne değinilecektir. Sonraki aşamada afet ortamında bulunan rampaların saptanması ve bu rampaların robotun seyrüseferi için nasıl kullanıldığı açıklanacaktır. Bölüm 5'te kullanılan yöntemler ile elde edilen sonuçlar yorumlanacak ve ileriki çalışmalar için izlenebilecek olası yollara değinilecektir.

2. LİTERATÜR ARAŞTIRMASI

Bir mobil robotun, arama ve kurtarma görevlerini yerine getirebilmesi için, afet ortamında seyrüsefer görevini yerine getirebilmesi gerekmektedir. Afet ortamları, yapısal olarak düzensiz, dağınık ve kestirilemez olmaları gibi nedenlerden ötürü seyrüsefer görevinin yerine getirilmesini bir hayli zorlaştırmaktadır. Bu nedenlerle bu alandaki ilk çalışmalarda robotların kontrolü operatörler tarafından yapılmıştır (Murphy, 2004). (Murphy ve Burke, 2005) ve (Casper ve Murphy, 2003)'deki çalışmalarda, insan-robot etkileşiminin kullanıldığı durumlarda karşılaşılan problemlerden ve olası çözüm önerilerinden bahsedilmiştir. Robotların operatör kontrolünde kullanıldığı durumlarda, afet ortamlarının karmaşık yapısından dolayı, insanların robottan gelen verileri anlamlandırabilmesi zor olmakta, bu durumda da robotlar sıkışıp kalabilmektedir. Bu gibi nedenlerden ötürü, insan etkileşiminin minimuma indirilmesi ve robotların otonom özellik kazanması, arama ve kurtarma görevlerinin daha yüksek verimle yapılabilmesini sağlayacaktır. Bu doğrultuda, son yıllarda robotlara farklı seviyelerde otonomluk kazandırabilecek çalışmalar yapılmaktadır. (Liu ve Nejat, 2013)'e göre, bazı çalışmacılar robotların düşük seviye kontrolünü geliştirmek amacıyla, otonom olarak merdiven tırmanma ve düzgün olmayan zeminde hareket edebilme gibi alanlara yönelirken, bazı çalışmacılar da ortamın keşfediilmesi ve kazazedelerin tanınması gibi yüksek seviye kontrol görevlerinin geliştirilmesine yoğunlaşmışlardır. Bu görevler ile de operatörün işyükünün azaltılmasını hedeflemişlerdir.

(Doroodgar vd., 2010)'daki çalışmada, durumsal farkındalığın önemine değinilmiştir. Durumsal farkındalık, bir zaman ve uzay diliminde bulunan ortamdaki nesnelerin algılanması, bu nesnelerin anlamlandırılması ve durumlarının yakın gelecekteki izdüşümlerinin bulunması olarak tanımlanmaktadır. Operatör kontrolündeki robotlar kullanılarak yapılan arama ve kurtarma çalışmalarında, operatörlerin bahsi geçen durumsal farkındalığı sağlayabilmek için o anki yaptıkları işleri durdurduklarını ve zamanlarının %30'unu bu işe harcadıkları belirtilmektedir. Ayrıca, arama ve kurtarma ortamlarında, operatörlerin çabucak stres altında kalabildikleri ve yorulabildikleri belirtilmiştir. Çalışmada, bahsedilen sorunlara karşı bir çözüm olarak, hiyerarşik pekiştirmeli öğrenme (hierarchical reinforcement learning, HRL) tabanlı bir kontrol algoritması önerilmiştir. Bu algoritma ile robota öğrenme ve karar verme yetisi verilerek, belirli bir zamanda hangi görevlerin yerine getirilebileceğine ve en uygun sonuç için bu görevlerin insan tarafından mı yoksa robot tarafından mı yapılabileceğine karar vermesi amaçlanmıştır.

Bazı çalışmalarda robotun verimli olarak arama kurtarma görevini gerçekleştirebilmesi için, operatörün robotun otonomluk seviyesini değiştirebildiği kontrol yöntemleri de önerilmiştir. (Bruemmer vd., 2002)'deki çalışmada, manuel kontrol, güvenli mod, paylaşımlı kontrol ve tam otonom olmak üzere 4 modlu bir sistem mimarisi kullanılmıştır. Manuel kontrol modunda, robotun kontrolü tamamen kullanıcı tarafından sağlanmaktadır. Robot sadece haberleşme koparsa otomatik olarak durmaktadır. Güvenli modda, robotun hareketi kullanıcı tarafından sağlanmakta ancak robot kendini koruması gerektiğinde kontrolü ele almaktadır. Robot bu modda iken gerçekleştiremeyeceğine inandığı görevi yapmayı reddetmektedir. Paylaşımlı kontrol modunda, robot yerel olarak kendi yolunu seçebilmekte ve çevresine otonom olarak tepki verebilmektedir. Ancak hareket etmesi gereken genel yönelimi kullanıcıdan istemesi gerekmektedir. Tam otonom modda ise global olarak yol planlaması yapmakta ve gideceği yolları kendisi seçebilmektedir.

(Wegner ve Anderson, 2006)' da arama ve kurtarma alanındaki çalışmaların çoğunluğunda robotların el ile bir operatör tarafından kontrol edildiğine değinilmiştir. Ancak durumsal farkındalığın kaybedilmesi, operatörlerin yorulması ve stres altında kalması gibi nedenlerle bu yöntemlerin pek verimli olmadığından bahsedilmiştir. Çözüm olarak otonom ve manuel kontrolün birleştirildiği bir yöntem önermişlerdir. Geliştirdikleri yöntemde, her robotta çalışacak şekilde tasarlanmış iki tip erkin bulunmaktadır. Bunlardan ilki robotun otonom işlemleri ile operatörden gelen komutları kaynaştıran arabuluculuk erkini, diğeri de operatörün uyarılması gereken durumları algılayıp operatöre haber veren müdahale erkini olarak tanımlanmıştır. Arabuluculuk erkini gelen komutları değerlendirebilme yetisine sahiptir. Gelen komutlar harfi harfine uygulayabilmekle birlilkte, bazı komutlar robotun otonom süreci ile birleştirilebilir veya robota hasar verebilecek bir durum algılanırsa o komutu reddedilebilmektedir. Müdahale erkini ise robotun algılayıcılarından gelen bilgileri kullanarak çevresini analiz etmekte, operatör müdahalesine gerek olabilecek durumları belirlemekte ve bu durumları robotun müdahale olmadan görevine devam edebileceği durumlardan ayırmaktadır. Çalışmada, bir arama kurtarma görevi sırasında müdahalenin gerekebileceği üç durum belirlenmiştir. Bunlardan ilki, robotun hareketi sırasında sıkışmasıdır. Robotun eyleyicilerine komut gitmesine rağmen, robotun algılayıcısında gelen bilgilere göre hareket algılanmıyorsa, robot sıkışmıştır denilmektedir. İkinci durum ise robotun ortamda kaybolması durumudur. Robotun ortamı tanıması bazı karakteristik noktaların algılanması ile sağlanmaktadır. Robot bu noktaları kaç kere algıladığını ve her algılama arasında geçen süreleri hafızasında tutmaktadır. Böylece robotun belirli bir alanda kaybolup kaybolmadığı anlaşılmakta ve operatöre bilgi verilmektedir. Son durum ise ortamdaki kazazedelerin tanınması görevidir. Çalışmanın sonucunda, otonom ve manuel kontrolün birleştirilmesi ile arama ve kurtarma görevinin daha verimli yapılabileceği belirtilmiştir.

Son yıllarda, el ile operasyon ve yarı-otonom operasyon yöntemlerinin yanı sıra, robotların tam otonom olarak hareket ederek arama ve kurtarma görevlerini yerine getirebilmesi amacıyla yapılan çalışmalara daha sık rastlanmaktadır. (Calisi vd., 2007)'daki çalışmada, arama ve kurtarma robotları için, yapılandırılmamış içortamlar için otonom bir arama ve keşif yöntemi düşünülmüştür. Amaç, operatöre iletilmek üzere arama-kurtarma göreviyle ilgili bilgi ararken, ortamın keşfedilmesi ve haritalandırılması olarak tanımlanmış; bu gibi amaçlar ile ilgili çalışmaların genellikle ortamdaki ilgi çekici özelliklerin tanınması ve analiz edilmesi üzerine yoğunlaştığı belirtilmiştir. Çalışmada düşünülen yöntemin sonucunda tanımlanan görevin gerçeklenmesi ile ortamın ilgi çekici unsurlar(örneğin kazazedeler) ile işaretlenmiş 2 boyutlu bir haritasının elde edilmesi planlanmıştır. Ortam, sabit olarak kabul edilmiştir. Önerilen yöntemi, arama ve keşfetme, SLAM(eşzamanlı konumlandırma ve haritalama), seyrüsefer ve insan vücudunun tanınması olarak dört ana modül ile göstermişlerdir. Seyrüsefer modülünde, üst seviyede topolojik bir yol planı oluşturulurken, alt seviyede Rastgele Kinodinamik Ağaçlar (Randomized Kinodynamic Trees) algoritması tabanlı bir hareket planlayıcı ve Vektör Alan Histogramı (Vector Field Histogram) tabanlı bir seyrüsefer modülü kullanmışlardır.

(Calisi vd., 2005)'de arama ve kurtarma ortamı için otonom bir seyrüsefer ve keşif yöntemi önerilmiştir. Son zamanlarda, klasik deterministik ve geometrik yöntemlerdeki hesaplama yüklerini düşürebilmek amacıyla olasılıksal yöntemlerin robot hareket kontrolünde kullanılmaya başlandığını belirtmişlerdir. Bu olasılıksal algoritmalara örnek olarak Olasılıksal Yol Haritası (Probabilistic RoadMap) ve Hızlı-keşfeden Rastgele Ağaçlar (rapid exploring random trees-RRT) algoritmalarını göstermişlerdir. Önerdikleri yöntemde de, bu algoritmalara benzer olarak, yerel haritayı ve algılayıcı okumalarını kullanan bir yerel planlayıcı ve bütün ortamı hesaba katan bir global planlayıcı kullanmışlardır. Küresel yol planlayıcı için Olasılıksal Yol Haritası ve Genişleyen Sinirsel Gaz (Growing Neural Gas) algoritmalarını, yerel hareket planlaması için de Rastgele Kinodinamik Planlama (Randomized Kinodynamic Planning) algoritmalarını kullanmışlardır.

(Zhao vd., 2009)'daki çalışmada, maden ortamlarında oluşabilecek afet durumlarına karşı bir yol planlama yöntemi önerilmiştir. Çalışmada, geliştirilmiş bir genetik algoritma benimsenmiştir. Genetik algoritmanın toptan hesaplama, yavaş hızda çalışması, zamanından önce yerel optimum yol planlarının oluşması gibi zayıflıklarının üstesinden gelmek için ilk popülasyonun oluşturulmasında Karınca Kolonisi Optimizasyonu ve öncelik gruplandırmasından yararlanılmıştır. Afet ortamının temsilinde 2 boyutlu ızgara modelini kullanmışlardır. Sundukları algoritmanın gerçeklenmesinde seçim, genetik değişim, mutasyon ve silme olmak üzere dört tane geleneksel genetik operatör kullanmışlardır. (Zhang vd., 2009)'daki çalışmada da Karınca Kolonisi Optimizasyonu algoritması kullanılarak arama ve kurtarma görevi için global bir yol planlama yöntemi gerçeklenmiştir.

(Baltes ve Anderson, 2003)'de belirsiz dinamik afet ortamları için bir yol planlama yöntemi önerilmiştir. Yol planlama yöntemleri temel olarak Görünürlük Çizgesi, Voronoi Diyagramları, Yol Haritaları gibi iskeletleştirme yöntemleri, dörtlü-ağaç ayrışması gibi hücre ayrıştırma yöntemleri ve potansiyel alanlar gibi yerel yöntemler olarak üç sınıfa ayrılmıştır. Görünürlük Çizgesi gibi iskeletleştirme yöntemlerinde duvar ve engellere yakın hareket etmenin çarpışmalara neden olabileceği, potansiyel alanlar gibi yerel yöntemlerde de yerel maxima/minima noktalarından robotun kurtulabilmesi için ek algoritmalar kullanılmasının gerekliliği gibi dezavantajlardan bahsetmişlerdir. Dörtlü-ağaç gibi hücre ayrışması yöntemleri ile yol planlamanın basit ve ortam tamamen ayrıştıktan sonra verimli olduğu belirtilmiş, ancak karmaşıklığının yüksek olması nedeniyle yol planlama yöntemlerinde pek kullanılamadığına değinmişlerdir. Ayrıca yol planlamanın yapılabilmesi için ortamın ayrışmasının bitmesinin gerekliliği de bir dezavantaj olarak belirtilmiştir. Hücre ayrışmasındaki bahsi geçen problemlerin çözümü için, ayrıştırma işlemini açık alanları çabucak tanıyabilecek ve bu alanları ağacın bağımsız bölgelerinde oluşturmak yerine birlikte gruplayabilecek, bu işlerin yapılmasını da hesaplama açısından mümkün kılabilecek bir yönteme ihtiyaç duyulduğu belirtilmiştir. Bunun için de ikili alan ayrıştırma algoritmasını kullanan bir yöntem geliştirmişlerdir.

(Colas vd., 2013)'de arama ve kurtarma görevlerinde karşılaşılan karmaşık ortamlar için gürültülü algılayıcı bilgilerini kullanarak yol planlaması yapan bir yöntem önerilmiştir. Afet ortamlarında yol planlaması için sıklıkla kullanılan A*, Dijkstra gibi yöntemlerin ortamdaki boş alanların tanımlanmasına bağlı olarak 2 boyutlu ortamlarda çalıştığına değinilmiştir. Ancak, yükseltilerin eklendiği 2.5 boyutlu olarak tanımlanan ortamlarda bu yöntemlerin , ortamın birden fazla kattan ve merdivenden oluşan katmanlı yapısından ötürü, çalışmasının mümkün olmayacağına değinilmiştir. Robotun bir alanda engel olmadığı için orayı boş alan olarak tanımlayıp yol planı yapabileceği, ancak engel olmamasına rağmen bir boşluk var ise robotun düşmesine sebep olabileceğinden bahsedilmiştir. Çalışmada, algılayıcı verisi olarak nokta kümeleri kullanılmıştır. Genel işleyiş iki aşamada anlatılmıştır: belirtilmiş herhangi bir konumun engelsiz olmasının, yeterli desteğe sahip olmasının ve yuvarlanma ile yunuslama açılarının sınırları dahilinde olmasının belirlenmesi ve iki komşu konum arasından geçilip geçilemeyeceğinin belirlenmesi. Bir ortamdan geçilip geçilemeyeceğinin belirlenmesi için gereken geometrik bilginin elde edilmesinde tembel tensör oylama yöntemi kullanılmıştır. Ayrıca karmaşıklığın düşürülmesi için hem nokta sayısı hem de yoğunluk birbirine çok yakın noktalar filtrelenerek düşürülmüştür. Temelde yol planlama için D* algoritması kullanılmıştır. Ortamdaki konumların bir çizge ile bağlanması için de konumların 26 komşuluğuna bakılmıştır. Yapılan testler sonucunda robotun önerilen yöntem ile 2 boyutlu olarak yol seyrüsefer yapabildiği ve 3 boyutlu bir merdiveni tırmanıp inebildiğini gözlemlemişlerdir.

Robotlar yardımıyla arama ve kurtarma görevi için yeni yöntemlerin geliştirilebilmesi amacıyla çeşitli yarışmalar da düzenlenilmektedir. Bunlardan en önemlilerinden birisi Robocup Kurtarma Yarışmalarıdır(*Robocup Rescue Competitions Description*). İlk Robocup yarışmaları ve konferansları 1997 yılında düzenlenmiştir. Bu yarışmaların temel amacı robotlarla oluşturulmuş bir futbol takımının insanlardan oluşan bir futbol takımını yenebilmesidir. 2001 yılında ise Robocup yarışmalarına kurtarma görevleri de eklenmiştir (Nagatani vd., 2011). (Akin vd., 2012)'ye göre, Robocup kurtarma yarışmalarının amacı, gerçekçi olarak benzetilmiş afet ortamlarında kurtarma görevleri için robot takımlarının koordinasyonunu ve kontrolünü sağlayan algoritmaların performansının karşılaştırılmasıdır.

(Nagatani vd., 2011)'deki çalışmada, bir afet durumunda mobil robotlar yardımıyla ortamdan hızlı biçimde veri toplanabilmesi görevi için, üç boyutlu haritalandırmanın önemli bir aşama olduğuna değinilmiştir. Üç boyutlu haritalamanın gerçekleştirilebilmesi için, düzgün olmayan zeminde hareket etme, sürekli olarak ortamdan üç boyutlu veri toplama, kapsamlı yol planlama, farklı robotlardan gelen haritaların merkezileştirilmesi ve harita verilerinin birleştirilmesi olmak üzere çözülmesi gereken beş tane problem saptamışlardır ve her bir problem için bir çözüm yöntemi uygulamışlardır. Ortamın ifade edilmesi için dijital yükselti haritası(digital elevation map-DEM) kullanmışlardır. Anlık olarak elde edilen her yerel DEM, artırımlı en yakın nokta(iterative closest point-ICP) algoritmasıyla duvarları ifade eden verilerin düzeltilmesi ile birbiriyle eşleştirilmiş ve küresel bir DEM elde edilmiştir. Elde ettikleri küresel dijital yükselti haritasını kullanarak da kapsamlı yol planlaması yapmışladır. Yol planlaması için, küresel dijital yükselti haritasına bölütleme uygulayarak, haritadaki hücreleri boş alan ve engel bulunan alan olarak sınıflandırmışlardır. Bölütlenmiş harita üzerinde sınır hücresi algoritması ile gidilebilecek bölgeler belirlenmiştir. Bu sınır bölgelerinden robota en yakın olanını da alt-hedef olarak seçilmiş ve bu bölgeye olan en kısa yolun bulunması için Dijkstra En Kısa Yol Algoritması kullanılmıştır. Böylelikle alt-hedefe olan en kısa yolu belirten hücre tabanlı bir yol planı elde edilmiş olmaktadır.

(Balakirsky vd., 2007)'de ilk RoboCup Rescue Virtual yarışmasında kullanılan yöntemlerden bahsedilmektedir. "Rescue Robots Freiburg" takımı, bilinmeyen ortamların keşfedilmesi ve kazazede konumlarıyla güçlendirilmiş bir topolojik haritasının oluşturulmasını amaçlamışlardır. Çalışmalarının seyrüsefer ve keşif bölümündeki genel düşünceleri, robotların ortamın kendilerine göre olan yerel görüşlerini kullanarak keşif ve yol planlaması yapmalarıdır. Robotların yerel görüşlerinin tutarlı olması için de birbirleriyle RFID etiketleri vasıtasıyla dolaylı olarak gerçekleşen haberleşmelerinden faydalanmaları düşünülmüştür. Ortamın ifadesi için boyutu hızlı hesaplamaya uygun olacak şekilde belirlenen doluluk ızgarasından faydalanmışlardır. Keşif işlemi ile sürekli olarak belirli

periyotlarda hedefler seçilmiş ve bu hedeflere yol planlarının oluşturulması için ızgara hücrelerini kullanan A* algoritması kullanılmıştır. Robotların birbirleriyle çarpışmasının önlenmesi için de, RFID etiketlerinden faydalanılmıştır. Her robot, doluluk ızgarası limiti içerisinde algılanan bütün RFID'leri kendi yerel RFID serisinde tutmaktadır ve ortamdaki hareketi süresince kendi RFID'lerini yayınlamaktadır. Böylelikle, robotlar bu RFID etiketlerini kullanarak ortamdaki robotların kendilerine göre bağıl olarak yerdeğiştirmelerini ve dolayısıyla bağıl konumlarını hesaplayabilmektedirler. Bu bilgi de doluluk ızgarasındaki hücrelerin etiketlenmesinde kullanılmakta, bu şekilde de elde edilecek yol planlarının çarpışmasız bir biçimde oluşması sağlanmaktadır. "Virtual IUB" takımı, robotların birbirleriyle koordine hareket etmek yerine tek başlarınaymış gibi davranarak hareket ettikleri bir sistem tasarlamışlardır. Robotların elde ettikleri kısmi bilgilerin birleştirilmesi görev tamamlanınca gerçekleşmektedir. Takımın amacının, yeni algoritmik yöntemlerin geliştirilmesi yerine bütünleşik bir sistemin gerçeklenmesi olduğu belirtilmiştir. Kullandıkları sistemdeki seyrüsefer bloğunun görevinin temelde ortamda çarpışmasız bir şekilde hareketin sağlanması olduğunu belirtmişlerdir. Bu görevin yerine getirilebilmesi için, bir engele bir eşik değerden fazla yaklaşılırsa bir itici güç yardımıyla engelden kaçınmayı amaçlamışlardır ve bu amaç doğrultusunda ses üstü algılayıcılardan faydalanmışlardır. Haritalandırma görevi için ise bir çeşit SLAM (eşzamanlı konumlandırma ve haritalama) algoritması kullanmışlardır. "UVA Rescue" takımı, diğer takımlardan farklı olarak bir kurtarma durumundaki çok robotlu sistemler için eşzamanlı konumlandırma ve haritalama(SLAM) problemini çözmeye odaklanmışlardır. Seyrüsefer ve keşif için karmaşık bir algoritma yerine bir dizi küçük çaplı tepkisel davranış kullanmışlardır. Davranışların arasındaki geçiş için bir durum makinesinden (state machine) faydalanmışlardır. Kullandıkları yöntemde, robot mümkün olduğu kadar düz bir doğrultuda hedefine doğru hareket etmekte, bir engelle karşılaşırsa ise rastgele olarak sola veya sağa dönüp hareketine devam etmektedir. Bu davranış ile koridor ve açık alanlar gibi ortamlardaki uzun mesafelerin daha çabuk katedilebildiğini belirtmişlerdir. Ayrıca rastgele davranış nedeniyle de robotların birbirleriyle koordinasyonu olmadan ortama yayılabildiklerine değinilmiştir. Konumlandırma ve haritalama aşamasında robotun elde ettiği lazer taramalarının birbirleriyle eşleşmesinin kontrolü yapılmıştır. Güncel lazer taramasıyla belirli bir zaman önceki lazer taramasının karşılaştırılması yapılmış, elde edilen yerdeğiştirme miktarı robotun güven derecesini düşürecek kadar büyüdüğü zaman ortamda yeni bir düğüm yaratılarak haritanın yeni bir bölümü öğrenilmiş olmaktadır. Düğümlerden oluşan çizge tabanlı bir harita gösteriminin kullanılmasının, hafızada birden fazla birbirinden bağımsız haritanın tutulmasını sağladığı belirtilmiştir. Bu haritalar arasında üst üste binen bölümler olduğunda, çakışan kısımlardaki birbirine yakın noktaların lazer okumaları karşılaştırılmış ve birbirlerine yeteri kadar yakınlarsa bir kapalı devre(loop closure) yöntemiyle her iki haritadaki noktalar birleştirilerek kesinlik derecesi artırılmıştır. "STEEL" takımı, insan-makine etkileşimli bir sistem olarak Machinetta vekilleri yöntemine

dayalı bir yöntem kullanmışlardır. Kullanıkları yöntemde, pek çok koordinasyon kararı vekiller tarafından alınırken, anahtar kararlar operatöre bırakılmaktadır. Vekillerden oluşan takımlar, ortak bir amaç doğrultusunda yapılacak olan bireysel görevleri ve bunlar arasındaki kısıtlamaları ifade eden takım tabanlı planları (TTP) gerçekleştirmektedirler. TTP'ler, çalışma esnasında belirli şartlar sağlandığı zaman dinamik olarak belirlenirler. Bir insan-makine etkileşimli sistem olmasında ötürü, STEEL takımının seyrüsefer ve keşif görevlerini de diğer takımlara göre farklı bir şekilde gerçekleştirdiğine değinilmiştir. Düşük seviye kontrol aşamasında, robotların hareketi algılayıcı verileri göz önüne alınarak Machinetta tarafından sağlanmaktadır. Ara seviye kontrol aşamasında, kullanıcı robotun ve robotun bulunduğu ortamın durumunu kontrol ederek gerekirse güvenli bir şekilde hareketine devam edebilmesi için robota müdahale edebilmektedir. Ayrıca robot boş durumdayken, kullanıcı potansiyel keşif planları oluşturabilmektedir. Buna ek olarak, bir kazazedenin algılandığı durumlarda da kullanıcının kazazedeyi inceleyecek şekilde bir plan oluşturması mümkün olmaktadır. Bu aşamada, oluşturulan planın gerçeklenmesinin Machinetta vekili tarafından TTP'nin tetiklenmesiyle gerçekleştiği belirtilmiştir. Sonuç olarak bu tasarımda, insan müdahelesi sadece gerekli durumlarda gerçekleşecek şekilde, insan ve robotun birlikte çalışması sağlanmış olmaktadır.

Bu tez çalışması kapsamında, bilinmeyen bir afet ortamında bir veya birden fazla gezgin robotun arama-kurtarma görevini gerçekleştirebilmesi amacıyla bir seyrüsefer sisteminin gerçeklenmesi hedeflenmiştir. Bu amaç doğrultusunda çeşitli çalışmalar incelenmiş, robotun otonom şekilde hareketini sağlayabileceği bir takım yöntemin kullanılmasına karar verilmiştir. İlk olarak harici keşif erkininden gelen verilerin kullanılarak bir topolojik haritanın oluşturulması üzerinde durulmuş ve bunun sonucunda ortam robotun hareketinin her aşamasında genişleyen bir topolojik harita ile ifade edilmiştir. Bu topolojik harita ile sıklıkla kullanılan yol planlama algoritmalarından birisi olan Dijkstra en kısa yol algoritması gerçeklenerek robotun bulunduğu konumdan hedef konumuna hareketi sağlanmıştır. Bunlara ek olarak robotun duvarlara çarpmadan hareket edebilmesi için Vektör Alan Histogramı (VFH - Vector Field Histogram) algoritması kullanılmıştır. Ayrıca, birden fazla robotun kullanılması durumunda, çarpışmanın önlenmesi için bir protokol tasarlanmıştır. Son olarak, robotun afet ortamındaki rampalar göz önünde bulundurularak hareketinin sağlanabilmesi için oluşturulan rgb-d kamerası tabanlı bir yöntem de bahsedilen diğer yöntemlerle birlikte yol planlama ve seyrüsefer için kullanılmıştır.

3. MATERYAL VE YÖNTEM

Bu bölümde yapılan çalışmalarda kullanılan benzetim ortamından ve robot kontrol çatısından kısaca bahsedilecektir.

3.1 Robot Kontrol Çatısı

Robotların otonom olarak kendilerine verilen görevleri yapabilmeleri için, algılayıcıları vasıtasıyla ortam hakkında edindikleri bilgileri değerlendirerek bir takım kararlar vermeleri ve bu kararlar doğrultusunda uygulamaları gerçekleştirebilmeleri gerekmektedir. Bu karar verme ve uygulama adımları ile robotun hareketi sonucu ortamdan alınan bilgiler değişecek ve her adımda bu işlemler tekrarlanacaktır. Bu karar verme ve uygulama işlemlerinin yapılabilmesi için bir robot kontrol çatısına ihtiyaç duyulmaktadır.

Robot kontrol çatılarına popüler bir örnek olarak ROS (Robot Operating System) verilebilir. ROS robot kontrol yazılımlarının geliştirilebilmesi için kullanılabilen esnek bir robot kontrol çatısıdır. Bünyesinde geniş bir yelpazedeki robot platformları için karmaşık ve gürbüz programların yazılabilmesini kolaylaştırmak amacıyla pek çok araç ve kütüphane barındırmaktadır. Bu araç ve kütüphaneler topluluğunun oluşturulmasındaki amaç, insanlar için kolay görünen görevlerin robot açısından bakıldığında çok karmaşık hal almaları, ayrıca farklı ortam ve durumlarda çok değişken sonuçlar elde edilmesinden ötürü gürbüz bir tasarım oluşturmanın zor olması olarak açıklanmaktadır. Sonuç olarak, ROS işbirliği içerisinde robotik yazılımların ve uygulamaların daha rahat bir biçimde geliştirilebilmesini sağlamak amacıyla oluşturulmuştur. İşbirlikçi bir şekilde tasarıma örnek olarak, özellikle iç ortamlarda konumlandırma ve haritalama üzerine çalışmakta olan ve bu alanda uzmanlaşmış bir laboratuvar ile, elde edilen haritaları kullanarak seyrüsefer yapılması üzerine yoğunlaşmış başka bir grubun ortak çalışabilmesi verilebilir. Yine bu gruplara karmaşık ortamlarda bilgisayarla görü yöntemleri kullanarak ortamdaki ufak cisimleri veya insanları tespit etmek üzerinde çalışan başka bir grup da katılabilir. Böylelikle ROS çatısı altında farklı alanlarda uzmanlaşan grupların ortak çalışarak daha karmaşık ve gürbüz bir sistemin tasarımının yapılabilmesinin önü açılmış olmaktadır (ROS.org, 2018).

3.2 Benzetim Ortamı

Benzetim ortamları, farklı algoritmaların mümkün olduğu kadar çabuk ve gerçekçi bir biçimde test edilmelerine imkan vermesinden ötürü robotik çalışmalarında önemli bir yere sahiptir. Bu benzetim ortamlarından en popüler olanlarından biri olan Gazebo, verimli ve doğru bir şekilde robot topluluklarının, nesnelerin ve algılayıcıların karmaşık iç ve dış ortamlarda benzetilebilmesini sağlamaktadır (Gazebo, 2016).

Gazebo, robotun hareketi sırasında karşılaşabileceği dinamik ortam koşullarının oluşturulabilmesi amacıyla tasarlanmıştır. Benzetim ortamındaki bütün nesneler kütle, hız, sürtünme gibi itildiğinde, çekildiğinde, taşındığında ya da devrildiğinde gerçekçi tepkiler verilebilmesini sağlayan özelliklere sahiptir. Benzetim ortamındaki robotların kendileri de benzer şekilde eklemlerle birbirine bağlı sert gövdelerden oluşan dinamik cisimlerdir. Yüzey ve eklemlere uygulanabilen hem açısal hem de doğrusal hızlar yardımıyla çevre ile etkileşim sağlanabilmektedir. Gazebo benzetim ortamındaki aydınlanmadan sürtünme katsayılarına kadar hemen hemen herşey kontrol edilebilir özelliktedir. Bütün bu özellikleriyle birlikte Gazebo benzetim ortamı robotik sistemlerin test edilmesini ve hızlı bir biçimde geliştirilmesini sağlayan zengin bir benzetim ortamı sunmaktadır (Koenig ve Howard, 2004).

4. BULGULAR VE TARTIŞMA

Bu bölümde öncelikle yapılan çalışmalar birkaç ana başlık içerisinde incelenecektir. İlk aşamada Spektral Kümeleme yöntemi ile topolojik haritanın oluşturulması ve Dijkstra en kısa yol algoritmasının gerçeklenmesi anlatılacaktır. İkinci aşamada robotun seyrüseferi sırasında oluşabilecek çarpışmalardan kaçınması için kullanılan Vektör Alan Histogramı algoritmasından bahsedilecektir. Üçüncü aşamada, birden çok robotun kullanılabileceği bir durum göz önüne alınarak oluşturulan bir çarpışmadan kaçınma protokolünden bahsedilecektir. Dördüncü aşamada ise robotun afet ortamında seyrüseferi sırasında karşılaşabileceği rampaları aşabilmesi için kullanılan yöntemlerden bahsedilecektir.

4.1 Topolojik Harita Kullanılarak Dijkstra ile Seyrüsefer

Bu tez çalışmasında, harici keşif erkininden robotun hareketine bağlı olarak belirli zaman aralıklarında ortamın keşfedilen bölümlerini ifade eden bir metrik haritanın elde edildiği varsayımı yapılmıştır. Elde edilen bu metrik harita, bilinmeyen, boş ve dolu olarak üç durumdan oluşan bir doluluk ızgarasıyla ifade edilmektedir. Gelen haritada engellerin ve duvarların olduğu bölgeler dolu, robotun geçebileceği şekilde boş olan alanlar da boş durumları belirtmektedir. Her adımda gelen haritanın boyutu sabit olup, keşif sonucu dolu ya da boş durumda olmayan hücreler de bilinmeyen durumuyla ifade edilmiştir.

(Thrun, 1998)'deki çalışmada, hem topolojik hem de metrik haritaların seyrüsefer için birlikte kullanıldığı bir yaklaşımdan bahsedilmektedir. Metrik haritalarda, ortamlar doluluk ızgaralarıyla ifade edilmekte ve doluluk ızgaraları kolaylıkla oluşturulabilmektedir. Öte yandan, metrik haritalarda hücrelerin gerçek ortamda karşılık geldikleri boyutun belirlenmesi problemi ortaya çıkmaktadır. Ortam ne kadar çok hücre ile ifade edilirse, hesaplama yükü o kadar artmaktadır. Ortamın az hücre ile ifade edilmesi durumunda da doğruluk (accuracy) düşecektir. Topolojik haritalarda ise ortamın ifade edilmesi için çizgeler kullanılmaktadır. Çizgedeki düğümler ortamdaki belirli bölgelere karşılık gelirken, düğümler arasındaki ayrıtlar da düğümler arasında alınabilecek yolları ifade etmektedir. Topolojik haritaların metrik haritalara göre en önemli avantajının oluşturulan haritanın sıklığı olduğundan bahsedilmektedir. Ortam düğümler ve ayrıtlarla ifade edildiğinden dolayı metrik haritalara göre daha az hafıza-zaman karmaşıklığına sahiptirler. Topolojik haritaların dezavantajları ise, elde edilecek yol planlarının ideal olmayışı, robotun görüş açısına bağımlı olması ve aynı ortamdan geçip geçmediğini tanıyamayabilecek olması ile büyük ortamlarda haritayı oluşturmanın zor olabilmesi olarak sıralanabilir.

Topolojik haritaların oluşturulmasıyla ilgili yapılan ilk çalışmalarda, Voronoi diagramlarından faydalanılmıştır (Thrun, 1998; Choset, 1996). (Thrun, 1998)'deki çalışmada, kritik noktaların ve çizgilerin bulunmasında Voronoi diyagramından faydalanılmıştır. (Choset, 1996)'daki çalışmada, çarpışmasız bir yol planının elde edilmesi için bir yol haritası (roadmap) oluşturulmak istenmiştir. Yol haritası, robotun bulunduğu ortamı ifade eden bir boyutlu eğrilerden oluşan bir ağ olarak tanımlanmış ve yol haritasının oluşturulması için Hiyerarşik Genelleştirilmiş Voronoi Diyagramı'ndan (HGVD) faydalanılmıştır. Bu yöntemde, düğümler ve ayrıtlar birbirine eşit uzaklıktaki cisimler kullanılarak oluşturulmuştur.

(Buschka ve Saffiotti, 2002)'deki çalışmada, bina içi ortamların genellikle birbirine bağlı oda benzeri alanlardan oluştuğu belirtilmiş ve bu alanların saptanabilmesini amaçlayan bir yöntem önerilmiştir. Önerilen yöntem, bölütleme ve özellik çıkarımı olmak üzere iki aşamada çalışmaktadır. Bölütleme ile, oda benzeri alanlar saptanmakta ve robotun yeni bir oda benzeri alana girip girmediği belirlenmektedir. Özellik çıkarımı aşamasında ise saptanan alanlar seyrüsefer ve tanıma için kullanılabilecek bir takım geometrik özellik ile ilişkilendirilmektedir. Önerdikleri yöntemin artımlı (incremental) bir biçimde çalıştığından ve bu yöntem ile ortamın odalar ve koridorlar olarak ayrılmasıyla topolojik bir haritanın oluşturulmasının sağlandığından bahsetmişlerdir.

(Tapus ve Siegwart, 2006)'daki çalışmada, ortamların parmakizlerini kullanarak artımlı ve otonom bir şekilde topolojik haritalama ve küresel konumlandırma yapılabilen bir yöntem önermişlerdir. Önerilen yöntem ile ortam hakkında bir önbilgiye veya yapay yer işaretlerine gerek duyulmadan gezgin bir robotun seyrüseferinin gerçekleştirilebilmesi amaçlanmıştır. Ortamların parmakizine, robotun çevresindeki alanın bir takım ayırt edici özelliğini barındıran bir liste örnek verilebilir. Yapılan çalışmada da ayırt edici özellikler olarak görsel veriden alınan renk aralıkları ve dikey ayrıtlar ile lazer algılayıcıdan elde edilen köşeler kullanılmştır.

(Mozos ve Burgard, 2006)'daki çalışmada, bina içi ortamlar için mesafe verileri kullanılarak elde edilen geometrik haritalardan topolojik haritaların yaratılabilmesini amaçlayan bir yaklaşım önerilmiştir. Geometrik haritadaki her bir nokta, AdaBoost denetimli (supervised) öğrenme algoritması kullanılarak kapı, oda ve koridor olmak üzere anlamsal sınıflara ayrılmıştır. Daha sonra ortamdaki birbirinden farklı her bir anlamsal sınıf ve aralarındaki bağlar kullanılarak bir topolojik harita oluşturulmuştur. Böylelikle ortam, her düğüm bir anlamsal sınıf ile ifade edilen bölgeyi ve her ayrıt anlamsal sınıflandırılmış bölgeler arasındaki yolları ifade edecek şekilde, bir çizge ile topolojik olarak ifade edilmiştir.

Son yıllardaki çalışmalar incelendiğinde, topolojik haritaların elde edilmesi için spektral kümeleme yöntemini (Luxburg, 2007) kullanan çalışmalar görülmektedir. (Brunskill vd., 2007)'deki çalışmada, ham algılayıcı verisinden topolojik haritaların üretilmesi için çevrimiçi bir yöntem önerilmiştir. Çalışmada, ortamın haritası spektral kümeleme yöntemi ile alt-haritalara bölünmüştür. Robotun hareketi sırasında bu alt-haritaları tanıyabilmesi için, her bir alt-haritanın lazer taraması profilini sınıflandırmak amacıyla AdaBoost denetimli öğrenme algoritması kullanmışlardır. (Brunskill vd., 2007)'deki çalışmada, verimli yol planlamasının yapılabilmesi için, temel seviyedeki bir haritadan bölütleme ile yüksek seviye bir haritanın oluşturulması amaçlanmıştır. Temel seviyedeki harita, hem geometrik hem de görünüş tabanlı gösterimlere dayalı bir çizge olarak ifade edilmiştir. Bu harita üzerindeki düğümler, bir çizge bölütleme algoritması olan spektral kümeleme yöntemiyle kümelenerek, her küme merkezi yüksek seviye haritadaki bir düğümü belirtmektedir. Böylelikle yüksek seviye harita, çizge ile ifade edilmiş bir topolojik harita olarak kullanılmaktadır. (Valgren ve Lilienthal, 2008)'deki çalışmada, büyük ve hem iç hem dış olarak karışık bir ortamın, görüş tabanlı bir topolojik harita ile ifade edilmesi amaçlanmıştır. Topolojik haritanın artımlı olarak oluşturulduğuna değinilmiştir. Ortamın haritası imgelerin benzerliğine dayalı olduğundan, elde edilen bölütlenmiş gösteriminin, bir insanın ortamı algısına büyük ölçüde benzediği belirtilmiştir. Son olarak, bu çalışmada kullanılan yönteme benzer bir yöntem (Choi vd., 2011)'deki çalışmada sunulmuştur. Çalışmalarında, bir ev ortamından düşük maliyetli ses üstü mesafe algılayıcıları vasıtasıyla elde edilen verileri kullanmışlardır. Çalışmalarında, ortamın metrik olarak ifade edildiği bir doluluk ızgarasına hücre ayrıştırma yöntemi uygulanarak ortamdaki hareketin mümkün olduğu boş hücreler elde edilmiş ve bu hücrelerden spektral kümeleme yöntemiyle topolojik bir harita elde edilmiştir. Önerilen yöntem çevrimdışı bir biçimde kullanılmıştır.

Bu tez kapsamında, metrik haritanın harici keşif erkininden belirli aralıklarla geldiği varsayımına dayanarak, boş hücreler vasıtasıyla topolojik haritanın üretilmesi amaçlanmıştır. Bunu yapabilmek için, topolojik harita üretme yöntemleri incelenmiş ve spektral kümeleme yönteminin kullanılması uygun bulunmuştur. Spektral kümeleme yönteminde, kümelenecek hücrelerin belirlenmesi amacıyla ilginlik matrisi adı verilen ve hücrelerin birbiriyle olan ilişkisini gösteren bir matris kullanılmaktadır. (Liu vd., 2011)'deki çalışmaya göre, spektral kümelemenin üç tane dezavantajı olduğu belirtilmiştir. Bunlardan ilki, ilginlik matrisinin boyutudur. Bu matrisin boyutunun büyümesi, hesaplama yükünü artıracağından topolojik haritanın üretilmesi için gereken süreyi uzatmaktadır. İkinci dezavantaj hücrelerin kaç kümeye ayrılacağı problemidir. Küme sayısının fazla olması durumunda, daha sonra bu kümelerin kullanılmasıyla elde edilecek hareket düğümlerinin sayısı fazla olacak, bu da robotun hareketi sırasında sık sık dur-kalk yapmasına neden olacaktır. Üçüncü dezavantaj ise, spektral kümeleme yönteminin aynı

metrik harita üzerinde arka arkaya çalıştırılması sonucu aynı sonuçların elde edilemeyebilecek olmasıdır. Bu problemlerin çözümü için, (Kaleci vd., 2015)'deki çalışmada bu dezavantajları giderebilmek için ilk olarak, ilginlik matrisinin boyutunun büyümesini önlemek amacıyla, spektral kümeleme harici keşif erkininden metrik harita geldikçe çalıştırılmaktadır. Bu noktada, elde edilen metrik harita, sadece bir önceki adımdan sonra elde edilen yeni boş hücrelerden oluşmaktadır. Bu çalışmada, hem Dijkstra algoritmasının çalışma süresini düşürmek, hem de robotun daha verimli seyrüsefer yapmasını sağlamak için en az sayıda düğüm ile ortamın ifade edilmesi amaçlanmıştır. Bununla birlikte, küme sayısının küçük olması ortamdaki küçük odalar gibi bazı alanlara ulaşımın olmamasına neden olabilir. İkinci dezavantaj olan küme sayısının belirlenmesi problemi çözülürken, bahsedilen bu iki durum göz önünde bulundurularak, küme sayısı uyarlamalı bir şekilde belirlenmektedir. Spektral kümelemenin son aşamasında, hücreleri kümelere ayırmak için k-ortalama yöntemi kullanılmaktadır. Bu yöntemde, ilk andaki küme merkezlerinin rastgele belirlenmesi, üçüncü dezavantaj olan farklı küme merkezlerinin elde edilmesine neden olmaktadır. Bunu önlemek için, çalışmada basit bir algoritma önerilmiştir.

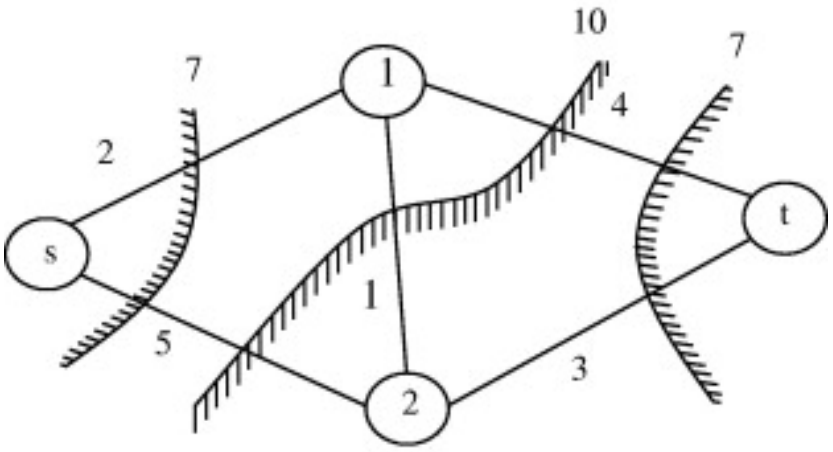
Topolojik harita elde edilirken son adım olarak, spektral kümeleme ile elde edilen küme merkezleri hareket düğümü olarak kabul edilmekte ve bu düğümlerin birbirine bağlanabilmesi için en küçük kapsayan ağaç yöntemi kullanılmaktadır.

4.1.1 Normalleştirilmiş Kesim Problemi

Topolojik haritaların oluşturulabilmesi için, öncelikle bazı düğümlerin oluşturulması gerekmektedir. Bu düğümlerin oluşturulabilmesi için, ortamı belirli parçalar olarak ele almak ilk akla gelecek çözümlerden birisidir. Bu parçaları elde etmek için metrik haritadan faydalanılabilir. Bu noktada, metrik haritanın bir matris olarak gösterilmesinden ötürü iki boyutlu ve ayrık olduğu varsayımına dayanarak, bu haritayı bir resim olarak düşünmek mümkün olmaktadır. Bir resmi parçalara ayırmak için geçmiş çalışmalarda sıklıkla resim bölütleme algoritmalarının kullanıldığı görülmüştür.

Geçmiş çalışmalarda en sık kullanılan çizge algoritmalarından biri (Shi ve Malik, 2000)'de önerilmiştir. Yazarlar bu algoritmada çizge bölütlemek için normalleştirilmiş kesim probleminin çözümünün kullanılabileceğini göstermişlerdir.

Çizge teorisine göre kesim, en temel anlamda çizgedeki düğümlerin iki ayrık altkümeye bölünmesi olarak tanımlanabilir (Bondy, Murty, vd., 1976). Kesim, içinde bir uç noktası bir altkümede ve diğer uçnoktası da diğer altkümede olan ayrıtları bulunduran bir kümeden oluşmaktadır. Bu noktada sonlu sayıda kesim elde edilebilir. Bu kesimler arasında, diğerleriyle karşılaştırıldığında kesimdeki eleman sayısı en küçük olan kesim, en



Şekil 4.1: Çizge kesim örneği (Duong vd., 2014)

küçük kesim olarak, eleman sayısı görece olarak en büyük olan da en büyük kesim olarak tanımlanabilir (Şekil 4.1).

$$kesim(A,B) = \sum_{u \subset A, v \subset B} W_{uv} \tag{4.1}$$

A ve B altkümelerini oluşturacak olan kesim Denklem 4.1'de verilmiştir. Bir V düğümler kümesi düşünülerek denklem incelendiğinde, W boyutu $|V|$ olan bir simetrik kare matrisi ifade etmektedir. W_{uv} ise u'nci ve v'nci düğümler arasındaki ayrıtın ağırlığına karşılık gelmektedir.

V düğümler kümesini A ve B altkümelerine ayırmak için, iki ölçüt kullanılmaktadır. Bu ölçütlerin 4.1 ile ifade edilen ilkinde, alt kümeler arasındaki benzerlik ifade edilmekte ve bu ölçütün düşük olması amaçlanmaktadır. Bu ölçütün düşük olmasının istenmesindeki amaç, altkümelerin arasında daha az ayrıt bulunarak maliyetin düşük, dolayısıyla da alt kümelerin birbirinden daha ayrık olması ve bu sayede de daha iyi bir bölütlemenin yapılmak istenmesidir.

$$iliski(A,V) = \sum_{u \subset A, v \subset V} W_{uv} \tag{4.2}$$

Kullanılan bir diğer ölçüt 4.2'de verilmiştir. Bu ölçütte, alt kümenin kendi içindeki benzerlik ifade edilmekte ve bu ölçütün yüksek olması amaçlanmaktadır. Basit şekilde özetlemek gerekirse; 4.1 ile gösterilen ilk ölçüt, altkümeleri oluşturmak için koparmamız

gereken ayrıtları ifade ederken, 4.2'deki ölçüt oluşturulan altkümelerin içlerindeki düğümler arasındaki bağlılığı, başka türlü ifade edersek altkümedeki ayrıt sayısını ifade ediyor denilebilir.

Bu noktada, V düğümler kümesinin $A \subset V$ ve $B \subset V$ alt kümelerine bölünmesi amaçlanmaktadır. Bu amaç doğrultusunda Denklem 4.1 ve Denklem 4.2'de verilen ölçütler kullanılmaktadır.

$$Nkesim(A,B) = \frac{kesim(A,B)}{iliski(A,V)} + \frac{kesim(A,B)}{iliski(B,V)} \tag{4.3}$$

(Shi ve Malik, 2000) incelendiğinde, Shi ve Malik normalleştirilmiş kesim problemini Denklem 4.3'de verildiği gibi tanımlamıştır. Denklemde görüldüğü üzere, $kesim(A,B)$'nin küçük, $iliski(A,V)$ ve $iliski(B,V)$'nin yüksek olması, diğer bir deyişle iki farklı altkümedeki düğümlerin arasındaki benzerliğin düşük, altkümelerin kendi içlerindeki düğümler arasındaki benzerliğin yüksek olması, normalleştirilmiş kesimin maliyetini düşürecektir. En düşük maliyete sahip olan normalleştirilmiş kesim Denklem 4.4 ile verilmiştir.

$$minKesim = \underset{A,B}{argmin}(Nkesim(A,B)) \tag{4.4}$$

Bu noktada, en küçük normalleştirilmiş kesim probleminin çözümünü bulmak amacıyla Shi ve Malik (Weiss, 1999)'da bu problemi detaylı olarak incelemiştir ve bu problemin genelleştirilmiş özdeğer problemine dönüştürülebileceğini farketmişlerdir. Genelleştirilmiş özdeğer problemi Denklem 4.5'te tanımlanmaktadır. Denklemde W düğümler arasındaki ilişkiyi tanımlamakta ve ilginlik matrisi olarak adlandırılmaktadır. D ise köşegen bir matris olup köşegendeki elemanların değerleri W matrisinin o satırdaki elemanlarının toplamını ifade etmekte ve 4.6 ile gösterilmektedir.

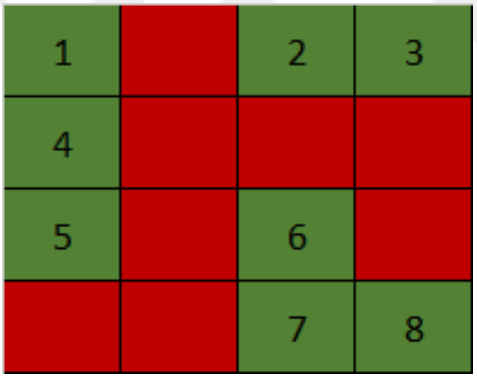
$$(D - W)y = \lambda Dy \tag{4.5}$$

$$D(i,i) = \sum_{j} W(i,j). \tag{4.6}$$

Genelleştirilmiş özdeğer problemi çözümü elde edildiğinde, özvektörler, en küçük özdeğere karşılık gelen özvektörden en büyüğüne doğru sıralanmaktadırlar. Bu noktada, ideal bir durumda ilk özvektörün elemanlarının tamamının sıfır olması beklenmektedir. Dolayısıyla bu özvektör bölütleme için herhangi bir bilgi taşımamaktadır. Bu durumda, ikinci özvektörün elemanlarının, ideal bir durumda iki farklı değerden oluşması beklenmektedir. Bu iki farklı değer kullanılarak düğümler iki farklı kümeye ayrılabilir. Düğümlerin ikiden fazla olması kümeye ayrılması istendiği durumlarda, üçüncü, dördüncü veya diğer özvektörler kullanılabilir (Shi ve Malik, 2000).

4.1.2 Topolojik Haritanın Oluşturulması

Harici keşif erkininden metrik harita gönderildikten sonra, (Shi ve Malik, 2000)'de bahsedilen normalleştirilmiş kesim problemi ele alınmış ve Shi ve Malik'in (Weiss, 1999) algoritması kullanılarak çözülmüştür. Bu bağlamda, ilk olarak oluşan metrik haritadan bir ilginlik matrisi oluşturulmalıdır. Örnek bir metrik harita Şekil 4.2'de verilmiştir. Şekilde kırmızı hücreler dolu ve yeşil hücreler boş durumundaki hücreleri göstermektedir. Şekilde görülen numaralar boş hücrelerin indeksleri olup ilginlik matrisinin oluşturulmasında kullanılmaktadır.



Şekil 4.2: Metrik Harita

	1	2	3	4	5	6	7	8
1	1	0	0	0.9	0	0	0	0
2	0	1	0.9	0	0	0	0	0
3	0	0.9	1	0	0	0	0	0
4	0.9	0	0	1	0.9	0	0	0
5	0	0	0	0.9	1	0	0	0
6	0	0	0	0	0	1	0.9	0.7
7	0	0	0	0	0	0.9	1	0.9
8	0	0	0	0	0	0.7	0.9	1

Şekil 4.3: İlginlik Matrisi

	1	2	3	4	5	6	7	8
1	1.9	0	0	0	0	0	0	0
2	0	1.9	0	0	0	0	0	0
3	0	0	1.9	0	0	0	0	0
4	0	0	0	2.8	0	0	0	0
5	0	0	0	0	1.9	0	0	0
6	0	0	0	0	0	2.6	0	0
7	0	0	0	0	0	0	2.8	0
8	0	0	0	0	0	0	0	2.6

Şekil 4.4: D Matrisi

İlginlik matrisi W boyutu boş hücre sayısına eşit olan kare, simetrik bir matristir ve bu matrisin elemanları boş hücreler arasındaki ilişki durumunu belirtmektedir. İlişki durumlarından 0 tam benzemezliği ve 1 ise tam benzerliği ifade etmektedir. Şekil 4.3'de verilen metrik harita için hesaplanan ilginlik matrisini belirtmektedir. Şekilde görüldüğü üzere her hücrenin kendisine karşılık gelen indekse 1 atanmaktadır. Değeri 1 olan her hücrenin, yatay ve dikey komşularından boş hücre olanına 0.9, köşegenindeki komşularından boş hücre olanlarına da 0.7 değeri atanmaktadır.

D köşegen matrisi, ilginlik matrisi kullanılarak hesaplanmakta ve Şekil 4.4'de gösterilmektedir. Genelleştirilmiş özdeğer problemi (Denklem 4.5) Şekil 4.3 ve Şekil 4.4 değerleri için çözülmüştür. Elde edilen özvektörler en küçük özdeğere karşılık gelen özvektörden en büyüğüne doğru sıralanmıştır.

	1	2	3
1	-0,54	0,00	0,00
2	0,00	0,00	0,70
3	0,00	0,00	0,70
4	-0,65	0,00	0,00
5	-0,54	0,00	0,00
6	0,00	0,57	0,00
7	0,00	0,59	0,00
8	0,00	0,57	0,00

Şekil 4.5: Örnek özvektörler

Bu noktada, elde edilen özvektörler Şekil 4.5'de verilmiştir. Bu özvektörler kullanılarak boş hücrelerin kümelenmesi amaçlanmaktadır. Ancak, spektral kümelemenin dezavantajlarından biri olan küme sayısı belirleme problemi bu aşamada karşımıza çıkmaktadır. Bir ortamın çok sayıda küme merkezi ve dolayısıyla düğüm ile ifade edilmesi, topolojik harita yönteminin metrik harita yöntemine göre sağlamış olduğu hafıza-zaman karmaşıklığı avantajını azaltacaktır. Bir noktada her hücrenin bir kümeyi ifade etmesi

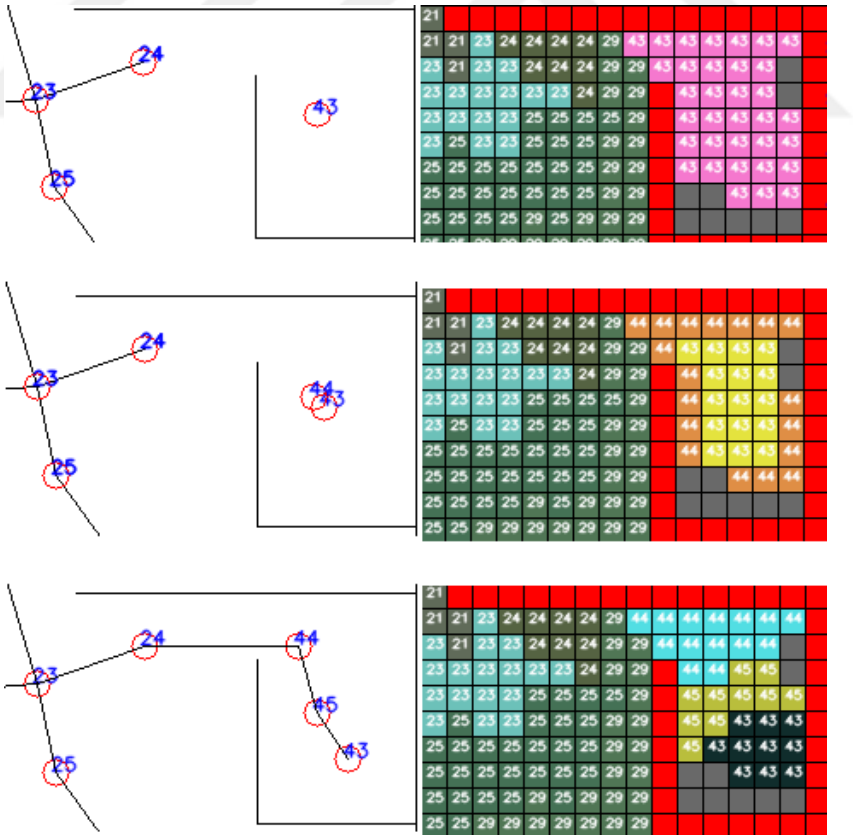
durumunda topolojik harita metrik haritayla aynı zaman karmaşıklığına ulaşacaktır. Öte yandan, ortamın az sayıda küme merkezi ve dolayısıyla düğüm ile ifade edilmesi durumunda, ortamın bazı bölgeleri topolojik harita tarafından ifade edilmeyebilir. Bu durumda robotun bu bölgelere ulaşamaması durumuyla karşı karşıya kalınabilir. Haliyle, kaç boş hücreden bir küme oluşturulacağına bahsedilen iki durum göz önüne alınarak karar vermek gerekmektedir. Küme sayısının belirlenmesinde, belirli sayıda boş hücrenin ($bosHucredenKumeSayisi$) bir küme oluşturması istenmiştir. Bu durumda küme merkezi sayısı $kumeMerkeziSayisi$ Denklem 4.7 ile hesaplanmaktadır.

$$kumeMerkeziSayisi = \frac{BosHucreSayisi}{bosHucredenKumeSayisi} \tag{4.7}$$

Bu aşamada, normalleştirilmiş kesim probleminin çözümünden elde edilen özvektörlerden küme merkezlerine karar verilmesi gerekmektedir. Shi ve Malik'in (Shi ve Malik, 2000)'deki çalışmalarında küme merkezlerini bulabilmek için çeşitli yöntemler kullanılabileceğinden bahsetmişlerdir. Bu yöntemler ayırt edici bir özellik kullanarak özvektörleri kümelemektedirler. Örneğin sıfır ayırt edici özellik olarak seçilip, özvektörün elemanları işaretlerine göre kümelenebilir. Başka bir yöntem özvektördeki elemanların ortalamasının kullanılarak bu değerin altında ve üstünde olan küme elemanlarının ayrıştırılmasıdır. Şekil 4.5'te Şekil 4.2'deki örnek metrik harita kullanılarak elde edilen özvektörler verilmiştir. İlk özvektör incelendiğinde, 1., 4. ve 5. elemanların 0 ayırt edici özellik olarak kullanılmasıyla kümelendiği görülmektedir. İkinci özvektöre bakıldığında, ilk özvektörde ayrılan elemanlardan geriye kalanlarından 6., 7. ve 8. elemanların 0'dan farklı değerlere göre ayrıldığı gözlemlenebilir. Son özvektörde ise sonra kalan 2. ve 3. elemanların kümelendikleri görülmektedir. Böylelikle Şekil 4.2'de verilen metrik harita örneğinden 3 adet küme merkezi elde edilmiştir. Örnek metrik harita incelendiğinde de sayı gruplamalarının gerçekten tutarlı bir şekilde kümelendiği anlaşılabilir.

Öte yandan, boş hücrelerin kümelenmesi için, elde edilen özvektörlerden yararlanılarak k-ortalama gibi yöntemlerden de faydalanılabilir. Çalışmanın bu aşamasında da, belirlenen küme sayısına göre ayarlanan sayıda özvektör kullanılarak k-ortalama yöntemi ile de kümeleme yapmak mümkün olmaktadır. Spektral kümelemenin son dezavantajı olan tekrar edilemeyen sonuçlar k-ortalama yönteminin küme merkezlerinin ilk konumlarının rastgele atanmasından kaynaklanmaktadır. Bunu önlemek amacıyla, k-ortalama yöntemindeki küme merkezlerinin ilk konumları birbirine en uzak örnekler olacak şekilde seçilmektedir. Burada örnekler arasındaki öklid uzaklıklarından faydalanılmıştır.

Bu aşamada, elimizde k-ortalama yöntemi ile elde edilen küme merkezleri bulunmaktadır. Bu küme merkezleri, topolojik haritanın düğümlerini ifade etmektedir. Bununla birlikte, bu düğümler arasındaki ayrıtları oluşturmak ve topolojik haritayı tamamlamak amacıyla en küçük kapsayan ağaç algoritmasından yararlanılmıştır. Bu algoritma, topolojik haritayı oluştururken ayrıt ağırlıklarının toplamının en küçük olmasını sağlamaya çalışmaktadır. Bu durumda, bazı ayrıtların metrik haritadaki dolu durumunda bulunan hücrelerden geçebildiği görülmüştür. Böyle durumlarla karşılaşıldığında, iki düğüm arasındaki ayrıtı ifade eden sanal bir çizgi oluşturulmakta ve bu çizginin metrik harita üzerinde geçtiği hücrelerin durumu kontrol edilmektedir. Eğer bu sanal çizgi dolu bir hücreden geçiyorsa bu ayrıtın ağırlık değeri çok büyük bir sayı yapılarak en küçük kapsayan ağaç algoritması tekrar çalıştırılmaktadır. Oluşturulan topolojik haritada bahsedilen bu durum yüzünden bazı düğümlere ulaşılabilirliği ifade eden ayrıtların oluşturulmadığı görülmüştür. Bütün düğümlerin topolojik harita üzerindeki ayrıtlar ile ulaşılabilir olması için, kapı gibi dar bölgelerde engelden geçmeyen ayrıtlar üretmek amacıyla bir takım yeni düğümler oluşturulmaktadır. Bu düğümlerin oluşturulması için $kumeMerkeziSayisi$ parametresi ve özvektör sayısı uyarlamalı olarak değiştirilmektedir.



Şekil 4.6: Uyarlamalı düğüm oluşturma örneği

Algorithm 1: Çarpışmasız Topolojik Harita Üretme Algoritması (Kaleci vd., 2015)

Girdi: Lazer Mesafe Taraması
Çıktı : Elde edilen düğümler
Metrik haritayı oluştur
repeat
  Boş hücreleri bul
  W ve D matrislerini oluştur
  Denklem 4.5'teki genelleştirilmiş özdeğer problemini çöz
  K-ortalama ile kümelendir
  Küme merkezlerini kullanarak düğümlere karar ver
  En küçük kapsayan ağaç algoritmasını çalıştır ve çarpışmasız topolojik haritayı oluştur
  if *bağlanamayan veya engelden geçen düğümler mevcut* **then**
    $kumeMerkeziSayisi$ ve özvektörü sayısını güncelle
  end
until *her belirli zaman aralığında*;

Çarpışmasız topolojik harita adım adım oluşturulurken robot hedefine gitmek için topolojik haritadaki düğümleri kullanmaktadır. Bu aşamada robotun topolojik haritayı kullanabilmesi için kendine en yakın ve robot ile düğüm arasında dolu hücrenin bulunmadığı düğüme karar vermesi gerekmektedir. Benzer şekilde, topolojik haritadan ayrılıp hedefine varması için asıl hedefine en yakın düğüm bulunmaktadır. Bu düğüm ile asıl hedefi arasında herhangi bir engel varsa en yakın ikinci düğüme bakılmaktadır ve bu süreç uygun bir hedef düğümü bulunana kadar devam etmektedir. Robotun yol planı, üç aşamada gerçekleşmektedir. İlk aşamada bulunduğu yerden kendisine en yakın düğüme, daha sonra topolojik harita üzerindeki düğümler ile hedefe en yakın düğüme ve bu düğümden de asıl hedefine hareket etmek şeklinde gerçekleşmektedir. İkinci aşamanın planlanabilmesi için oluşturulan bir çizgedeki herhangi iki düğüm arasındaki en kısa yolu bulmak amacıyla çeşitli algoritmalardan yararlanılabilir. Bunlardan en sık kullanılanlarından bir tanesi, bir çizgede belirlenen başlangıç ve bitiş düğümleri arasındaki en kısa yolu bulmak için kullanılan Dijkstra'nın algoritmasıdır. Algoritma, girdi olarak bir çizge ile birlikte başlangıç ve hedef düğümlerini kabul eder. Başlangıç düğümünden başlayarak, komşu hücrelerini ziyaret eder ve komşularından maliyeti en küçük olanını, başlangıçtan hedefe olan yolu ifade edecek olan düğümlerin tutulduğu öncelik kuyruğuna ekler. Bu işlem her eklenen düğümün komşuları için hedef düğüme ulaşılıncaya kadar devam eder.

Algoritmada kullanılacak çizge, dolayısıyla her bir düğümün hangi düğümlerle bağlantılı olduğu ve her bağlı iki düğüm arasındaki ayrıtın maliyeti bir benzerlik matrisi ile ifade edilebilir. Çizgedeki düğümler arasındaki maliyetin belirlenmesinde, düğümlerin

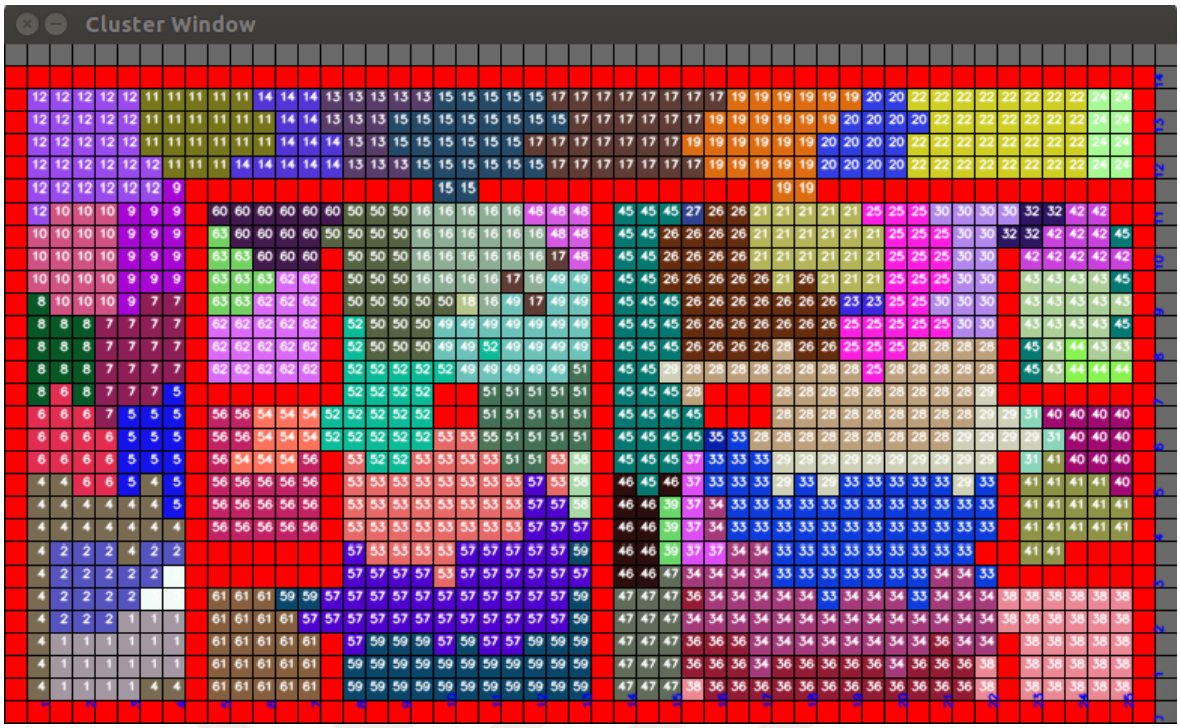
bulundukları koordinatlar arasındaki öklid uzaklıklarından faydalanılmıştır. Benzerlik matrisinde, birbirine bağlı olmayan düğümlere karşılık gelen yerlere 0 atanırken, birbirine bağlı olanlara aralarındaki öklid uzaklıkları atanmıştır. Dijkstra algoritması ile oluşturulan bu benzerlik matrisi kullanılmış ve herhangi iki düğüm için elde edilen en kısa yolu belirten düğümler bir vektöre kaydedilmiştir. Robotun iki düğüm arasındaki hareketi de bu vektördeki düğümlerin gerçek koordinatları kullanılarak gerçeklenmiştir.

Topolojik harita yönteminin kullanılmasındaki temel amaç, ortamın, bütün oda ve koridorlara ulaşılabilecek şekilde mümkün olan en az sayıda düğüm ile ifade edilmesidir. Bunun sağlanabilmesi için de her odaya ait en az bir tane düğüm bulunması gerekmektedir. Klasik spektral kümeleme yöntemi ile düğüm oluşturulduğunda, düğümler aralarındaki mesafeler yaklaşık olarak aynı olacak şekilde üretilmektedir. Düğümler arasındaki bu mesafenin çok küçük olması durumunda, fazladan gereksiz düğümler üretilebilmektedir. Eğer mesafe çok büyük olursa da, bazı odalara ulaşmak mümkün olmayabilmektedir. Geliştirilen uyarlamalı yöntemin kullanılmasıyla, bu mesafeler ayarlanarak, koridorlar gibi büyük alanlarda az sayıda düğüm oluşturulurken, küçük odalar gibi dar alanlarda da daha fazla sayıda düğüm kullanılarak ortamın her yerine erişim mümkün kılınmaktadır. Bunlara ek olarak, topolojik harita oluşturma yöntemi, çizge oluşturma algoritması belirli zaman aralıklarında çalıştırılarak, o zaman aralıklarında elde edilen boş hücrelere uygulandığından, çevrimiçi bir şekilde çalışmaktadır. Bu sayede de hesaplama yükü azaltılarak, hafıza karmaşıklığından kazanç sağlanmaktadır. Şekil 4.7 ve 4.8'da, bahsedilen uyarlamalı yöntem ile elde edilen kümeleme sonuçları ve topolojik haritaya bir örnek gösterilmektedir. Algoritma 1'de de kullanılan çarpışmasız bir biçimde topolojik harita üretme algoritması ifade edilmiştir.

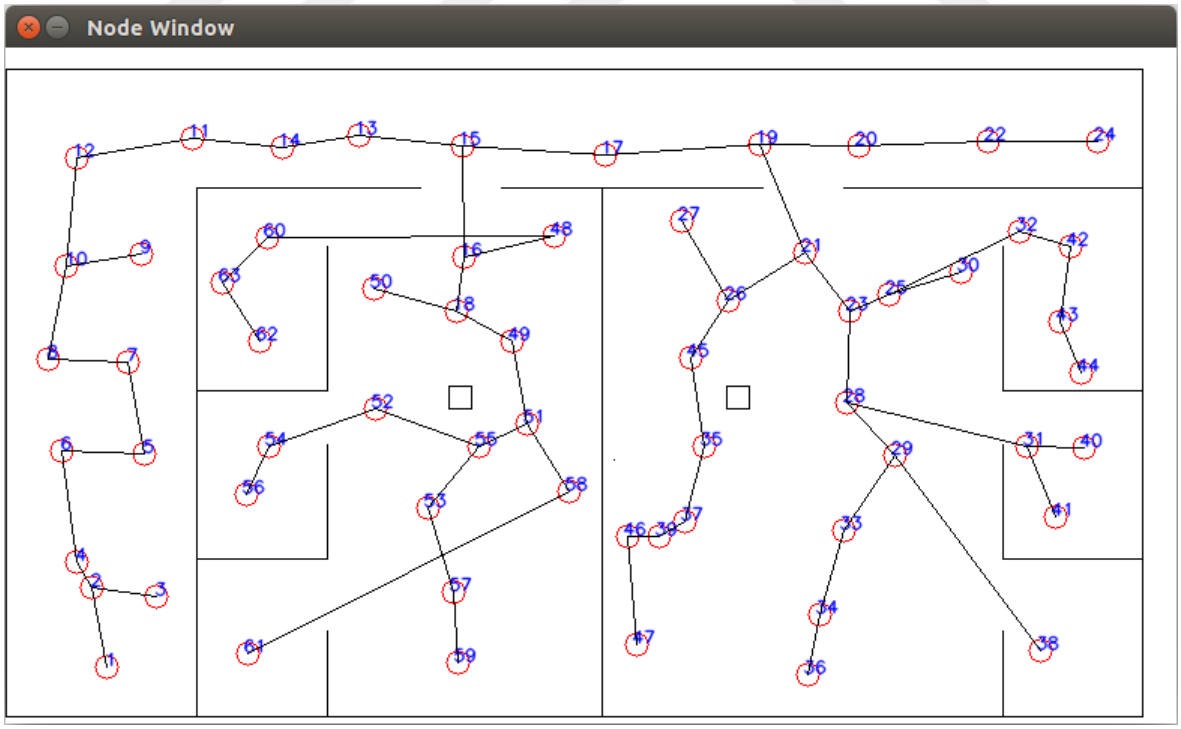
4.2 Vektör Alan Histogramı (VAH)

Bir önceki bölümde anlatılan topolojik harita üretme ve bu haritanın kullanılmasıyla yol planı yapıldığında, robotun herhangi bir engele çarpmadan seyrüseferini gerçekleştireceği varsayımı yapılmaktadır. Bununla birlikte ortamda bilinmeyen veya hareketli(diğer robotlar, insanlar vb.) bir takım engellerin bir çok gerçek zamanlı uygulamada mevcut olabileceği düşünülmelidir. Bu gibi dinamik ortamlarda ve birden çok robotun eş zamanlı çalıştığı durumlarda, robotun herhangi bir engele çarpmadan güvenli bir şekilde seyrüseferinin yapılması gerekmektedir. Bu problem genellikle engelden sakınma olarak adlandırılmaktadır.

Engelden sakınma probleminin çözümüne yönelik olarak en bilinen yöntemlerden birisi potansiyel alanlar yöntemidir (Khatib, 1985). Bu yöntemde çekici(hedef) ve itici (engel)



Şekil 4.7: Uyarlamalı yöntem ile elde edilen kümeleme sonuçları



Şekil 4.8: Uyarlamalı yöntem ile elde edilen topolojik harita

kuvvetlerin birleşiminden oluşan vektör robotun seyrüseferini yaparken bir sonraki adımda nereye gideceğini belirlemektedir. Bu yöntemin en büyük dezavantajı, kuvvetlerin birbirini nötrlemesi durumlarında karşılaşılan robotun hareket edememesi durumudur.

VFH (Vector Field Histogram = Vektor Alan Histogramı) son yıllarda popülerliği artan ve robotun güvenli bir biçimde seyrüseferini gerçekleştirmesini sağlayan bir çarpışmadan sakınma yöntemidir. Bu yöntemin en büyük avantajı yüksek hızlarda dahi engelden sakınmayı sağlamasıdır. Yöntemde, iki boyutlu bir kartezyen histogram haritası kullanılmaktadır. İlk olarak, bu ızgara tek boyutlu polar histograma çevrilmektedir. Bu polar histogram sektörlere bölünmekte ve her sektör içinde bulunan engellerin yoğunluğuna göre bir değer ile ifade edilmektedir. Daha sonra, daha az engel içeren uygun bir sektör seçilerek robotun açısal ve doğrusal hızlarına bu sektöre göre karar verilmektedir (Borenstein ve Koren, 1991; Ulrich ve Borenstein, 1998). Bu tez çalışması kapsamında, (Akçakoca vd., 2014)'de anlatılan yöntem, kinect algılayıcısı verisi yerine, lazer mesafe algılayıcısı verisi kullanılarak gerçeklenmiştir.

4.3 Çok Robotlu Seyrüsefer için Çarpışmadan Kaçınma

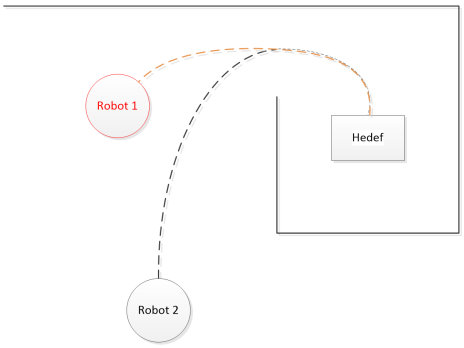
Çok robotlu sistemlerde, çarpışmadan kaçınma protokolleri robotların verilen görevleri istenilen şekilde yerine getirebilmesi için önem arz etmektedir. Bu konuda literatürde çeşitli çalışmalara rastlanmaktadır. (Liu vd., 2000)'daki çalışmada, normalde çarpışma olarak görünmemesine karşın, robotların hareketleri düşünüldüğünde çarpışmanın gerçekleşebileceği durumlardan bahsedilmiştir. Ayrıca bu durumları aşabilmek için bir çarpışmadan sakınma algoritması da önerilmiştir. Önerilen algoritma merkezi bir şekilde çalışmaktadır. Ortamın ifadesi için doluluk ızgarası kullanılmakta ve her robot en uygun yol planını A* algoritmasını kullanarak planlamaktadır. Her robot, koordinatör olarak adlandırılan merkezi planlayıcıya o anda bulunduğu düğümün konumunu, bir önceki düğümün konumunu, hedef düğümün konumunu ve hedefe olan uzaklığı göndermektedir. Eğer herhangi bir çarpışma durumu algılanmazsa, robotlar kendi yol planlarına uygun hareket etmektedirler. Eğer çarpışma algılanırsa, çalışmalarında tanımladıkları Super A* algoritması çalıştırılarak robotların birbirine çarpmadan hareket etmesi sağlanmıştır. Bu algoritmaya göre, robotlardan hedefe daha yakın olanına öncelik tanınmaktadır. Super A* algoritmasına göre de robotlar çarpışmadan sakınabilecekleri bir yol planı oluşuncaya kadar, en uygun A* yol planından saparak farklı yol planları denerler. Bu durumda, robotlardan birisi diğeri etrafında dolanarak çarpışma durumundan kurtulduktan sonra kendi yoluna devam ederken, diğer robot çarpışma durumundan kurtulduğu zaman direk kendi yoluna gitmektedir. Yazarlar bu durumu gerçek ortam testleriyle gösterdiklerini belirtmektedirler.

(Bennewitz ve Burgard, 2000)'daki çalışmada, merkezi olmayan(ayrılmış) ve robotların önceliklerini göz önüne alarak çarpışmalar önlenmeye çalışılmıştır. Robotların yol planlarını A* kullanarak hazırlamışlardır. Bu noktada, yol planlarını kontrol ederek çakışma olup olmadığı kontrol edilmekte ve düşük öncelikli robotların planları tekrar yapılmaktadır. Planlama esnasında bir robotun bir anda bir konumda olma olasığına karar verilmekte ve maliyet hesabında bu olasılık kullanılmaktadır. Önerdikleri yöntemi, yeniden planlama yapmayan ve robotların hızlarını ayarlayarak çarpışmayı önleyen yöntem ile karşılaştırmış ve daha verimli çalıştığını göstermişlerdir.

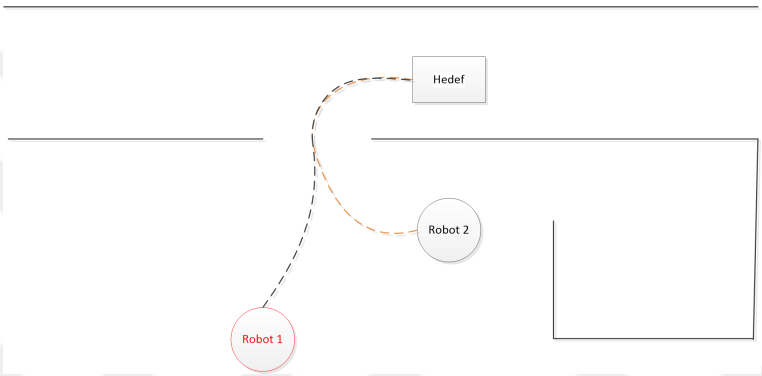
(Jager ve Nebel, 2001)'daki çalışmada, ayrışmış ve merkezi bir yöntem önerilmiştir. Bu yöntemde robotlar arası mesafe belli bir eşik değerin altına düştüğünde iki robot arasında bir koordinasyon hattı oluşturulmaktadır. Bu hatta robotun biri koordinatör diğeri ise partner olarak seçilmektedir. Koordinatör partnerinden yol planını istemekte ve her iki robot için çizelgeleme yapmaktadır. Bu çizelgelemede yol planları parçalara bölünmekte ve bu parçalardan hangi zaman aralıklarında geçileceği belirlenmektedir. Her robot bir parçayı bitirdiğinde diğer parça için uygulanacak planı istemekte ve uygulamaktadır.

(Peasgood vd., 2008)'da, birleştirilmiş ve merkezi bir yaklaşım önerilmiştir. Bu yöntemlerin ölçeklenirlik ve hesaplama maliyeti dezavantajlarını önlemek için ortamın topolojik haritası kullanılmaktadır. Daha sonra, ortamın geometrik merkezine en yakın düğümün kök olarak seçildiği, bütün düğümlerin bağlı olduğu ve döngü içermeyen bir kapsayan ağaç oluşturulmaktadır. Topolojik harita kullanılarak yol planları elde edildikten sonra robotlar arasında bir çarpışma olup olmayacağına kapsayan ağaç ile karar verilmektedir. Bu sayede hesaplama maliyeti doğrusal olan merkezi ve birleştirilmiş bir yöntem gerçeklenmiştir.

Bu tez kapsamında oluşturulan çarpışmadan kaçınma protokolünün tasarımı için iki adet birbiriyle aynı özelliklere sahip robotun afet ortamında çalıştığı bir durum düşünülmüştür. İlk aşamada, robotların hedeflerinin harici keşif düğümünden gelen doluluk ızgarasındaki konumları karşılaştırılmaktadır. Karşılaştırma için, hedeflerin doluluk ızgarasında denk düştüğü hücre koordinatlarının x ve y eksenleri bazında mutlak farkına bakılmaktadır. Eğer hedeflerin hem x hem de y hücre koordinatları arasındaki fark $neigThr$ parametresiyle tanımlanan bir eşik değerin altındaysa, hedefler birbirlerinin komşuluğundadır denilmektedir ve bu durumda robotların aynı hedefe gitmek istedikleri düşünülmektedir. Bu durumlar genellikle her iki robotun birden aynı odaya girmek istediği (Şekil 4.9)ya da odada bulunan robotların aynı koridora çıkmak istediği (Şekil 4.10) zaman meydana gelmektedir.



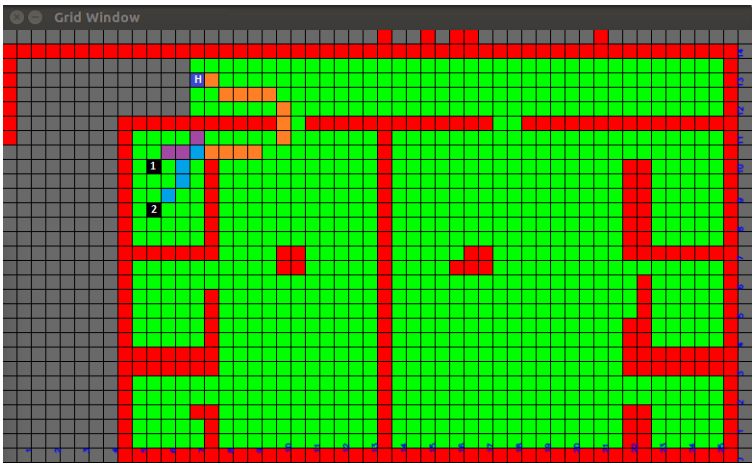
Şekil 4.9: Robotların aynı odaya girmek istediği an



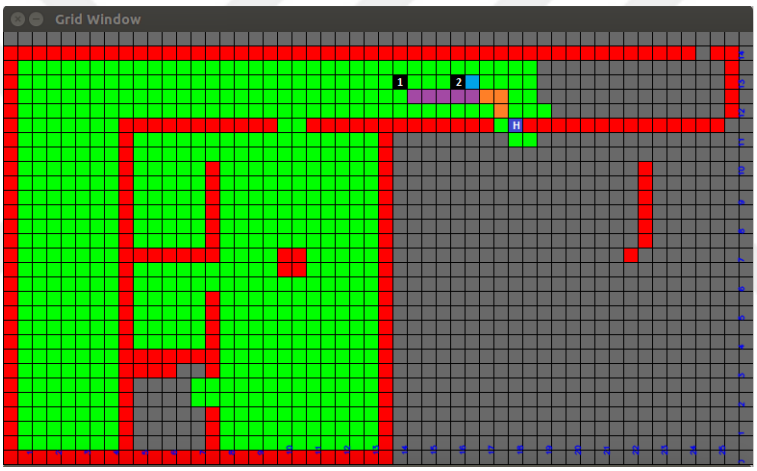
Şekil 4.10: Robotların aynı koridora çıkmak istediği an

Bu örneklere benzer durumlar ile karşılaşıldığında, robotlardan birisinin dururken diğerinin ortak hedefe devam etmesinin çarpışmayı önleyeceği düşünülmüştür. Dolayısıyla, ikinci robot beklemekte iken birinci robotun ortak hedefe doğru yoluna devam ettiği olaylar Durum 1 olarak adlandırılmıştır. Durum 1, birinci robotun hedefe ikinci robottan daha yakın olduğu zamanlarda oluşmaktadır (Şekil 4.11). Durum 2 ise ikinci robotun hedefe birinci robottan yakın olduğu zamanlarda meydana gelmekte ve ikinci robot hedefe giderken birinci robot beklemektedir (Şekil 4.12). Şekillerde, robotların çarpışma durumlarını daha iyi ifade edebilmek için, robotların konumları ile ortak hedef arasında harici keşif erkinden gelen doluluk ızgarası kullanılarak A* algoritmasıyla oluşturulan hücre bazlı yol planları gösterilmektedir. Robotlar, içinde numaralarının yazılı olduğu siyah karelerin bulunduğu konumlarda bulunmaktadır. Mor ve mavi renkle gösterilen kareler sırasıyla birinci ve ikinci robotun yol planlarını göstermektedir. Robotların ikisinin birden farklı yollarla kapıdan çıkmaları mümkün olmayacağı için kapıdan sonraki yol planları aynı olmakta ve turuncu renk ile ifade edilmektedir. Son olarak, hedef ise lacivert renkle gösterilmektedir.

Sonuç olarak, her iki robotun da ortak bir hedefe gitmek istemesi durumunda, hedefe yakın olan robotun hareketine devam ederken diğer robotun beklemesi ile



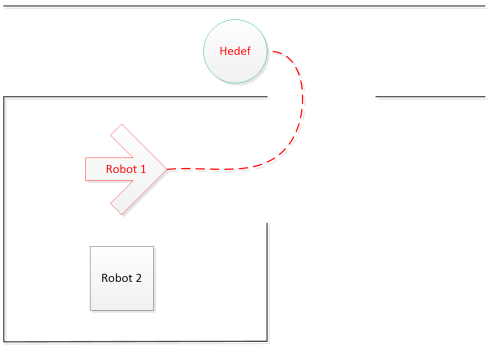
Şekil 4.11: Durum 1 Örneği



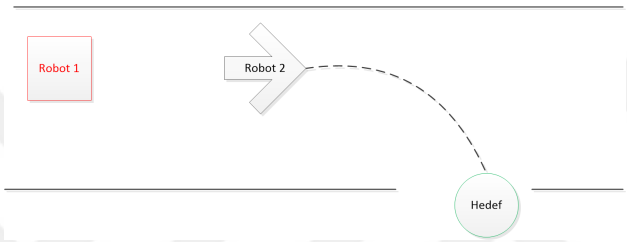
Şekil 4.12: Durum 2 Örneği

çarpışmadan kaçınmanın sağlanacağı düşünülmüştür. Sonuçlar bölümünde verilecek olan görsel sonuçların anlaşılmasını kolaylaştırmak amacıyla, birinci robot ile ilgili görseller kırmızıyla, ikinci robot ile ilgili görseller ise siyah renk ile gösterilecektir. Durum 1 ve Durum 2'de olduğu gibi her iki robotun da gitmek istediği ortak hedef yeşil çember ile gösterilecektir. Çemberin içindeki sayının rengi, ortak hedefe hangi robotun gittiğine göre belirlenecektir. Kare şekli ile ise beklemekte olan robot ifade edilecektir (Şekil 4.13 ve Şekil 4.14).

Bu noktaya kadar robotların ortak hedefe gittiği durumda nasıl çarpışmadan kaçınacakları anlatılmıştır. Robotların hedeflerinin farklı çıkması durumunda ise, ilk başta çarpışma olup olmayacağının tespit edilmesi gerekmektedir. Bu amaç doğrultusunda robotların yol planlarının karşılaştırılması yapılmaktadır. Bu noktada, yol planları



Şekil 4.13: Durum 1 Örneği için görsel sonuç



Şekil 4.14: Durum 2 Örneği için görsel sonuç

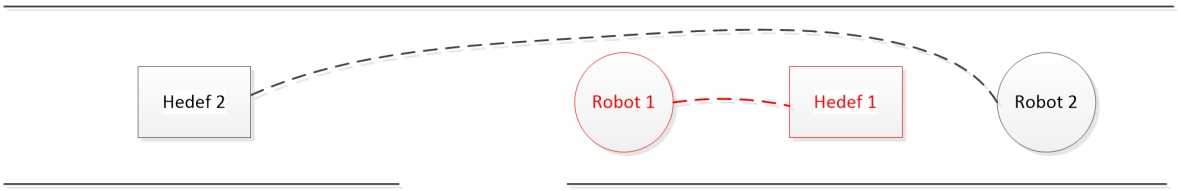
karşılaştırılırken çakışmaların daha kesin bir biçimde saptanabilmesi için, hedeflerin aynı çıktığı durumların ifade edilmesinde kullanılana benzer şekilde yeniden A* algoritmasından faydalanılmıştır. A* algoritmasından faydalanılmasının nedenini açıklamak için, Dijkstra algoritmasıyla elde edilen yol planının yapısını düşünmek gerekmektedir. Dijkstra ile elde edilen yol planındaki her eleman bir düğümü ifade etmektedir. Yol planları karşılaştırılırken bu düğümlerin kullanılması durumunda, düğümler arasında bir çakışma olmamasına rağmen, robotların düğümler arası hareketi sırasında karşılaşmaları sonucu çarpışma durumuyla karşılaşılabilir. Bu nedenden ötürü, çarpışmaların daha doğru biçimde saptanması amacıyla yol planlarının karşılaştırması yapılırken A* algoritmasından yararlanılmıştır.

Çarpışma protokolü, robotların hareketinin fazla aksamaması için, robotların arasındaki mesafe $collDist$ parametresi ile tanımlanan bir eşik değerinin altında ise çalıştırılmaktadır. Dolayısıyla, yol planlarının karşılaştırılması yapılırken, bu parametre ile tanımlanmış menzilin içinde kalacağı düşünülen ilk $distThr$ parametresi ile tanımlanan kadar sayıda elemanın kullanılması yeterli olacaktır. Karşılaştırmaya, robotlardan birinin yol planının ilk elemanı ile diğer robotun yol planın $distThr$ sayısında elemanına bakılarak başlanmaktadır. Bakılan elemanların her iki eksendeki koordinatları arasındaki fark, hedeflerin ortak olduğu durumdakine benzer şekilde $neigThr$ eşik değerinin altında ise,

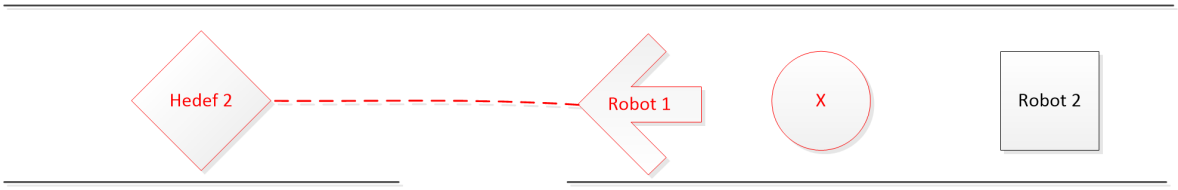
robotların yol planlarının birbirlerinin yakın komşuluklarında olmaları dolayısıyla çakıştığı ve bu durumda bir çarpışmanın muhtemel olduğu anlaşılmaktadır. Bu işlemler karşılaştırılan yol planlarının ilk $distThr$ kadar elemanı için bir çarpışma bulununcaya kadar tekrarlanmaktadır.

Robotların yol planları karşılaştırıldıktan ve bir çakışma saptandıktan sonra, robotların davranışlarına karar vermek için ilk aşamada robotlar ve hedefler arasındaki uzaklıklar hesaplanmaktadır. Bu uzaklıklar kullanılarak da robotların beklemek, kendi hedefine devam etmek ya da diğer robotun hedefine gitmek şeklindeki davranışlardan hangisini yapacağına karar verilmektedir.

Şekil 4.15)'de her iki robotun da birinci robotun hedefine daha yakın olduğu bir durum gösterilmiş ve bu olay Durum 3 olarak ifade edilmiştir. Şekil incelendiğinde, robotlardan herhangi birinin yakın olmasından ötürü birinci robotun hedefine gitmesi ile çarpışma tehlikesinin ortadan kalkmayacağı görülmektedir. Bu durumda çarpışma tehlikesini ortadan kaldırmak amacıyla birinci robotun hedefine hiç gidilmemesi ve ikinci robotun hedefine de yakın olan robotun giderken diğer robotun beklemesi şeklinde bir yol izlenebilir. Bu çözümü gösteren durum Şekil 4.16'te verilmektedir. Şekil incelendiğinde, ikinci robot beklemekte olup kare ile gösterilmiştir. Birinci robot ise ikinci robotun hedefine doğru yol almakta olup, birinci robotun hedefine herhangi bir robot gitmeyecektir ve buna benzer şekilde hiçbir robotun gitmeyeceği düğümler X harfiyle ifade edilmiştir. Son olarak, bir hedefe, hedefin ait olduğu robot yerine diğer robotun geldiği durumu belirtmek için dörtgen(baklava dilimi) şekli kullanılmış ve ikinci robotun hedefi de bu şekilde gösterilmiştir.

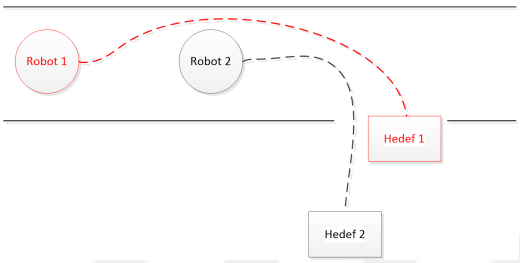


Şekil 4.15: Durum 3 Örneği

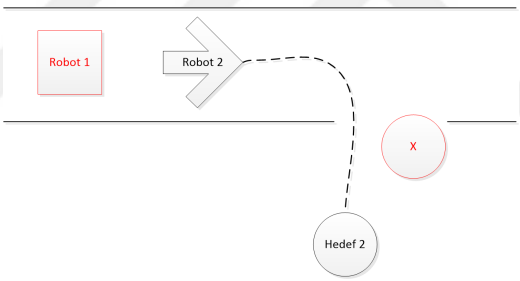


Şekil 4.16: Durum 3 Örneği için görsel sonuç

Şekil 4.17'te Durum 3'tekine benzer bir durum olmakla birlikte, bu sefer ikinci robot kendi hedefine yakın olmaktadır ve bu olay Durum 4 olarak tanımlanmıştır. Bu durum için, yine birinci robotun hedefine hiçbir robot gitmediği ancak ikinci robotun hedefine bu sefer Durum 3'ün aksine ikinci robotun yakın olmasından dolayı, ikinci robotun hareket ederken birinci robotun beklediği bir çözüm uygulanabilir. Bu çözümü anlatan görsel Şekil 4.18'te verilmiştir. Şekilde birinci robot beklediği için kare ile ve bu robotun hedefi ise X harfiyle gösterilmiştir.



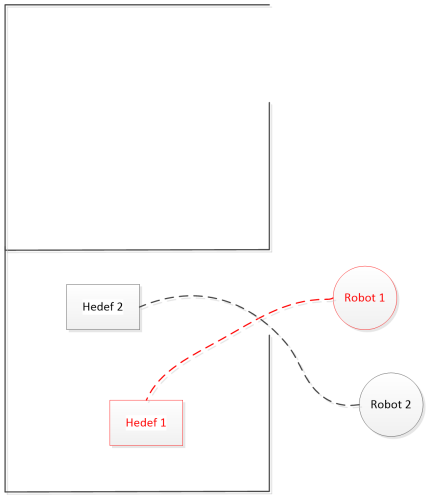
Şekil 4.17: Durum 4 Örneği



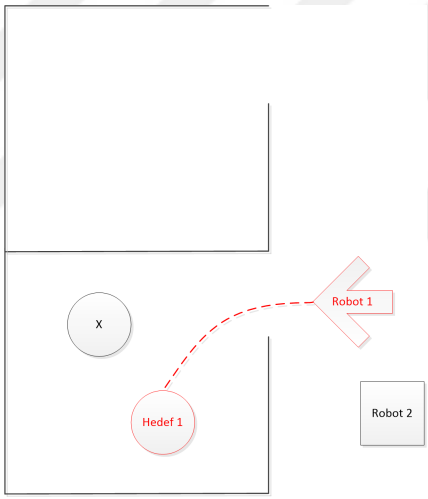
Şekil 4.18: Durum 4 Örneği için görsel sonuç

Durum 5 ve Durum 6, her iki robotun da ikinci robotun hedefine daha yakın olduğu durumlarda gözlemlenmektedir. Durum 3 ve Durum 4 ile benzer şekilde robotlardan herhangi birinin ikinci robotun hedefine gitmesi çarpışma tehlikesini ortadan kaldırmayabilir. Bu durumda ikinci robotun hedefine herhangi bir robotun gitmemesi ve iki robottan birinin durarak diğerinin birinci robotun hedefine gitmesi çarpışmayı önleyebilir. Bu olay ile karşılaşıldığında eğer birinci robot kendi hedefine yakınsa bu an Durum 5 olarak adlandırılmaktadır (Şekil 4.19).

Durum 5'teki çakışmayı çözmek için birinci robotun kendi hedefine gitmesi ve ikinci robotun da beklemesi gerekmektedir. Bu durumda oluşacak görsel Şekil 4.20'te verilmiştir. Şekilde ikinci robot beklediği için kare ile ve bu robotun hedefi ise X harfiyle gösterilmiştir.



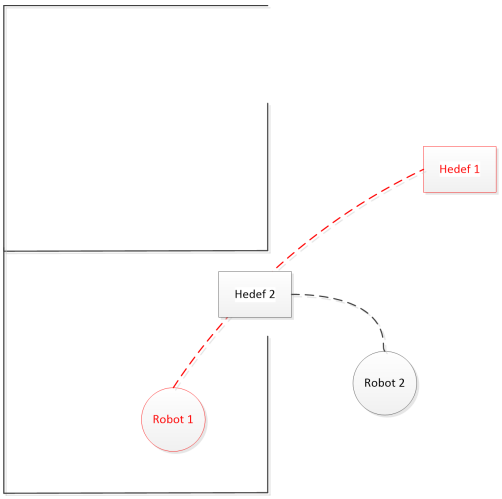
Şekil 4.19: Durum 5 Örneği



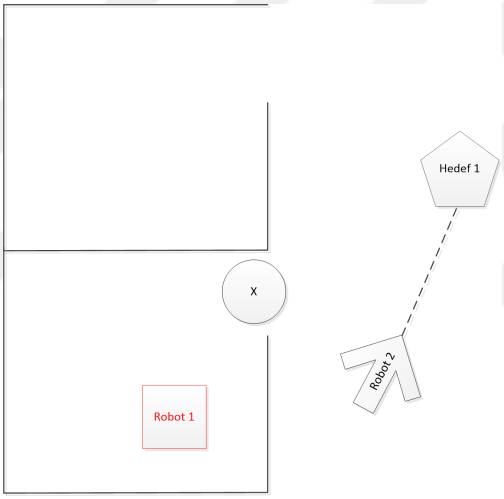
Şekil 4.20: Durum 5 Örneği için görsel sonuç

Şekil 4.21'de birinci robotun hedefine bu sefer Durum 5'in aksine ikinci robotun daha yakın olduğu durum gösterilmektedir ve bu olay Durum 6 olarak tanımlanmaktadır. Durum 6'daki çakışmayı çözmek için ikinci robotun birinci robotun hedefine gitmesi ve birinci robotun da beklemesi gerekmektedir. Bu durumda oluşacak görsel Şekil 4.22'te verilmiştir. Şekilde birinci robot beklediği için kare ile gösterilmiştir. İkinci robotun hedefi için X ve ikinci robotun birinci robotun hedefine gittiğini göstermek için beşgen şekli kullanılmıştır.

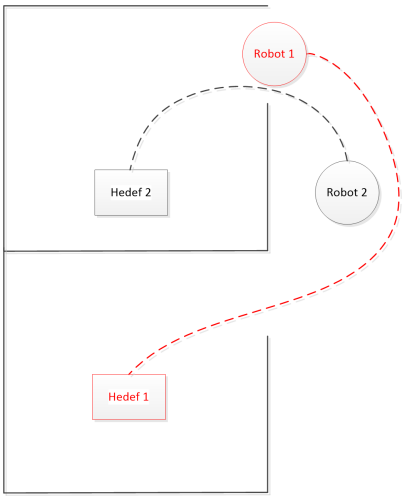
Son olarak, her iki robotun da birbirlerinin hedeflerine kendi hedeflerinden daha yakın olduğu olaylar Durum 7 olarak tanımlanmış ve Şekil 4.23 ve Şekil 4.24'te iki adet örneği gösterilmiştir.



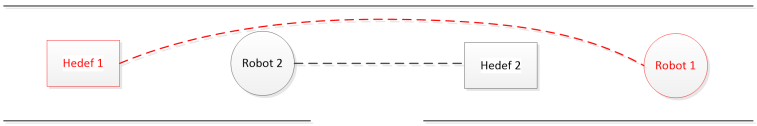
Şekil 4.21: Durum 6 Örneği



Şekil 4.22: Durum 6 Örneği için görsel sonuç

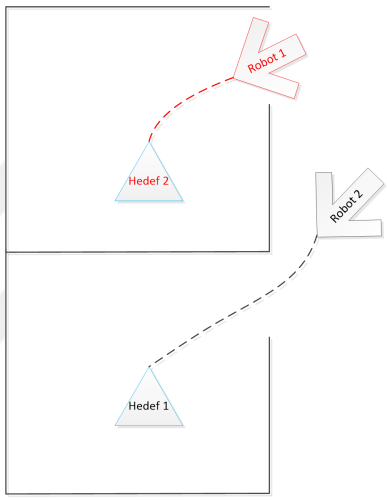


Şekil 4.23: Durum 7 Örnek 1



Şekil 4.24: Durum 7 Örnek 2

Örnekler incelendiğinde, uzaklıklara bakıldığında, birinci robotun ikinci robotun hedefine, ikinci robotun da birinci robotun hedefine hareket etmelerinin çarpışmayı önleyeceği düşünülmüştür. Şekil 4.23 ve Şekil 4.23'daki iki örnek için robotların hareketleri Şekil 4.25 ve Şekil 4.26 ile gösterilmiştir.



Şekil 4.25: Durum 7 Örnek 1 için görsel sonuç

Şekil 4.26: Durum 7 Örnek 2 için görsel sonuç

4.4 Rampa Algılama ve Hareket Düğümü Belirleme

Şekilde (Şekil 4.27), bu çalışmada kullanılan Gazebo benzetim ortamı ile yaratılmış afet ortamı gösterilmektedir. Şekil incelendiğinde, rampalar ile oluşturulmuş iniş ve çıkışların robotun hareketini zorlaştıracağı görülecektir. Bu nedenle, robotun seyrüseferine başarılı bir biçimde devam edebilmesi için iniş-çıkış gibi engellerin algılanabilmesi gerekmektedir. Çalışmada, bu amaç doğrultusunda robota bir rgb-d kamera eklenmesi düşünülmüştür.

Şekil 4.27: Örnek Afet Ortamı Haritası

Özellikle son yıllarda, mobil robotlar da dahil olmak üzere çeşitli platformda bir takım görevler için rgb-d kameraların kullanılması yaygınlaşmıştır. Bunlara örnek olarak görsel odometri ve seyrüsefer, engellerden sakınma, 3 boyutlu ortam haritalandırması gibi görevler gösterilebilir. (Suarez ve Murphy, 2012)'daki çalışmada, bir RGB-D algılayıcı olan Microsoft Kinect algılayıcısından bahsedilmiş, nasıl çalıştığına ve maliyeti düşük olduğu için robotik araştırmalarında kullanılmasının avantajlı olduğuna değinilmiştir. Ayrıca robot seyrüseferi, nesne manipülasyonu, ortam haritalaması gibi popüler görevler için kullanıldığına da değinilmiştir. Son olarak, içortamlarda arama ve kurtarma görevleri için de kullanılabilir olduğu belirtilmiştir.

(Huang vd., 2011)'daki çalışmada, mikro insansız hava aracının konumunun bulunması için görsel odometri yöntemlerinden faydalanılmıştır. Kullandıkları algoritmanın ilk adımında, RGB görüntüsünün işlenmesi bulunmaktadır. Algılayıcıdan alınan RGB resmi, siyah-beyaza çevrilmiş ve bir Gauss filtresinden geçirilerek düzeltilmiştir. Daha sonra da Gauss piramidi oluşturulmuştur. İkinci adımda, FAST algoritması kullanılarak elde edilen Gauss piramidinden özellikler çıkarılmıştır. Üçüncü adımda, farklı resimlerdeki özelliklerin karşılaştırılması kolaylaştırabilmek için hava aracının dönüşünün tahmin edilmesi yapılmaktadır. Bunun için de, şimdiki ve bir önceki resimlerin alt-örneklenmiş durumları arasındaki piksel hatalarının minimize edilmesinden faydalanmışlardır. Dördüncü adımda, çıkarılan özelliklerin her birine bir tanımlayıcı(descriptor) atanmış ve

farklı resimler arasında bu tanımlayıcılar karşılaştırılarak birbirine denk düşen özellikler bulunmuştur. Beşinci adımda, resimler arasındaki hatalı eşleşmelerin azaltılması için, aracın dönüşünün tahminine ek olarak, tutarlı eşleşmeleri ifade eden bir çizge yaratılmaktadır. Çizgenin oluşturulmasında, eşleşen özellikler arasındaki Öklid uzaklıkları kullanılmıştır. Eğer ardışık resimlerde, eşleşen özellikler arasındaki uzaklıklar fazla değişmiyorsa, bu eşleşmeler doğru kabul edilmiştir. Son adımda ise, eşleşen özellikler kullanılarak aracın hareketi tahmin edilmektedir. Görsel odometri ile yerel olarak tutarlı konum tahminleri elde edilmekle birlikte, buna ek olarak, metrik bir haritanın oluşturuabilmesi ve uzun zaman aralıklarında seyrüseferin yapılabilmesi için konum tahminlerinin küresel olarak da tutarlı olması gerekmektedir. Bunun için de yazarlar daha önceden yaptıkları bir RGB-D haritalandırma yöntemini kullanmışlardır.

(Bachrach vd., 2012)'daki çalışmada da, (Huang vd., 2011)'daki şekilde görsel odometri için RGB-D kamera kullanılmış ve aynı adımlara değinilmiştir. Buna ek olarak, görsel odometri yönteminde kullanılan algoritmalara alternatifler tartışılmış ve karşılaştırmalar sunulmuştur. Ayrıca, yazarlar geliştirdikleri BRM(Belief RoadMap) yol planlama algoritmasının da RGB-D kamera ile nasıl çalışacağını göstermiş ve lazer mesafe algılayıcısı kullanılan sonuçlarla karşılaştırmışlardır. Sonuç olarak, geliştirdikleri sistemin RGB-D kamerası kullanarak dağınık üç boyutlu ortamlarda yüksek bir durumsal farkındalık ile başarıyla karmaşık yol planları oluşturabildiğini belirtmişlerdir.

(Valenti vd., 2014)'daki çalışmada, bir insansız hava aracının otonom bir şekilde uçabilmesi için, dış algılayıcı olarak bir RGB-D kamera kullanılmak suretiyle geliştirilen bir sistem tasarlanmıştır. Aracın otonom hareketi ve seyrüseferi için RGB-D kameraya dayanan bir görsel odometri algoritması kullanılmıştır. Yazarların kullandıkları görsel odometri algoritması, kameradan gelen farklı resimler arasındaki dönüşümlerin eşleştirme yöntemleriyle bulunmasına dayanmaktadır. Görsel odometri ile elde ettikleri dönüşümleri de bir eşzamanlı konumlandırma ve haritalama(SLAM) algoritmasıyla birlikte kullanmışlardır.

(Leishman vd., 2014)'daki çalışmada, GPS'in çalışmayacağı bir ortamda havada seyrüsefer için bir yöntem önerilmiştir. Yöntemde, RGB-D kameradan elde edilen veriler görsel odometri, yol planlama ve çarpışmadan sakınma, topolojik harita oluşturma gibi görevler için kullanılmıştır. Robotun belirli mesafede bir aldığı resimler, anahtar resimler olarak kabul edilmiştir. Görsel odometri için bu anahtar resimler birbirleriyle eşleştirilerek hizalama yapılması amaçlanmıştır. Bunun için de FAST özellikleri ve BRIEF tanımlayıcıları kullanılmış ve eşleşen özellikler RANSAC algoritmasına sokularak bir konum hareket tahmini elde edilmiştir. Daha sonra da bir SVD(singular value decomposition) algoritması ile anahtar resimler arasındaki üç eksenli dönüş ve öteleme

değerleri bulunmuştur. Yol planlama ve çarpışmadan sakınma için, RGB-D kamerasından gelen veriler ile ortamın nokta bulutu elde edilmiş ve bu buluttan ortam için bir maliyet oluşturulmuştur. Bu maliyeti belirten matris kullanılarak da Dijkstra algoritması ile yol planlaması yapılmıştır. Buna ek olarak, işlemlerinde kullandıkları harita da bir topolojik harita olup haritadaki düğümler için RGB-D kameradan elde edilen anahtar resimler kullanılmıştır. Düğümler arasındaki ayrıtları için de görsel odometriden elde edilen anahtar resimler arasındaki dönüşümler kullanılmıştır.

(Henry vd., 2010) ve (Henry vd., 2012)'daki çalışmalarda, düşük maliyetli RGB-D kameralar kullanılarak bir üç boyutlu içortam modelleme yöntemi geliştirilmiştir. Yapılan çalışmalarda, temel olarak ilk aşamada ardışık veri resimlerinin uzaysal olarak hizalanması, ikinci aşamada döngülerin kapatılmasının tespit edilmesi, son aşamada da bütün veri serisinin küresel olarak hizalanması olmak üzere üç aşama bulunmaktadır. (Henry vd., 2010)'deki çalışmada, ardışık resimler arasındaki hizalama hem görünüş hem de şekil, yani hem derinlik hem de renkli resimler(rgb) kullanılarak yapılmaktadır. Görünüş tabanlı hizalama SIFT özellikleri üzerine RANSAC algoritması uygulanarak yapılmıştır. Hizalama için SIFT tabanlı eşleştirmelerin yapılmasının avantajı, bir ilk konuma ihtiyaç duymamasıdır, ancak yeterli sayıda özellik algılanmazsa hizalama başarısız olabilmektedir. Şekil tabanlı hizalama ise ICP(Iterative Closest Point) algoritması ile yapılmıştır. ICP ile eşleştirme yapılabilmesi için bağıl konumun iyi bir ilk tahminine ihtiyaç duyulmakla birlikte, bütün üç boyutlu verinin kullanılmasına imkan vermesi bu yöntemin bir avantajı olmaktadır. Son durumda resimlerin hizalanmasında hem RANSAC algoritmasından hem de ICP algoritmasından gelen özellikler kullanılmakta, böylece sistem renkli resim veya derinlik resminin herhangi birisinin hizalama yapmaya olanak vermemesi durumunda bile başarıyla çalışabilmektedir. Kapalı devrelerin algılanabilmesi için, elde edilen her bir veri resmi, tüm veri resimlerinin önceden belirlenen bir alt-kümesine üç boyutlu SIFT algoritması kullanılarak eşleşme olup olmadığına bakılmaktadır. Küresel olarak tutarlı hizalamaların elde edilebilmesi için de TORO isimli bir optimizasyon aracı kullanılmıştır. Bu aşamalar sonucunda elde edilen sistemin içortamları başarıyla gerçek zamanlı olarak haritalandırırken, özellik çıkartılamayan koridor veya tamamen karanlık odalar gibi uç durumları da işleyebildiği belirtilmiştir. (Henry vd., 2012)'daki çalışma, (Henry vd., 2010)'daki çalışmanın devamı niteliğinde olmakla birlikte, farklı olarak değişik bir çeşit RANSAC algoritması denenmiş, hizalamaların elde edilmesinde eşleşmeler için SIFT özellikleri(feature) ve tanımlayıcıları(descriptor) yerine FAST özellikleri ve BRIEF tanımlayıcıları kullanılmış, TORO yerine de SBA(sparse bundle adjustment) algoritması kullanılmıştır. Ek olarak, ICP algoritması ile elde edilen parametrelerin nasıl SBA algoritmasına entegre edilebileceğine değinilmiştir. Ayrıca döngülerin kapatılmasının belirlenmesinin verimi, ortam tanıma kullanılarak artırılmıştır. Sonuç olarak, algoritmaların hem derinlik hem de renkli resimlerin kullanıldığı şekilde tümleşik olarak değil de kendi

başlarına çalıştığı düşünüldüğünde, tek başına RANSAC algoritması, gözle görülür biçimde daha iyi sonuçlar vermiştir. Ancak derinlik algılayıcısının menzili dışında kalan yerler için özellikler çıkarılamayabilir, bu durumda da RANSAC kullanılamayabilir. Öte yandan, aydınlatmadan ötürü görsel tanımlayıcılar da yeterli olmayabilir. Bunlar gibi durumlarda tasarlanan tümleşik algoritmanın, tek başına çalışan bir algoritmaya göre daha iyi sonuçlar vereceği belirtilmiştir. Bunlara ek olarak, TORO ve SBA algoritmalarının da kullanımı karşılaştırılmış ve SBA'nın kesinliği daha yüksek eşleşmeler verdiği belirtilmekle birlikte, TORO'nun daha verimli olduğu ve yeni versiyonlarının daha iyi sonuçlar vermesinin muhtemel olduğu belirtilmiştir. Yazarlar çalışmalar sonucunda, RGB-D algılayıcılarıyla elde edilebilen renkli resimlerin ve derinlik resimlerinin birlikte kullanılmasıyla, sadece RGB-D algılayıcılar kullanılarak robot seyrüsefer ve etkileşim sistemlerinin tasarlanabileceğini söylemektedirler.

(Kerl vd., 2013)'daki çalışmada, gerçek zamanlı olarak kısıtlı hesaplama kabiliyetine ve düşük hafızaya sahip platformlarda kullanılabilecek bir görsel odometri yöntemi önerilmiştir. Çalışmada, RGB-D kamerasıyla elde edilen renkli resimler ile derinlik resimlerinden faydalanılmıştır. RGB-D kamerasından elde edilen yoğunluk ve derinlik resimleri kullanılarak, ardışık resimler üst üste bindirilmiş ve bu resimler arasındaki fotometrik hata minimize edilerek kameranın pozisyonu saptanmıştır. Ayrıca önceki çalışmalarına ek olarak, olasılıksal bir formülasyon ile hareketin bir önceki durumunu ve algılayıcı gürültüsünü de hesaba katarak, daha doğru bir şekilde hareket tahmini yapabildiklerini belirtmişlerdir.

(Lee ve Medioni, 2011)'daki çalışmada, görme engelli kişilerin hareketini kolaylaştırmak için RGB-D kamera tabanlı bir seyrüsefer sistemi geliştirilmiştir. Algılayıcıdan elde edilen RGB resimlerindeki köşeler FAST algoritması yardımıyla bulunmuştur. Daha sonra bulunan köşeler ve derinlik rsimlerinden elde edilen 3 boyutlu derinlik bilgileri kullanılarak ardışık resimler arasındaki görsel akış KLT takip algoritmasıyla((Lucas, Kanade, vd., 1981; Tomasi ve Kanade, 1991; Shi vd., 1994)) hesaplanmıştır. Daha sonra farklı resimler arasında başarıyla takip edilen FAST köşeleri bir RANSAC algoritmasına tabi tutulmuştur. Sonuç olarak kameranın hareketi bir görsel odometri yöntemiyle bulunmuş olmaktadır. Çalışmanın devamında, elde edilen hareket bilgisi kullanılarak bir SLAM algoritması yardımıyla konumlandırma yapılmıştır. Daha sonra 2 boyutlu bir ızgara haritası yardımıyla geçilebilir alanlar belirlenmiş ve D* algoritması ile de en kısa yol hesaplanmıştır.

(Aladren vd., 2014)'daki çalışmada, gerek görme engelli insanlar için gerekse görme engeli olmayan ancak görüşün problem oluşturduğu özel durumlarda çalışan insanlar için RGB-D kamera tabanlı bir seyrüsefer yardım sistemi önerilmiştir. Çalışmanın

öneminin, sadece derinlik verisi yerine renkli resimlerin de kullanılması olduğu belirtilmiştir. Bilinmeyen ortamlarda çarpışmasız bir şekilde ilerlemeyi mümkün kılacak bir yol planının oluşturulabilmesi için, yer düzleminin ve engellerin saptanması gerektiği belirtilmiş, bu amaç doğrultusunda yer düzlemi üzerine bölütleme yapılmıştır. Sonuç olarak, düşük maliyetli bir RGB-D kamera sitemi ile bilinmeyen bir ortamda güvenli şekilde seyrüsefer edilebilemsini sağlayan bir sistem sunulmuştur. Sistemin aydınlanmadaki değişimlere, parlamalara ve yansımalara karşı gürbüz olduğu da belitilmiştir.

(Maier vd., 2012)'daki çalışmada, insansı robotlar için, Asus Xtion Pro Live algılayıcısından elde edilen derinlik verileri kullanılarak konumlandırma, engel haritalandırma ve çarpışmadan kaçınma görevlerini yerine getirebilen bir seyrüsefer sistemi önerilmiştir. Ortamın gösterimi için octree tabanlı OctoMap haritalandırma yapısından faydalanmışlardır. Bu yapıda ortam dolu ve boş voksellere bölünmekte ve her voksel bir doluluk olasılığına sahip olmaktadır. Çalışmada ortam iki çeşit 3 boyutlu harita ile ifade edilmiştir. İlk harita ortamın statik haritasından oluşmakta ve konumlandırma ile birlikte yol planının ilk bilgisi olarak işlev görmektedir. İkinci harita ise yerel engelleri belirtmekte ve robot hareket ettikçe güncellenmektedir. Bu harita da statik olmayan engeller de hesaba katılabilecek şekilde yol planı yapmak için kullanılmaktadır. Ancak 3 boyutlu yol planı yapmak zor bir problem olduğundan, bu haritanın yer üzerindeki 2 boyutlu izdüşümü kullanılmıştır. Yol planının daha güvenli yapılabilmesi amacıyla algılayıcının yere doğru baktığı varsayılmaktadır. Ayrıca vokseller yere ait olanlar ve diğerleri olacak şekilde ayrılmıştır. Bu ayrımın yapılabilmesi için de ortamdaki baskın düzlemler, yerel komşuluklardaki normallere RANSAC uygulanarak belirlenmiştir. Ayrıca 2 boyutlu izdüşüm yapılırken, robotun z eksenindeki boyunun üstündeki alanlar hesaba katılmamıştır. Robotun 2 boyutlu izdüşüm haritasındaki seyrüseferi için de A* en kısa yol algoritması kullanılmıştır.

(Biswas ve Veloso, 2012)'daki çalışmada, gerçek zamanlı olarak ve düşük işlemci gereksinimiyle çalışan ve sadece bir RGB-D kameradan elde edilen derinlik verisini kullanan konumlandırma ve seyrüsefer yöntemi önerilmiştir. İlk olarak derinlik verisi ve kemaranın gerçek(intrinsic) parametreleri kullanılarak oluşturulan 3 boyutlu nokta bulutunun boyutu, önerilen Hızlı Örnekleme Düzlem Filtreleme yöntemi ile indirgenmiştir. Bu algoritmanın sonucu olarak filtrelenmiş noktalar ve onlara karşılık gelen düzlemlerin normalleri elde edilmektedir. Daha sonra bu veriler kullanılarak ortamı doğru parçaları olarak ifade eden vektör haritası oluşturulmuştur. Bu harita, ortamın haritasının ve robotun tahmini konumunun bilindiği varsayılarak, ortamdaki hangi doğru parçalarının robot tarafından görülebileceğinin hesaplanılmasında ve elde edilen filtrelenmiş noktaların bu doğru parçaları üzerine yansıtılmasında kullanılmıştır. Ayrıca ortamı ifade eden 3 boyutlu

nokta bulutundaki hem filtrelenmiş düzlemlere ait noktalar hem de düzlemlere ait olmayan dışlanan noktalar kullanılarak çarpışmadan sakınma görevi gerçeklenmiştir.

(Endres vd., 2012)'daki çalışmada, Kinect gibi RGB-D kameraların kullanılarak anlık konumlandırma ve haritalandırma yapabilmesi için bir yaklaşım önerilmiştir. Tasarlanan sistem, kameranın yörüngesini tahmin etmekle birlikte, ortamın iki boyutlu bir haritasını oluşturmaktadır. Önerdikleri yöntem, ön uç ve arka uç olarak ikiye bölünmüştür. Ön uçta, farklı gözlemler arasındaki ilişki, RGB-D algılayıcılardan gelen renkli resimler kullanılarak bulunmaktadır. Bu amaç doğrultusunda her resimden özellikler çıkarılmakta ve tanımlayıcılar oluşturulmakta, bunlar da farklı resimler için RANSAC ile eşleştirilmektedir. Bunun sonucunda da algılayıcının konumları arasındaki dönüşümler bulunmaktadır. Bunlara derinlik verisi de eklenerek ortamın nokta bulutu ortak bir koordinat sistemine göre oluşturulabilmektedir. Arka uçta ise, küresel olarak en uygun algılayıcı konumları, bir önceki aşamada elde edilen dönüşümler kullanılarak, çizge tabanlı bir optimizasyon yöntemi ile bulunmaktadır. Son adımda ise ortam bir voksel doluluk ızgarası ile ifade edilmektedir.

Yapılan literatür taramaları sonucunda, RGB-D kameraların çeşitli görevlerde başarıyla kullanılabildiği görülmüştür. Bu kameraların, hem renkli (RGB) resimler vermesi, hem de derinlik görüntüleri vermesi ve bu sayede de üç boyutlu nokta kümelerinin oluşturulabilmesine imkan vermesi araştırmalarda kolaylık sağlamaktadır. Bunlara ek olarak maliyetinin de düşük olması sebebiyle bu çalışmada da afet ortamlarındaki rampa gibi karşılaşılabilecek durumların saptanması ve seyrüseferin bu durumlar göz önüne alınarak yapılabilmesi için bir RGB-D kamera kullanılmıştır. Bu doğrultuda çalışma kapsamında ASUS Xtion Pro Live algılayıcısından yararlanılmıştır (ASUS, 2018).

Çalışmada, RGB-D kameradan gelen derinlik verilerinden nokta kümelerine çevirmek suretiyle faydalanılmıştır. Bu işlem için OpenNI ROS paketinden yararlanılmıştır (Mihelich, 2018). Derinlik resimlerinden nokta bulutları oluşturulurken, kameranın varsayılan gerçek(intrinsic) parametreleri kullanılmıştır.

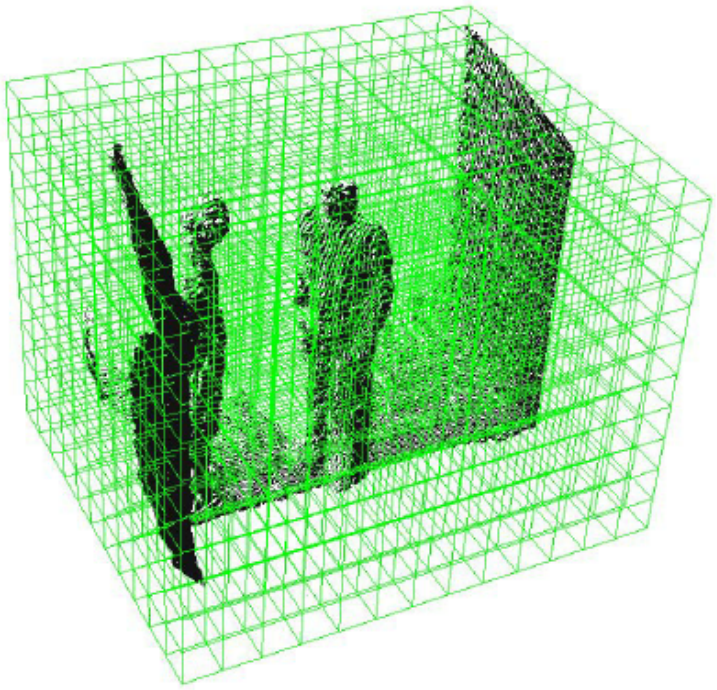
Elde edilen nokta bulutları ise PCL (Point Cloud Library) kullanılarak bir takım işlemlere tabi tutulmuştur. PCL, iki ve üç boyutlu resim ve nokta kümelerinin işlenmesi amacıyla oluşturulmuş bir kütüphanedir (Point Cloud Library (PCL), 2016). PCL yapısı, filtreleme, özellik tahmini, yüzey oluşturma, model oturtma, bölütleme gibi işlemleri yerine getirebilen pek çok sonteknoloji algoritmayı bünyesinde barındırmaktadır. Bu algoritmalar, üç boyutlu nokta kümelerini dikerek bir bütün oluşturmak, bir sahnedeki ilgili parçaları bölütlemek, ortamdaki nesnelerin geometrik şekillerine dayalı olarak tanınabilmesi

amacıyla anahtar noktaları çıkarmak ve tanımlayıcıları hesaplamak, nokta bulutlarından yüzeyler oluşturmak gibi pek çok görevde kullanılabilmektedir.

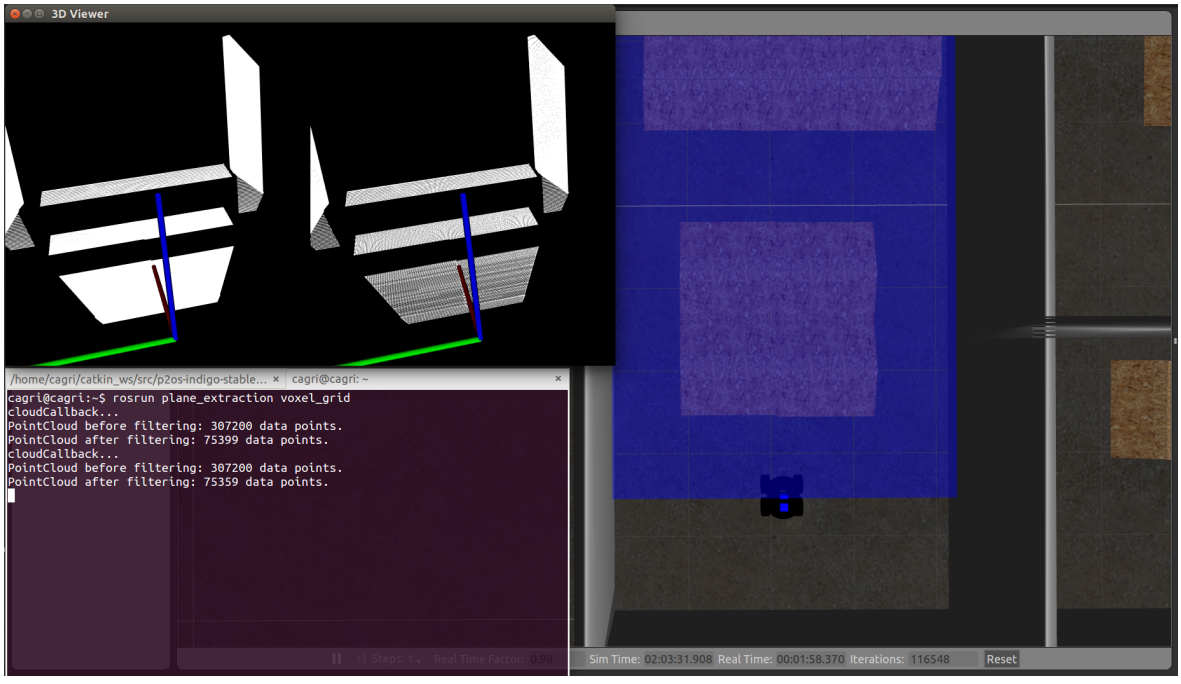
Yapılan bu tez çalışmasında da bahsedilen algoritmaların bazılarından, afet ortamındaki rampaların tespit edilmesi ve bu rampalar kullanılarak robotun başarıyla seyrüsefer yapabilmesine yardımcı olması amacıyla yerel yolnoktalarının(waypoint) hesaplanmasında faydalanılmıştır. Bu doğrultuda, robotun ve küresel hedefin konumu ile birlikte RGB-D kamerası ile elde edilen nokta kümesi işlenmektedir. Sonuç olarak ise robotun hareketini kolaylaştıracak bir yerel yolnoktası elde edilmesi amaçlanmıştır.

OpenNI paketi yardımıyla nokta bulutu elde edildikten sonnra, ilk aşamada voksel ızgarası filtreleme yöntemi ile nokta bulutu altörneklenmektedir. Bu noktada, kısaca voksellerden bahsetmekte fayda vardır. (Gebhardt vd., 2009)'daki tanıma göre, bir vokseli, üç boyutlu bir hacim pikseli olarak düşünebiliriz. Bunun yanında, voksellerin konumunun XYZ kartezyen koordinatlarından ziyade, diğer voksellere göre olan bağıl konumunun göz önüne alınarak yapıldığı belirtilmektedir. Yazının devamında da üç boyutlu bir modelin vokseller kullanılarak voksel ızgaraları, seyrek voksel modelleri veya sekizli ağaç(octree) voksel modelleri gibi farklı şekillerde örneklenebileceğine değinilmiştir. Örnek bir voksel ızgarası Şekil 4.28'de gösterilmiştir. Bu çalışmada da, PCL kütüphanesinde tanımlanmış voksel ızgaralama yöntemi kullanılarak, girdi olarak alınan nokta bulutuna bir üç boyutlu voksel ızgarası oturtulmuştur. Her bir vokselin içinde kalan noktaların ağırlık merkezlerine göre yaklaşık değerleri alınmaktadır. Böylelikle, bütün bir nokta bulutunun işlenmesi yerine voksellerdeki yaklaşık noktalar alınarak altörnekleme yapılmış ve bu sayede de işlenecek nokta sayısı düşürülerek algoritmaların çalışması hızlandırılmış olmaktadır. Bu noktada, noktaların yaklaşık değeri bulunurken vokselin merkezi yerine ağırlık merkezlerinin kullanılmasının, biraz daha yavaş bir yöntem olmasına rağmen nokta bulutları ile ifade edilen bölgenin daha iyi ifade edilmesini sağladığı belirtilmiştir (Point Cloud Library (PCL), 2016).

Şekil 4.29'de nokta kümelerinin gösterildiği pencerelerde, sol tarafta orijinal nokta kümesi, sağ tarafta ise voksel ızgarası filtresi ile altörneklenmiş hali görünmekle birlikte, voksel ızgarası ile altörnekleme yapılması ile nokta bulutu sayısındaki azalma da terminal penceresinde gösterilmiştir. Burada, kameranın merkezinden(orijin) uzaklaştıkça, bir voksel hücresine daha az sayıda nokta düştüğü, bu nedenle de filtrelenmiş durumda uzaktaki bölgelerdeki altörneklenmiş noktalar arasında daha fazla mesafe bulunduğu gözlemlenmiştir.

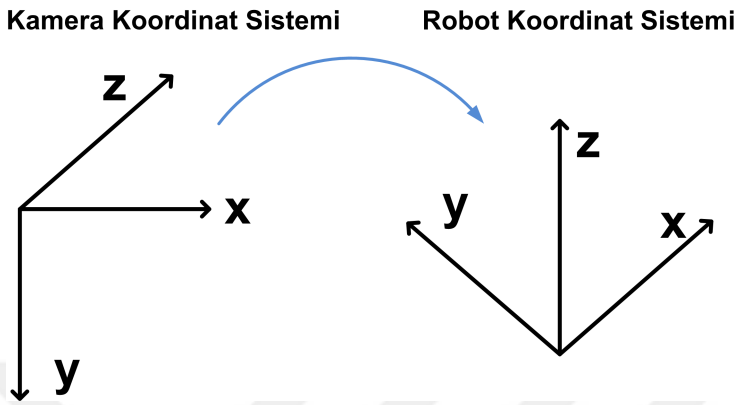


Şekil 4.28: Voksel Izgarası Örneği (Wang vd., 2012)



Şekil 4.29: Voksel Izgarası Uygulanmış Nokta Bulutu Örneği

Ortamın nokta bulutu voksel ızgara filtresi ile altörneklendikten sonra, nokta bulutunun koordinat sistemine dönüşüm uygulanarak robotun koordinat sistemiyle aynı olması sağlanmaktadır. Bu işlemi anlatan resim Şekil 4.30'de gösterilmiştir.



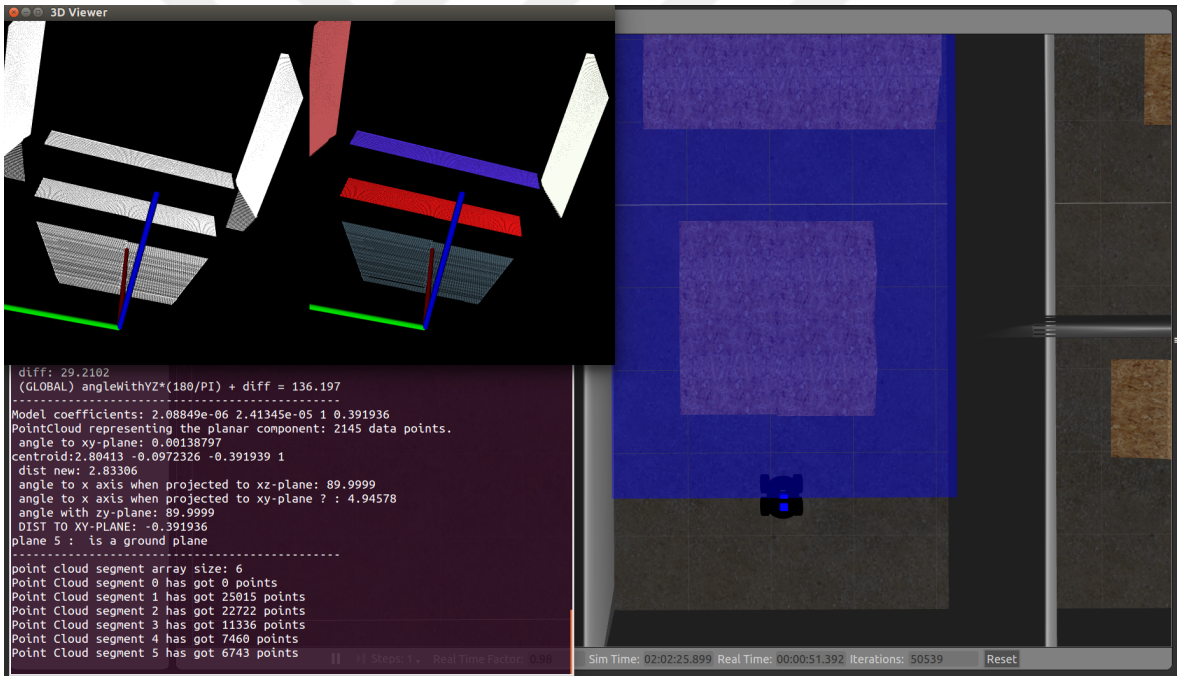
Şekil 4.30: Kamera Koordinat Sistemi - Robot Koordinat Sistemi Dönüşümü

Koordinat sistemine rotasyon uygulandıktan sonra, PCL kütüphanesindeki düzlem bölütleme algoritması kullanılarak ortamdaki düzlemler belirlenmektedir. Düzlemlerin belirlenmesinde, RANSAC(Random Sample Consensus) algoritmasından faydalanılmaktadır. RANSAC algoritması, aykırı değerler de bulunan bir veri kümesine, tekrarlamalı bir şekilde bir matematiksel modelin oturtulması için kullanılan bir yöntemdir. RANSAC algoritması, ilk olarak Fischler ve Bolles tarafından yayınlanmıştır (Fischler ve Bolles, 1981). Yazarlar, algoritmanın diğer düzleştirme(smoothing)/parametre tahmin etme algoritmalarından farklı olarak, mümkün olan en çok sayıda veri kullanarak bir ilk değer bulduktan sonra geçersiz olan verilerin elenmesi yerine, mümkün olan en küçük veri kümesiyle başladığını ve bu kümeyi uygun veriler buldukça genişlettiğini belirtmektedirler.

RANSAC algoritması, verilen veri kümesinde hem uyumlu(inlier) hem de aykırı(outlier) verilerin olduğunu varsayarak, bu veri kümesine en uygun modelin parametre tahminlerinin yapılması için bir oylama düzeni kullanır (Fischler ve Bolles, 1987). Veri kümesindeki her bir eleman bir veya birden fazla modele oy verebilir. Bu oy verme düzeni iki varsayıma dayandırılmıştır. Bunlardan ilki, aykırı elemanların sürekli aynı modele oy vermeyecek olmalarıdır. İkincisi de, iyi bir modelde karar kılınabilmesini mümkün kılacak sayıda verinin bulunmasıdır. Bu varsayımlar ışığında, RANSAC algoritmasının tekrarlamalı olarak çalışan iki ara adımda çalıştığı söylenebilir. İlk adımda, veri kümesinden mümkün olan en az sayıda eleman içeren bir altküme seçilerek örnek altküme olarak seçilir. Sadece bu örnek altkümedeki elemanlar kullanılarak bu elemanlara uygun bir model ve bu modelin parametreleri elde edilir. İkinci adımda ise algoritma, veri

kümesindeki bütün elemanların elde edilen bu modele uygunluğunu bir hata eşik değeri kullanarak kontrol eder. Eğer kontrol edilen veri elemanı örnek modele uymuyorsa aykırı olarak kabul edilir. Modele uyumlu elemanların oluşturduğu kümeye ise fikir birliği kümesi(consensus set) adı verilir. Algoritma, elde edilen fikir birliği kümesinde yeteri kadar sayıda uyumlu eleman bulunana kadar bahsedilen iki adımı tekrar eder.

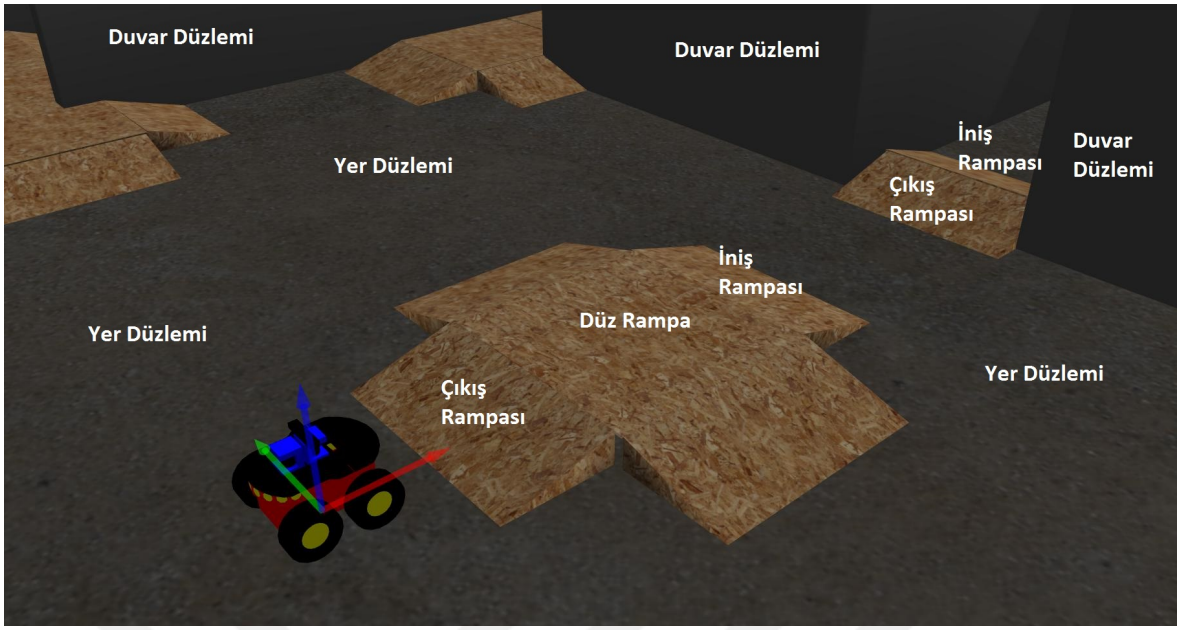
Düzlem bölütleme için kullanılan algoritma da nokta kümesi verisindeki düzlemleri saptayabilmek için RANSAC algoritmasından yararlanmaktadır. Burada, elde edilen düzlemler $ax + by + cz + d = 0$ denklemi göz önüne alınarak a, b, c ve d parametreleriyle ifade edilmektedir. Düzlem bölütleme algoritması, her adımda en uygun fikir birliği kümesiyle ifade edilen düzlem modeline uygun uyumlu noktaları döndürerek, bu işleme bütün nokta kümesi verisinin belirli bir yüzdesi geriye kalana kadar devam eder. Örnek bir düzlem bölütlenmesi uygulanmış ortam resmi Şekil 4.31'de görülebilmektedir.



Şekil 4.31: Örnek Düzlem Bölütleme Gösterimi

Nokta kümesi verisindeki düzlemler elde edildikten sonra, elde edilen düzlemler iniş rampası, çıkış rampası, düz rampa, duvar düzlemi ve yer düzlemi olmak üzere beş ana sınıfa ayrılmaktadır. Robotun bakış açısına göre bu sınıfları ifade eden durumlar Şekil 4.32'de gösterilmiştir.

Bir düzlemin iniş-çıkış rampası mı, duvar düzlemi mi yoksa düz rampa veya yer düzlemi mi olduğuna karar vermek için, o düzlemin z eksenindeki değerinin xy düzlemiyle



Şekil 4.32: Robotun Bakış Açısına Göre Rampaların Sınıflandırılması

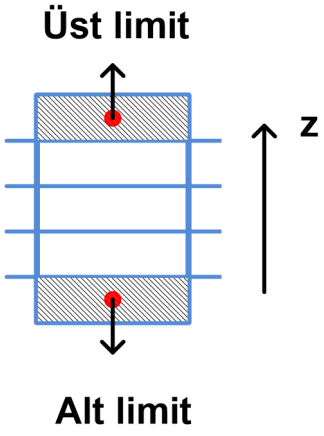
olan açısına bakılmaktadır. Eğer bu açı 90^o civarında yani dik bir açıda ise, bu düzlem bir duvar düzlemidir denilmektedir. Eğer düzlemin açısı 0^o civarında ise, bu düzlem ya yer düzlemidir ya da düz rampadır denilmektedir. Yer düzlemi mi yoksa düz rampa mı olduğuna karar vermek için ise, düzlemlerin kamera merkezine olan dik uzaklıklarına bakılmaktadır. Düz rampa ile ifade edilen düzlem, yerden yüksekte olup iniş ve çıkış rampaları arasında kalan düz alanları ifade ettiğinden dolayı, robot yerde de olsa bir rampa üzerinde de olsa, kameraya olan dik uzaklığı yer düzlemine göre daha az olmaktadır. Bunlara ek olarak, bir düzlemin iniş veya çıkış rampası mı olduğunu anlamak için, ilk olarak rampanın xy düzlemiyle yaptığı açının belirli bir aralıkta olup olmadığına bakılmaktadır. Bunun sonucunda düzlemin bir eğimli rampayı belirttiği anlaşılmaktadır. Bu rampanın iniş mi yoksa çıkış mı olduğu belirlenirken ise, rampanın xz düzlemine yansıması alınmakta ve bu yansıma sonucu elde edilen doğru parçasının x ekseniyle arasındaki açıya bakılmaktadır. Eğer bu açı 90^o'den büyükse düzlem bir çıkış rampası, küçükse de bir iniş rampasıdır denilmektedir.

Bütün düzlemlerin sınıfları belirlendikten sonra, düzlemlerin model parametreleri kullanılarak ağırlık merkezleri hesaplanmakta ve bu ağırlık merkezlerine olan öklid uzaklıkları hesaplanmaktadır. Daha sonra düzlemler uzaklıklarına göre yakından uzağa sıralanmaktadırlar. Sonraki adımlarda, eğer düzlemlerin kamera merkezine olan uzaklıkları $enBuyukMumkunUzaklk$ olarak adlandırılan parametreden büyük ise, bu düzlemler hesaplamaya dahil edilmemekte, böylelikle yolnoktası oluşturma algoritmasının çalışma süresinin düşürülmesi amaçlanmaktadır.

Rampa saptanması ile yolnoktası oluşturma algoritmasının bundan sonraki adımlarında, ortamdaki düzlemlerden iniş veya çıkış rampası olarak sınıflandırılanlar ile düz rampalar olarak sınıflandırılanların kullanılarak nasıl yolnoktaları elde edildiği açıklanacaktır. Bu olayın daha düzenli bir biçimde anlatılabilmesi açısından, sıradaki aşamalar üç ana başlık içerisinde incelenecektir. Bunlardan ilkinde, çıkış veya iniş rampası olarak sınıflanan düzlemlere ne gibi işlemlerin uygulanarak yolnoktası elde edildiği anlatılacak, ikinci başlıkta ise düz rampaların bulunduğu durumlardaki yolnoktalarının elde edilmesinden bahsedilecektir. Üçüncü başlığa gelindiğinde, ortamdaki $enBuyukMumkunUzaklk$ ile tanımlanan uzaklık içinde kalan rampalar işlenmiş ve (eğer bulunduysa) yerel yolnoktaları elde edilmiş olacaktır. Haliyle üçüncü adımda elde edilen bu yolnoktalarının nasıl bir karar verme aşamasından geçtiğinden, yani hangisinin ve nasıl hareket etmek üzere seçildiğinden bahsedilecektir. Bu üç aşamadan sonra ise rampa saptanması ile yolnoktası oluşturma algoritmasının çıktısının robotun seyrüseferi için nasıl kullanıldığından bahsedilecektir.

4.4.1 İniş/Çıkış (Eğimli) Rampalarında Yolnoktası Oluşturma

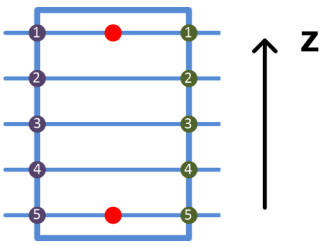
Algoritmanın bu bölümü, kamera merkezine göre olan uzaklıklarına göre sıralanan rampalar ele alınırken, işlenecek olan rampanın iniş veya çıkış rampası olarak sınıflanmış olması ve uzaklığının $enBuyukMumkunUzaklk$ parametresinden küçük olması durumunda çalıştırılmaktadır. İlk olarak, rampayı ifade eden nokta bulutunun her elemanı, z eksenindeki değerlerine göre küçükten büyüğe sıralanmaktadır. Sıralanmış bulutun, başından ve sonundan $egimliRampaBolme$ parametresine göre belirlenen oranlarındaki bölümlerinin ayrı ayrı ortalamaları alınmaktadır. Bu işlem sonucunda, eğimli rampanın üst ve alt limiti olarak kullanılacak olan iki tane limit noktası belirlenmiş olmaktadır (Şekil 4.33).



Şekil 4.33: Egimli Rampa Üst-Alt Limitlerinin Belirlenmesi

Bu noktada, üst ve alt limitlerin nasıl kullanılacağına değinmek gerekmektedir. Bu bölümdeki genel amaç, rampa üzerinde bir yolnoktasının oluşturulmasıydı. Bunun gerçekleşebilmesi için de, rampanın genel şeklinin kullanılmasının uygun olacağı düşünülmüştür. Ancak, robotun bakış açısı, rampa ile robot arasında bir takım engellerin bulunabilecek olması gibi nedenlerle rampayı ifade eden nokta bulutu kesin bir geometrik şekle (dikdörtgen gibi) sahip olamayabilmektedir. Bu gibi durumlar rampanın şeklini kullanabilmemizi sağlayacak kenar ve köşe gibi özelliklerin bulunmasını zorlaştırmaktadır. Bu nedenle, rampa üzerinde, nokta bulutu ile temiz bir görüntü alınamasa da rampanın şeklini kullanabilmemizi sağlayacak bir takım noktaların belirlenmesi amaçlanmıştır. Bu bağlamda, ilk olarak yukarıda anlatılan şekilde rampanın üst ve alt limitlerini belirtecek iki nokta elde edilmiştir. Eğimli rampanın üst ve alt limitleri belirlendikten sonra, bu limitler arasında kalan rampa düzleminin farklı z ekseni değerlerine sahip xy düzlemine paralel düzlemler ile kesişimlerinin bulunması amaçlanmıştır. Böylelikle, kesen düzlemler ve rampa düzlemlerinin kesiştiği yerlerde doğru parçalarını ifade eden nokta bulutları elde edilecektir. Bu doğru parçalarının da sağ ve sol olarak düşünülebilecek iki ucu hesaplanarak rampanın kenarlarını ifade edebilecek bir dizi nokta bulunacaktır.

Bulunacak olan noktalar, rampaya önden bakıldığı düşünüldüğünde rampanın görünen alanının sağ ve sol kenarları üzerine denk düşecektir. Rampanın kenarları yerine kullanılan bu noktalar uçnoktalar olarak adlandırılmıştır. Rampalar üzerinde kaç tane uçnoktanın hesaplanacağı, yani kaç adet xy düzlemine paralel düzlem ile kesişimin alınacağı, üst ve alt limitler arası fark ve $egimliRampaBolme$ parametresi kullanılarak bulunan bir artış oranına göre belirlenmektedir. Kısaca, alt limitten üst limite doğru, her adımda artış oranı kadar z ekseni değeri değiştirilmekte ve bu değerdeki kesişimler hesaplanmaktadır. Uçnoktaların bulunması için de, kesişimleri ifade eden doğru parçalarına ait nokta bulutundaki birbirine en uzak iki nokta bulunmaktadır (4.34).



Şekil 4.34: Egimli Rampa Uçnoktalarının Belirlenmesi

Bu aşamada, uçnoktaları bulunan rampanın robot tarafından geçilebilir olup olmadığının belirlenmesi amacıyla bir takım işler yapılmaktadır. Bu amaçla, rampanın küresel yönelim açısı ile robotun yönelim açısı arasındaki farka bakılmaktadır. Robotun

küresel yönelim açısı bulunurken, afet ortamındaki eğimli rampaların ortamın referans koordinat eksenine göre dik veya yatay biçimlerde bulundukları varsayılmıştır. Örnek afet ortamını gösteren resim incelendiğinde de (Şekil 4.27) eğimli rampaların bu varsayımı destekleyecek şekilde yerleştirildiği görülecektir.

Eğimli rampanın küresel yönelim açısı hesaplanırken, rampanın kenarlarını ifade eden uçnoktalardan faydalanılmaktadır. Ancak bazı durumlarda, bu uçnoktaların bazıları ortamın dağınık(cluttered) olması, düzlemlerin bölütlenmesinde oluşabilcek küçük hatalar vb. nedenlerle rampanın kenarından farklı yerlerde oluşabilir. Rampanın küresel açısı hesaplanırken de bütün uçnoktalar kullanılacağından, bu durumda hatalı sonuçlar elde edilebilir. Bunun engellenmesi için, elde edilen uçnoktalara ”aykırı düzeltme” adı verilen bir algoritma uygulanmaktadır. Algoritmada, her bir kenardaki ardışık uçnoktalar arası uzaklıkların ortalaması bulunmaktadır. Eğer herhangi ardışık uçnokta arasındaki uzaklık, bulunan ortalama uzaklıktan fazla olursa, bu uçnoktalar aykırı olarak ayrılmaktadır. Aykırı uçnoktalar da, aykırı olmayan uçnoktaların üzerine uydurulan bir doğru parçası üzerine yansıtılmaktadır. Böylelikle uçnoktaları rampaların görünen kısımlarının kenarlarını doğru olacak ifade edecek biçimde hizalanmış olmaktadır. Kullanılan aykırı düzeltme fonksiyonu, Algoritma 2’da gösterilmektedir.

Uçnoktalar aykırı düzeltme algoritması ile hizalandıktan sonra, ”küresel rampa yönelimi hesaplama” adı verilen bir algoritma kullanılarak küresel yönelim açısı bulunmaktadır. Algoritma, girdi olarak hizalanmış uçnoktaları ve robotun yönelim açısını almakta, çıktı olarak ise rampanın küresel yönelim açısını döndürmektedir. Algoritmada ilk aşamada, gelen uçnoktalar robotun yönelim açısına göre döndürülerek, yerel koordinatlardan küresel koordinatlara çevrilmektedir. Sonraki adımda ardışık uçnoktalar arasındaki açılar hesaplanmaktadır. Bu hesap yapılırken, ardışık iki uçnoktanın birinden diğerine tanımlanan bir vektör kullanılarak açı hesabı yapılmaktadır. Bu noktada, iniş ve çıkış rapması arasında bir fark açığa çıkmaktadır. Tanımlanan vektörün başlangıç ve bitiş noktasının konumları, hesaplanan açıyı değiştirmektedir. Haliyle, eğer rampa bir iniş rampası ise, tanımlanan vektörün başlangıç notkası ardışık uçnoktalardan z eksenindeki değeri büyük olanı olurken, çıkış rampasında ise z değeri küçük olan olmaktadır. Yani açı hesaplamasında kullanılan vektör çıkış rapması için aşağıdan yukarı, iniş rampası için yukarıdan aşağı olmaktadır.

Ardışık uçnoktalar arasındaki açılar belirlendikten sonra, bu açıların ortalamaları hesaplanmaktadır. Bu ortalamaların da, rampaların yöneliminin referans koordinat sistemine göre dik veya yatay olabileceği varsayımı da göz önünde bulundurularak, 0^o, 90^o, 180^o veya 270^o açılarından hangisine yakın oldukları belirlenmektedir. Bu noktada yönelim açılarının değerlerine göre dört tane durum oluşmaktadır. Durum 1’de ortalama açının 0^o’ye

Algorithm 2: İniş-çıkış rapması aykırı nokta düzeltme algoritması

Girdi: Düzlemin kenarlarını belirten uçnoktalar vektörü UN
Çıktı : Düzlemin kenarlarını belirten hizalanmış uçnoktalar vektörü HUN
Veri: Uçnoktaları belirten vektörün boyutu un_boyut, ardışık uçnoktalar arası uzaklıklar vektörü $ardUzak$, ardışık uçnoktalar arası uzaklıklar vektörü boyutu $ardUzak\_boyut$, aykırı uçnoktaların indislerinin tutulduğu vektör $aykirilar$, aykırıların indekselerinin tutulduğu vektörün boyutu $aykiri\_boyut$

$HUN \leftarrow UN$
`// Ardışık uçnoktalar arası ortalama uzaklığın hesaplanması`
for $i \leftarrow 0$ **to** $un\_boyut - 1$ **do**
  $topUzak \leftarrow topUzak + oklid\_uzaklik(UN(i), UN(i+1))$
  $ardUzak \leftarrow oklid\_uzaklik(UN(i), UN(i+1))$
end
$ortUzak = topUzak\ /\ (un\_boyut - 1)$ `// aykırı olan uçnoktaların bulunması`
for $i \leftarrow 0$ **to** $ardUzak\_boyut$ **do**
  if $ardUzak(i) > ortUzak$ **then**
    $aykirilar \leftarrow ardUzak(i)$'nin iki ucuna denk gelen uçnoktalar
  end
end
`// aykırı olan uçnoktaların, aykırı olmayan uçnoktaların üzerine oturtulan doğruya yansıtılması`
for $i \leftarrow 0$ **to** $aykiri\_boyut$ **do**
  $HUN(aykirilar(i)) \leftarrow dogruya\_yansit(UN(aykirilar(i)))$
end
return HUN

yakınlığı, Durum 2'de 90^o'ye yakınlığı, Durum 3'te 180^o ve Durum 4'te de 270^o'ye olan yakınlığı hesaplanmaktadır. Dört durum için elde edilen fark değerlerinin en küçüğü bulunmakta ve bu durumda rampanın yönelimi bu en küçük değerin ifade ettiği yöndedir denilmektedir. Bahsedilen rampanın küresel yönelim açısının hesaplamasında kullanılan algoritma için Algoritma 3 incelenebilir.

Rampanın küresel yönelim açısı belirlendikten sonra, robotun küresel yönelim açısı ile arasındaki farka bakılmaktadır. Eğer bu fark $aciFarkiParam$ ile adlandırılan parametreden büyükse, eğimli rampa robotun hareket yönüne göre fazla diktir denilmekte ve robot bu rampa üzerinden geçemez denilmektedir. Bu durumun da algoritmanın devamındaki karar verme aşamasında hesaba katılabilmesi amacıyla $rampaGecisBayrak$ isimli bir mantıksal değer kullanılmaktadır.

Eğimli rampa ile ilgili son olarak işlenmekte olan rampa için yolnoktası bulunmasıyla ilgili hesaplamalar yapılmaktadır. Bu amaçla ilk olarak, farklı z değerlerine sahip düzlemlerle kesilerek bulunan her birbirine en uzak konumdaki uçnokta çiftinin

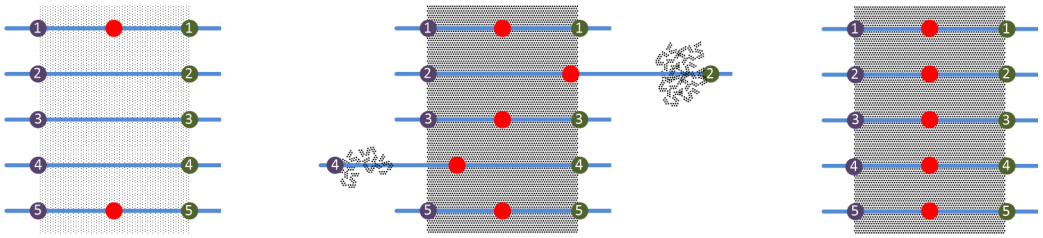
Algorithm 3: İniş-çıkış rampası küresel yönelim açısı hesaplama algoritması

Girdi: Düzlemin kenarlarını belirten uçnoktalar vektörü YUN ve robotun yönelim açısı th
Çıktı : Rampanın küresel yönelim açısı aci
Veri: Küresel koordinata çevrilmiş uçnoktalar vektörü KUN, uçnoktaları belirten vektörün boyutu un_boyut

```
// uçnoktaların yerel koordinat sisteminden küresel koordinat sistemine
   çevirimi
```

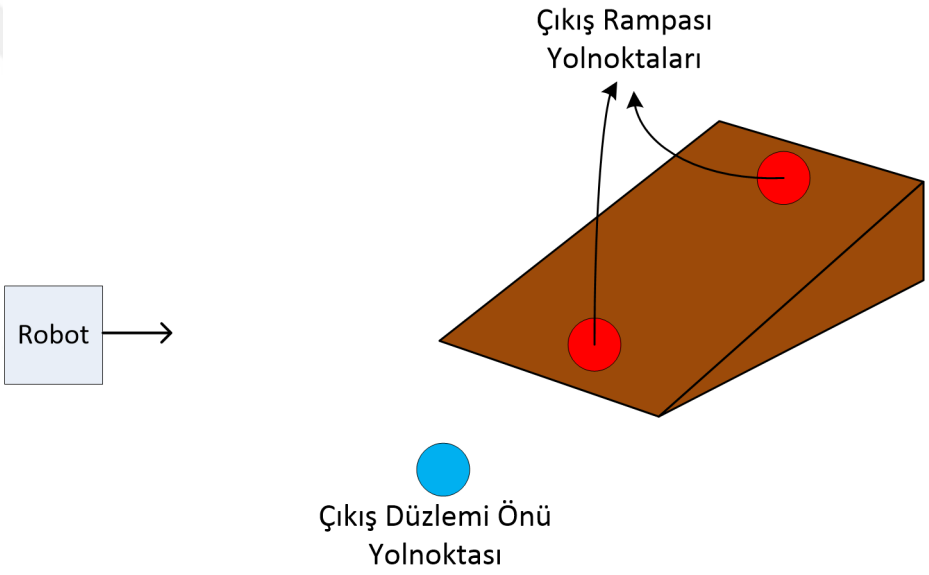
for $i \leftarrow 0$ **to** un_boyut **do**
  $KUN(i)x = YUN(i)x * \cos(th) - YUN(i)y * \sin(th)$
  $KUN(i)y = YUN(i)x * \sin(th) + YUN(i)y * \cos(th)$
  $KUN(i)z = YUN(i)z$
end
for $i \leftarrow 0$ **to** $un\_boyut - 1$ **do**
  $topA \leftarrow topA + |\arctan(KUN(i+1) - KUN(i))|$
end
$ortA \leftarrow topA \,/\, un\_boyut - 1$ `// Durum 1 Rampa` 0^o`'ye yakın`
$d1 = abs(0 - ortA)$ `// Durum 2 Rampa` 90^o`'ye yakın`
$d2 = abs(90 - ortA)$ `// Durum 3 Rampa` 180^o`'ye yakın`
$d3 = abs(180 - ortA)$ `// Durum 4 Rampa` 270^o`'ye yakın`
$d4 = abs(270 - ortA)$ `// En yakın olduğu durumu bul`
$enYaknDurum \leftarrow minimum(d1, d2, d3, d4)$
$aci \leftarrow$ En yakın olduğu durumun açısı
return aci

ortalaması alınır. Böylelikle, alt limit ile üst limit arasında bir dizi yolnoktası adayı belirlenmiş olur. Ancak, bu yolnoktalarının hesaplanmasında kullanılan uçnoktalar, eğimli rampa ile ilgili işlemlerin yapıldığı algoritmanın akışından dolayı henüz aykırı nokta düzeltme ile hizalanmamış olduklarından, aykırı olabilecek uçnoktaların bulunması durumunda elde edilen yolnoktası da istenmeyen bir konumda bulunuyor olabilir. Bu nedenle yolnoktası adaylarına da aykırı nokta düzeltme algoritması uygulanarak, yolnoktalarının rampa kenarlarındaki uçnoktalara benzer şekilde hizalanması sağlanmaktadır (Şekil 4.35).



Şekil 4.35: Egimli Rampa Uçnoktalarının ve Yolnoktalarının Aykırı Nokta Düzeltme Algoritması ile Hizalanması

Hizalanma aşamasından sonra, yolnoktası dizisindeki z ekseni değerleri en düşük ve en yüksek olan iki nokta daha sonra değerlendirilmek üzere $sonYolnoktasiVektoru$ olarak adlandırılan bir vektöre kaydedilmektedir. Buna ek olarak, eğer işlenmekte olan rampa bir çıkış rampası ise, rampanın yönelimi göz önüne alındığında rampanın hemen önüne denk gelecek fazladan bir yolnoktası da $cikisDuzlemiOnuYolnoktasi$ ismindeki parametreye kaydedilmektedir. Bu ek yolnoktasının hesaplanmasındaki amaç, robotun çıkış rampasından geçişini kolaylaştıracak, hatta bazı durumlarda açısal olarak geçilemeyecek olarak bulunsa bile bu yolnoktasının kullanımıyla geçilebilmesini sağlayacak bir nokta oluşturmaktır. Böylelikle her çıkış rampası üzerindeki yolnoktası ile birlikte kendine ait $cikisDuzlemiOnuYolnoktasi$ parametresi de sonradan değerlendirilmek üzere kaydedilmektedir (Şekil 4.36).



Şekil 4.36: Çıkış Rampası Önü Yolnoktası

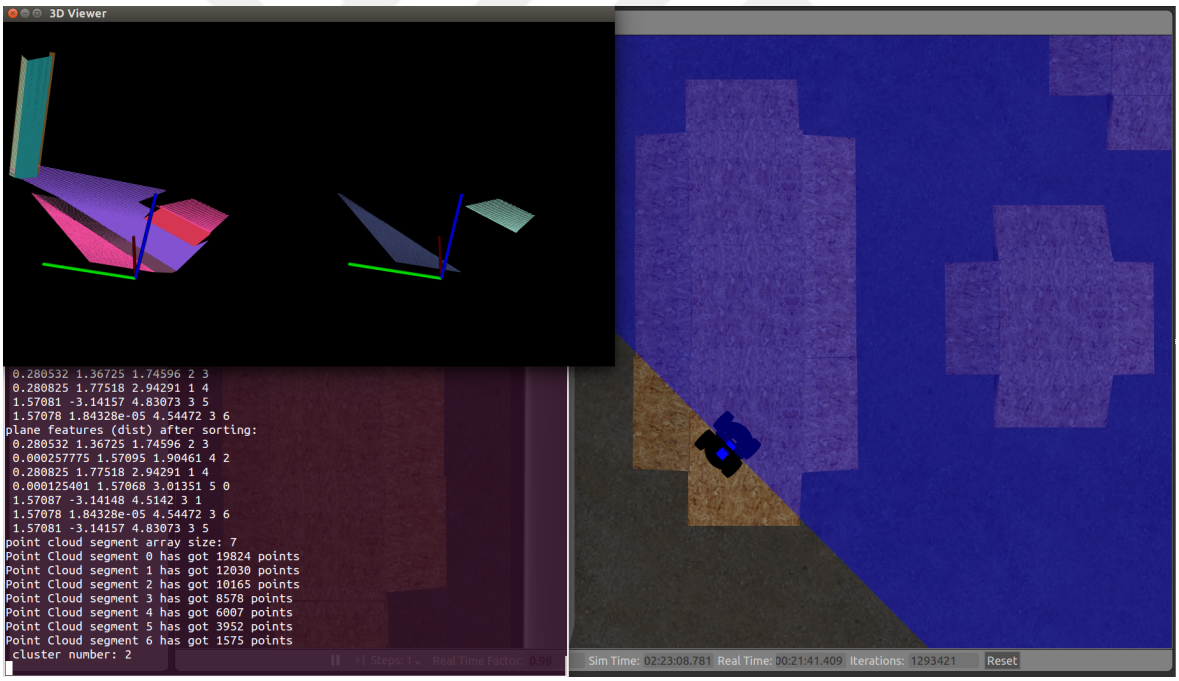
4.4.2 Düz Rampalarda Yolnoktası Oluşturma

Algoritmanın bu bölümü, önceki bölüme benzer şekilde kamera merkezine göre olan uzaklıklarına göre sıralanan rampalar ele alınırken, işlenecek olan rampanın bir düz rampa olarak sınıflanmış olması durumunda çalıştırılmaktadır.

Düz rampalara ait nokta bulutları incelendiğinde, bazı durumlarda bölütleme sonucu farklı düzlemlere atanmasını beklediğimiz nokta gruplarının, tek bir düz rampaya ait olarak atanmış olduğu gözlemlenmiştir. Bu gibi durumlarla görünüş olarak ayrı düzlemler olarak görünmesine rağmen, RANSAC uygulanmasıyla matematiksel olarak aynı düzleme dahil kabul edilen noktalardan ötürü karşılaşıldığı düşünülmüştür. Ancak, sağlıklı bir şekilde

yolnoktası bulunabilmesi için, bu düzlemlerin birbirinden ayrılması gerekmektedir. Bunu gerçekleştirebilmek için de düz rampaya ait nokta bulutuna bir kümeleme algoritması uygulanmaktadır. Bunun için de PCL kütüphanesinde bulunan Öklid Kümeleme(Euclidean Cluster Extraction) algoritması kullanılmıştır (Point Cloud Library (PCL), 2016).

Öklid Kümeleme algoritması ile temel olarak, birbirlerine olan uzaklıkları belirli bir değerin üstünde olan nokta bulutları farklı küme merkezlerine atanmaktadır. Algoritmanın çalışması ile ilgili detaylı bilgiye (Rusu, 2009)'dan erişilebilir. Bir küme oluşturabilecek en küçük ve en büyük nokta sayısı ile farklı kümelerin oluşturulması için gereken uzaklık ölçütleri parametre olarak girilmiştir. Bu çalışma için, uzaklık ölçütü 50cm. olarak alınmış, diğer parametreler için de varsayılan değerleri kullanılmıştır. Şekil 4.37'de bahsedilen durumla karşılaşılan bir düz rampanın kümeleme yapılmamış durumu(3DViewer Penceresi - Sol) ile birlikte kümeleme yapıldıktan sonraki durumu(3DViewer Penceresi - Sağ) verilmiştir.

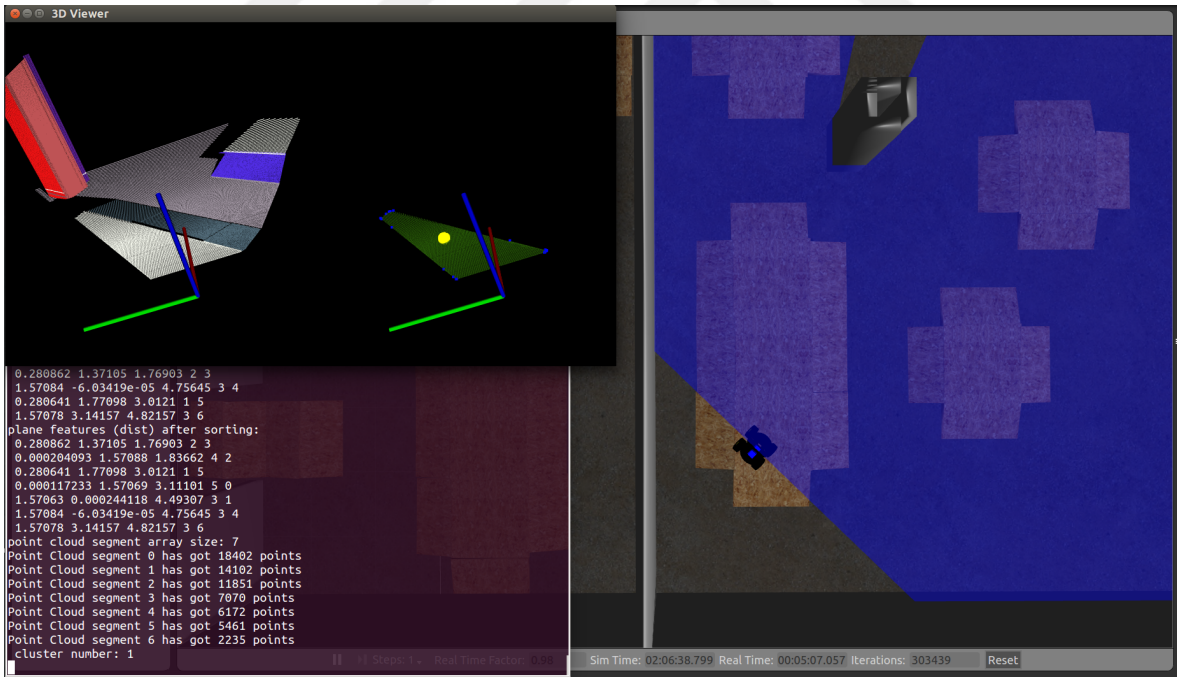


Şekil 4.37: Düz Rampa Düzlemine Kümeleme Uygulanması

Kümeleme işleminden sonra, elde edilen nokta bulutu kümelerinin ağırlık merkezleri hesaplanmakta ve kamera merkezine olan uzaklıkları bulunmaktadır. Bulunan uzaklıklar da $enBuyukMumkunUzaklik$ parametresi ile karşılaştırılarak, uzaklığı bu parametreden küçük olmayan nokta bulutu kümelerinin işlenmemesi sağlanmaktadır.

Uzaklık kontrolünden geçen nokta kümelerine de bölütleme kısmında kullanılan RANSAC algoritması uygulanarak, nokta kümelerinin düzlem modeli ve bu modele uyumlu küme elemanları bulunmaktadır.

Bu aşamada, düz rampa için yolnoktası elde etmek amacıyla, RANSAC modeline uyumlu olan küme elemanlarının oluşturduğu nokta bulutunu çevreleyen kenarların bulunması hedeflenmektedir. Bu doğrultuda, yine PCL kütüphanesi bünyesinde bulunan Dışbükey Zarf (Convex Hull) algoritmasından faydalanılmıştır (Point Cloud Library (PCL), 2016). Burada kullanılan algoritma ile ilgili detaylı bilgi için (Qhull, 2016) incelenebilir. Kısaca bir dışbükey zarf vaya dışbükey gövde, bir X kümesini kapsayan bütün dışbükey kümelerin kesişimi ya da X kümesinin elemanı olan bütün noktaların dışbükey kombinasyonları olarak tanımlanabilir. Elde edilen düz rampayı belirten nokta bulutuna dışbükey zarf uygulandıktan sonra elde edilen noktalar Şekil 4.38'de görülebilmektedir. Bu sayede, bütün nokta bulutu kullanılarak bir yolnoktası oluşturmak yerine, eğimli rampalarda olduğu gibi rampa düzlemini belirten az sayıda noktanın kullanılmasıyla bir yolnoktası elde edilmesi amaçlanmıştır.



Şekil 4.38: Kümelenmiş Düz Rampa Düzlemine Dışbükey Zarf Uygulanması ile Yolnoktası Elde Edilmesi

Düz rampanın kenarlarını belirten nokta kümesi elde edildikten sonra, eğimli rampalarda anlatılan yönteme benzer şekilde üst ve alt limitleri belirten noktalar bulunmaktadır. Ancak bu noktada, iniş-çıkış rampalarındaki işlemlerin aksine, herhangi bir

düzlem kesişimi ile uçnokta hesabı yapılmamakta, direk olarak bulunan limit noktalarının ortalaması alınmaktadır. Böylelikle, düz rampalar için, eğimli rampalar için bulunan iki ($cikisDuzlemiOnuYolnoktasi$ ile üç) yolnoktası yerine sadece bir tane yolnoktası bulunmaktadır. Bunun nedeni olarak da, eğimli rampalarda robotun kenarlara takılma ve düşme riskinin, düz bir rampadakine göre çok daha fazla olması verilmektedir. Son olarak elde edilen yolnoktası, $sonYolnoktasiVektoru$ ile tanımlanan vektöre kaydedilmektedir.

4.4.3 Yolnoktası Karar Verme Algoritması

Bu bölüme gelindiğinde, $enBuyukMumkunUzaklik$ ile belirlenen uzaklık sınırı içerisinde kalan bütün rampa düzlemleri ele alınmış ve bulunan bütün yolnoktaları $sonYolnoktasiVektoru$ olarak adlandırılan bir vektörde toplanmış bulunmaktadır. Bu noktada, elde edilen yolnoktaları arasından bir seçim yapılarak, robotun seyrüseferine başarıyla devam edebilmesi için gideceği bir yerel yolnoktası belirlenmesi amaçlanmaktadır.

İlk olarak, elde edilen bütün yolnoktaları robota(kamera merkezine) olan öklid uzaklıklarına göre yakın olandan uzak olana doğru sıralanırlar. Daha sonra, en yakındaki rampanın küresel hedefe uzaklığının, robotun küresel hedefe olan uzaklığıyla karşılaştırılması yapılır. Eğer en yakın yerel yolnoktası hedefe robottan daha yakında değilse, bu yolnoktasının robotun gideceği yolu uzatacağı düşünülerek bu yoknoktası atlanır ve sıradaki yolnoktalarına aynı işlemler uygulanır.

Bir sonraki aşamada, ilk olarak robotun bir düz rampada mı olduğu yoksa yer düzleminde mi olduğuna karar verilir. Bunun için, robotun z eksenindeki konumuna bakılır. Bu noktada, robotun eğimli bir rampa üzerinde kalmış olacağı durumu hesaba katılmamaktadır. Bunun nedeni, eğer robot eğimli bir rampa üzerindeyse, RGB-D kameranın görüş açısından dolayı rampa algoritmalarının sağlıklı çalışamayabileceğidir.

Bu noktada, robotun yer düzleminde veya yere paralel bir düz rampada olması durumlarına göre işlemler uygulanmaktadır. Robotun yerde olması durumunda, ya bir çıkış rampasına gidebileceği ya da yerde küresel hedefine doğru olan hareketine devam edebileceği varsayımları yapılmıştır. Eğer bir çıkış rampası algılandıysa ve rampa üzerindeki yolnoktası da küresel hedefe daha yakın konumdaysa, bu çıkış rampası üzerindeki yolnoktasına robotun gidip gidemeyeceğinin kontrolü yapılır. Bu doğrultuda, daha önceden bahsedilen $rampaGecisBayrak$ isimli mantıksal parametre kontrol edilir. Bu parametre, eğer robotun yönelim açısı ile rampanın yönelim açısı arasındaki fark $aciFarkiParam$'dan büyükse, yanlış olarak belirlenmekteydi ve o rampanın geçişe müsait olmadığını belirtmekteydi. Kontrol edilen yolnoktasına, geçilebilir durumda da olsa

geçilemez durumda da olsa, $rampaOnuYolnoktasiKontrolu$ isimli bir algoritma ile ikinci bir kontrol uygulanmaktadır.

$rampaOnuYolnoktasiKontrolu$ ile belirtilen algoritmanın amacı, eğer yolnoktasının ait olduğu rampa geçilebilir durumdaysa, bu yolnoktasına giderken başka rampalara çarpmanın önüne geçilmesidir. Eğer yolnoktası gidilemez durumdaysa da, $cikisDuzlemiOnuYolnoktasi$ parametresiyle tanımlanmış çıkış rampasının önündeki yolnoktasını ifade eden parametreye başka bir rampaya çarpmadan gidilip gidilemeyeceğine bakılmaktadır. Bunun nedeni de, önceden belirtildiği gibi, rampa gidilebilir bir açıda olmamasına rağmen, $cikisDuzlemiOnuYolnoktasi$ ile belirtilen noktaya gidilmesiyle, robotun bir sonraki adımda açısını ayarlayarak bu sefer rampadan geçebilecek bir konuma gelmesinin sağlanmasıdır.

$rampaOnuYolnoktasiKontrolu$ ile belirtilen eğimli rampa çarpışma kontrolü algoritması, girdi olarak rampanın kenarlarını belirten ve aykırı nokta düzeltme algoritmasıyla hizalanmış uçnoktalarını, kontrol edilmekte olan yerel yolnoktasını ve robotun küresel hedef noktasını almaktadır. Çıktı olarak ise mantıksal doğru/yanlış döndürmektedir. Burada, girdi olarak kabul edilen rampaların kenarlarını belirten uçnoktaları içerisinde, kontrol edilmekte olan yolnoktasının ait olduğu rampa dışındaki diğer bütün eğimli rampaların uçnoktalarının olduğunu belirtmekte fayda vardır. Bunun sebebi de, bu algoritma ile kontrol edilen yolnoktasının ait olduğu rampaya giderken diğer eğimli rampalarla çarpışmasının kontrol edilecek olmasıdır.

Eğimli rampa çarpışma kontrolü yapılırken, ilk önce çarpışmanın kontrol edildiği rampanın kenarlarını belirten uçnoktalara bir doğru parçası oturtulur. Daha sonra kenarları belirtmekte olan bu doğru parçaları ile robot ve küresel hedef arasında oluşturulan bir doğru parçasının kesişimine bakılır. Bu işlem yapılırken, doğru parçalarının z ekseni değerleri eşitlenerek iki boyutlu olarak doğru parçalarının kesişiminin kontrolü yapılmış olunur. Bu bahsedilen $rampaOnuYolnoktasiKontrolu$ ile belirtilmiş eğimli rampa çarpışma kontrolü algoritması, Algoritma 4'da ifade edilmiştir.

Yerel yolnoktasına karar verilirken, $rampaGecisBayrak$ mantıksal değişkeni ve $rampaOnuYolnoktasiKontrolu$ algoritması ile ifade edilen kontrollerden yararlanılmaktadır. Karar verilirken kullanılan algoritmanın genel yapısı Algoritma 5'de verilmiştir.

Genel algoritmaya göre, robot yerde ve en yakın yerel yolnoktası bir çıkış rampasına ait ise, ilk başta $rampaGecisBayrak$ parametresi kontrol edilmektedir. Bu kontrolden geçen yolnoktası, daha sonra $rampaOnuYolnoktasiKontrolu$ algoritması ile belirtilen

Algoritma 4: Eğimli rampa çarpışma kontrolü algoritması ($rampaOnuYolnoktasKontrolu$)

Girdi: İniş-çıkış rampalarının sağ ve sol kenarlarını belirten doğru parçalarını tutan vektör $rampaKenar$, rampa üzerinde bulunan yerel hedef noktası $yerelHedef$, robotun gitmek istediği küresel hedef noktası $kureselHedef$

Çıktı : Mantıksal doğru-yanlış

Veri: Rampaların kenarlarını tutan vektörün boyutu $rampaKenar\_boyut$, bir ucu robotta diğer ucu küresel hedefte olan doğru parçası $dogruParcasi$, ait olunan rampayı belirten kimlik $rampaKimlik$

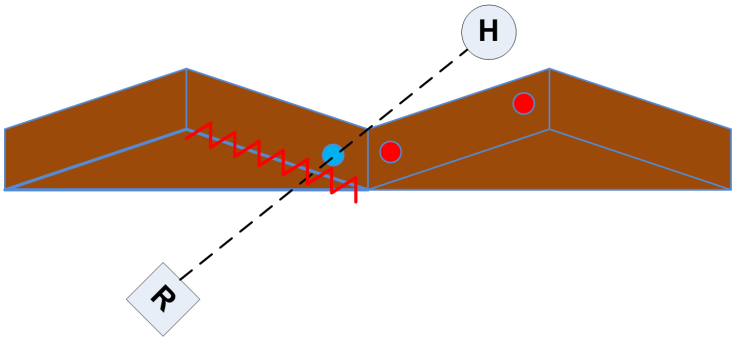
```
// Robot ile küresel hedef arasında, yerel hedef noktasının ait olduğu
   rampa dışında başka bir rampa ile karşılaşılma durumunun kontrolü
for i ← 0 to rampaKenar_boyut do
    // Robottan küresel hedefe bir doğru parçası oluştur
    dogruParcasi ← dogru_olustur(robot, kureselHedef)
    // Robottan küresel hedefe çizilen doğru parçası ile yerel hedefin ait
       olduğu rampa dışındaki rampaların kenarlarının kesişimini kontrol et
    if rampaKimlik(rampaKenar(i))! = rampaKimlik(yerelHedef) then
        if kesisme_var(dogruParcasi, rampaKenar(i)) == dogru then
            return yanls
        else
            return dogru
        end
    end
end
```

kontrole sokulmaktadır. Yolnoktası her iki kontrolden de geçerse, $cikisDuzlemiOnuYolnoktasi$ ile belirtilen çıkış rampasının önündeki noktayı ifade eden yolnoktası gidilecek yerel hedef olarak seçilmektedir. Eğer $rampaOnuYolnoktasiKontrolu$ ile ifade edilen ikinci kontrolden geçememiş ise, $cikisDuzlemiOnuYolnoktasi$ yerine rampanın kendisi üzerindeki yolnoktası gidilecek yerel hedef olarak seçilmektedir. Buradaki düşünce, rampa üzerindeki yolnoktası açısal olarak gidişe uygun çıkmasına rağmen, $rampaOnuYolnoktasiKontrolu$ sonucu itibariyle rampanın yakınlarında çarpışmaya neden olabilecek bir başka rampanın olması muhtemel olduğundan, eğer $cikisDuzlemiOnuYolnoktasi$ seçilirse, bu yolnoktasının çarpışma riski olan rampa üzerinde kalabilecek olması olayıdır (Şekil 4.39). Toplayacak olursak, eğer çıkış rampası üstündeki yolnoktası $rampaGecisBayrak$ ile açısal olarak gidişe uygun durumdaysa ve $rampaOnuYolnoktasiKontrolu$'nün sonucu da doğru dönüyorsa (başka rampa ile çarpışma yoksa), $cikisDuzlemiOnuYolnoktasi$ gidilecek yerel yolnoktası olarak seçilmektedir. Ancak $rampaGecisBayrak$ doğru olmasına rağmen $rampaOnuYolnoktasiKontrolu$ yanlış oluyorsa (başka rampa ile çarpışma riski varsa), yerel hedef olarak rampa üzerindeki yolnoktası seçilmektedir.

Algorithm 5: Yerel Yolnoktası Karar Verme Algoritması

Girdi: Sıralanmış yol noktası vektörü YNV
Çıktı : En uygun yol noktası $EUYN$
Data: YNV boyutu $boyut\_YNV$, robotun pozisyonu P_{robot}, küresel hedef P_{hedef}, düz rampa yükseltisi $rampa_z$, rampa uygunluk eşik değeri $rampa_{th}$

```
for i ← 0 to boyut_YNV do
    mesafeY ← sqrt((YNV_i^x − P_hedef^x)^2 + (YNV_i^y − P_hedef^y)^2)
    mesafeR ← sqrt((P_robot^x − P_hedef^x)^2 + (P_robot^y − P_hedef^y)^2)
    if mesafeY < mesafeR then // yol noktası hedefe robottan daha yakınsa
        if P_robot^z < rampa_z then // Robot yerde ise
            if YNV_i^durum ==çıkış then // Çıkış rampası ise
                if |P_robot^th − YNV_i^th| > rampa_th then
                    if rampaOnuYolnoktasiKontrolu==true then
                        return cikisDuzlemiOnuYolnoktasi
                    else
                        continue
                    end
                else
                    if rampaOnuYolnoktasiKontrolu==true then
                        return cikisDuzlemiOnuYolnoktasi
                    else
                        return YNV_i
                    end
                end
            end
        end
        if P_robot^z > rampa_z then // Robot yerden yüksek düz rampada ise
            if YNV_i^durum ==çıkış then // Çıkış rampası ise
                if |P_robot^th − YNV_i^th| > rampa_th then
                    if rampaOnuYolnoktasiKontrolu==true then
                        return cikisDuzlemiOnuYolnoktasi
                    else
                        continue
                    end
                else
                    if rampaOnuYolnoktasiKontrolu==true then
                        return cikisDuzlemiOnuYolnoktasi
                    else
                        return YNV_i
                    end
                end
            end
            if YNV_i^durum ==iniş then
                if |P_robot^th − YNV_i^th| > rampa_th then
                    if (sqrt((P_robot^x − P_rampa^x)^2 + (P_robot^y − P_rampa^y)^2)) < 1 then
                        continue
                    else
                        return YNV_i
                    end
                else
                    return YNV_i
                end
            end
            if YNV_i^durum ==düz then
                return YNV_i
            end
        end
    end
end
```



Şekil 4.39: $rampaOnuYolnoktasiKontrolu$ Algoritmasıyla Rampa Çakışma Kontrolü

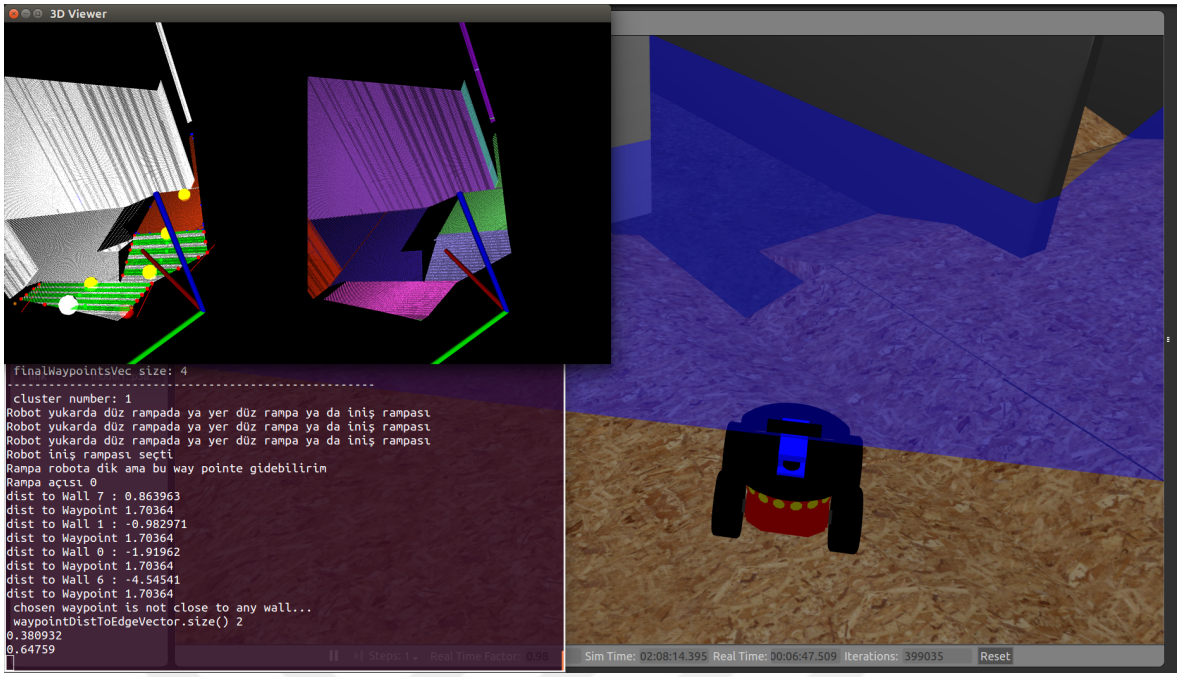
Öte yandan, eğer yolnoktası $rampaGecisBayrak$ ile yapılan ilk kontrolden geçemez ise, $sonYolnoktasiVektru$ vektöründeki sıradaki yolnoktasına geçilmeden önce, $rampaOnuYolnoktasiKontrolu$ algoritması kontrolünden geçirilmektedir. Eğer bu kontrol sonucu doğru dönerse, yani yol üzerinde başka rampa ile çarpışma riskinin olmadığı anlaşılırsa, rampa önündeki $cikisDuzlemiOnuYolnoktasi$ gidilecek yerel hedef olarak seçilir. Böylelikle, rampa açısal kontrole göre direk olarak geçilemeyecek durumda olmasına rağmen, başka bir eğimli rampayla çarpışma riski olmadığından $cikisDuzlemiOnuYolnoktasi$ ile belirtilen noktaya gidilerek, robotun daha sonra yönünü ayarlaması ile birlikte rampadan geçebilmesi için ortam sağlanmış olmaktadır.

Bu noktaya kadar robotun yerde olduğu durumlardaki davranışından bahsedilmiştir. Eğer robot yerde değil de bir düz rampada ise, en yakın yolnoktasının iniş rampasına, çıkış rampasına veya düz rampaya ait olmasına göre kontrollerden geçirilmektedir.

Eğer robot düz rampadayken en yakın yolnoktası bir düz rampaya ait ise, yerel hedef olarak bu düz rampaya ait yolnoktası seçilerek robot hareketine devam eder.

Eğer en yakın yolnoktası bir iniş rampası ise, $rampaGecisBayrak$ parametresi kontrol edilir. Eğer parametrenin değeri doğru ise, rampa geçişe açısal olarak uygundur denir ve bu rampaya ait yolnoktası yerel hedef olarak seçilir. Eğer $rampaGecisBayrak$ parametresinin değeri yanlış ise, rampanın robota olan uzaklığına bakılır. Eğer bu uzaklık belirli bir değerden(1m. gibi) fazla ise, bu yolnoktasına açısal olarak uygun olmamasına rağmen gidilebilir denilmektedir. Bunun nedeni, bu gibi durumlarla Şekil 4.40'de ifade edildiği gibi durumlarda karşılaşılıyor olunmasıdır.

Şekilde görüldüğü üzere, bu gibi durumlarla karşılaşıldığında, rampa açısı robotun yönelimine göre dik bir konumda olmaktadır. Ancak rampaların yerleşimi göz önüne alındığında, robotun bu durumlarda açısal kontrolden geçememiş olmasına rağmen yerel



Şekil 4.40: İniş rampasının dik gelmesine rağmen gidilebileceği durum

yolnoktasına herhangi bir çarpışma veya devrilme durumu olmaksızın ilerleyebildiği gözlemlenmiştir.

Son olarak, eğer en yakın yolnoktasının ait olduğu rampa bir çıkış rampası ise, daha önceden robotun yerde bulunduğu durumlarda anlatıldığı gibi iki adımlı bir kontrol işlemi uygulanmaktadır. İlk adımda, $rampaGecisBayrak$ parametresinin değeri kontrol edilir. Doğru gelirse, ikinci adımda $rampaOnuYolnoktasiKontrolu$ algoritmasının döndürdüğü değer kontrol edilir. Doğru gelmesi durumunda, $cikisDuzlemiOnuYolnoktasi$ yerel hedef olarak seçilirken, yanlış gelmesi durumunda çıkış rampası üzerindeki yolnoktası yerel hedef olarak belirlenir. Eğer $rampaGecisBayrak$ parametresinin durumu yanlış ise, yine $rampaOnuYolnoktasiKontrolu$ algoritmasının çıktısına bakılır. Algoritmanın sonucu mantıksal yanlış ise, bu rampaya ait olan yolnoktası kullanılamaz denilir ve $sonYolnoktasiVektru$ vektöründeki sıradaki yolnoktasının kontrol işlemleri yapılır. Mantıksal doğru dönmesi durumunda ise $cikisDuzlemiOnuYolnoktasi$ yerel hedef noktası olarak seçilir.

Yerel yolnoktası karar verme algoritması ile bir yolnoktası yerel hedef olarak seçilmiş ise, son olarak bu yolnoktasından çevredeki duvar düzlemi olarak bölütlenmiş nokta bulutlarına olan uzaklıkları kontrol edilir. Eğer seçilen yolnoktası duvarlardan birine belirlenen bir eşik değerine göre fazla yakın çıkmışsa, seçilen yolnoktası, duvardan yolnoktasına çizilen dik uzaklık vektörü doğrultusunda (duvarın tam zıttı doğrultuda) yine

belirlenen bir miktar kadar kaydırılır. Bunun nedeni, yolnoktasının duvara yakın olması durumunda, robot bir yandan bu yolnoktasına hareket etmeye çalışırken, bir yandan da VAH algoritmasının duvarı çok yakın görmesi nedeniyle robotu duvardan uzaklaştırmaya çalışması ile oluşan sıkışma durumunu önlemektir. Bu aşamadan sonra seçilen yolnoktası robotun küresel yönelim açısına göre yerel koordinatlardan küresel koordinatlara çevrilmektedir.

Ara noktaların nasıl belirlendiğini anlattıktan sonra nasıl kullanılacağı konusuna değinmekte fayda görülmektedir. Bu noktada, robot küresel hedefi belli olduğunda yönünü bu hedefe döndürmektedir. Daha sonra yuvarlanma ve yunuslama açılarını kontrol etmektedir. Bu açıların kontrol edilmesinin sebebi, rampa üzerinde yolnoktası belirleme işleminin robotun yer düzleminde veya düz rampada olduğu durumlarda yapılmasının istenmesidir. Bu açılar belirlenen bir eşik değerinin altındaysa ara nokta belirleme aşamasına geçmektedir. Aksi takdirde, robot küresel hedefine gitmeye devam etmektedir. Ara nokta belirleme algoritmasından 3 tane sonuç elde edilebilmektedir. Bunlardan ilki, robotun görüş alanında rampa olmadığı durumdur. Bu durumda robot küresel hedefine hareketine devam etmektedir. İkinci durumla uygun yolnoktası bulunamadığında karşılaşılmaktadır. Bu durumda da robot küresel hedefine gitmektedir. Son olarak, uygun bir yolnoktasının belirlenmesi durumunda robot yolnoktasına doğru harekete başlamaktadır. Bu esnada robotun konumu ile hem küresel hedef hem de yolnoktası arasındaki uzaklık kontrol edilmektedir. Eğer önce küresel hedefi ile olan uzaklığı belirlenen eşik değerin altına düşerse, robot süreci tekrar başlatarak bir sonraki küresel hedef için yönünü ayarlamaktadır. Diğer yandan, eğer önce rampa üzeri yolnoktasına geldiyse yuvarlanma ve yunuslama açılarını kontrol ederek sürece devam etmektedir. Bu noktada en yakındaki rampanın küresel hedefe uzaklığının, robotun küresel hedefe olan uzaklığıyla karşılaştırması yapılmaktadır. Eğer bu rampa robotun küresel hedefine robottan daha yakın değilse bu rampanın robotu küresel hedefinden uzaklaştıracağı düşünülerek başka bir rampa seçilmektedir. Böylece robotun sürekli olarak yerel noktalar seçerek hedefinden sapması önlenmektedir.

5. SONUÇ VE ÖNERİLER

Önerilen yöntemler, Gazebo benzetim ortamında modellenen Eskişehir Osmangazi Üniversitesi Elektrik - Elektronik Laboratuvar binasında test edilmiştir. Ortamın boyutları 26x15 metre olup 52x30'luk bir metrik ızgara ile ifade edilmiştir. Izgaranın oluşturulmasında openCV kütüphanesinden faydalanılmıştır (openCV, 2016). Benzetimlerde lazer mesafe algılayıcısı bulunan Pioneer P3-AT robotu kullanılmıştır. Izgara hücrelerinin durumunu güncellemek için 5'er derece aralıklı 37 tane lazer ışını kullanılmıştır. Robotun kontrolü ve kullanılan yöntemlerin gerçeklenmesi için Robot İşletim Sistemi (ROS) kullanılmıştır. Robotun benzetim ortamında hareketinin sağlanabilmesi için p2os ros paketi kullanılmıştır (p2os, 2018). Bu paket yardımıyla VAH'da üretilen açısal ve doğrusal hızlar robota uygulanabilmektedir. Bunlara ek olarak, RGB-D algılayıcı verisinin kullanılabilmesi amacıyla OpenNI(Mihelich, 2018) ve PCL(Open Perception, 2018) paketlerinden de faydalanılmıştır.

Şekil 5.1: Test Ortamı

Şekil 5.1'de tek robot kullanılarak gerçekleştirilen testler için kullanılan örnek afet ortamı görülmektedir. Ortam oluşturulurken hector-nist-arenas-gazebo paketi (Simon ve Kohlbrecher, 2018) içerisindeki rampa örneklerinden faydalanılmıştır. Bu noktada,

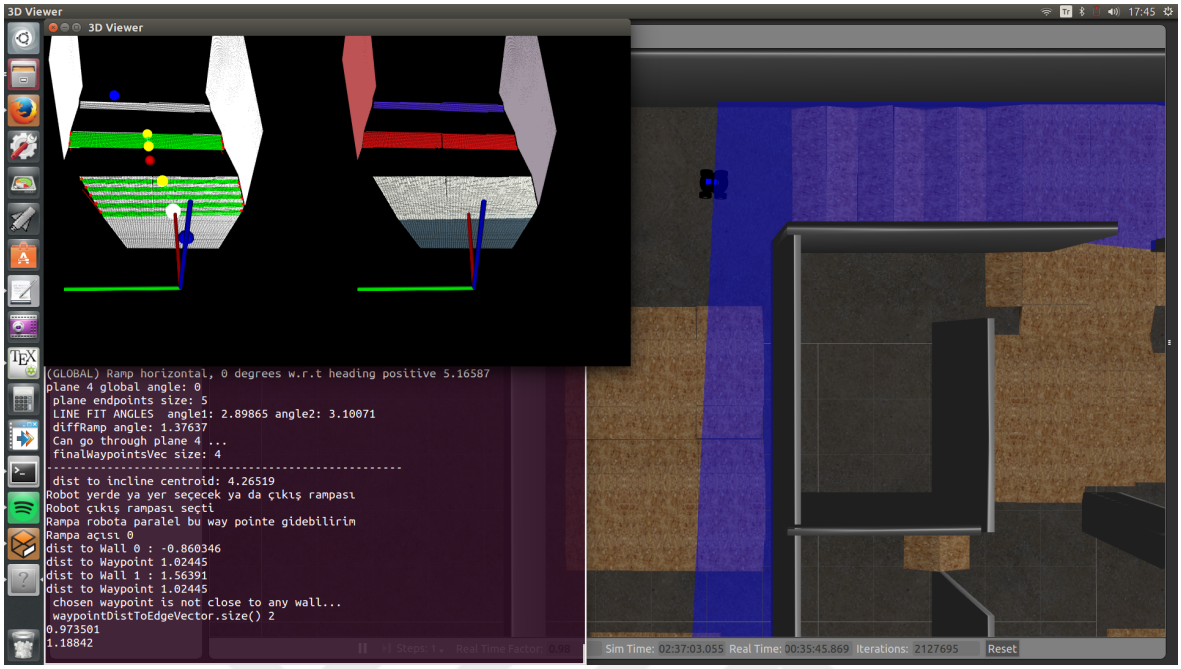
kullanılan bazı parametrelerden bahsetmek faydalı olacaktır. Azami hızlar, doğrusal hız için $120mm/sn$, açısal hız için $0.40rad/sn$ olarak belirlenmiştir. Vektör Alan Histogramı'nda kullanılan iki boyutlu histogram $3x1$ metre seçilmiş ve 0.1 metre çözünürlükle kullanılmıştır. Hedef noktaların ağırlığı 0.6, engellerin ağırlığı ise 0.4 olarak belirlenmiştir. Polar histogramın açısal çözünürlüğü 10 derece seçilmiştir. Robotun açısal ve doğrusal hızları her 1 saniyede güncellenen kartezyen ve polar koordinatları yardımıyla hesaplanmaktadır.

Topolojik harita kullanılarak Dijkstra algoritmasının gerçeklenmesi sırasında, daha önce bahsedilen $BeKS$ parametresi 25 boş hücre olarak seçilmiştir. Eğer topolojik haritanın oluşturulması sırasında kullanılan ikinci ve üçüncü en küçük özdeğerlere karşılık gelen özvektörlerle elde edilen düğümler, engeller sebebiyle bir ağaç oluşturamazsa, $BeKS$ parametresi küçültülmekte ve/veya özvektör sayısı artırılmaktadır. Bu işlem uyarlamalı olarak tüm düğümler bağlanana kadar devam etmektedir.

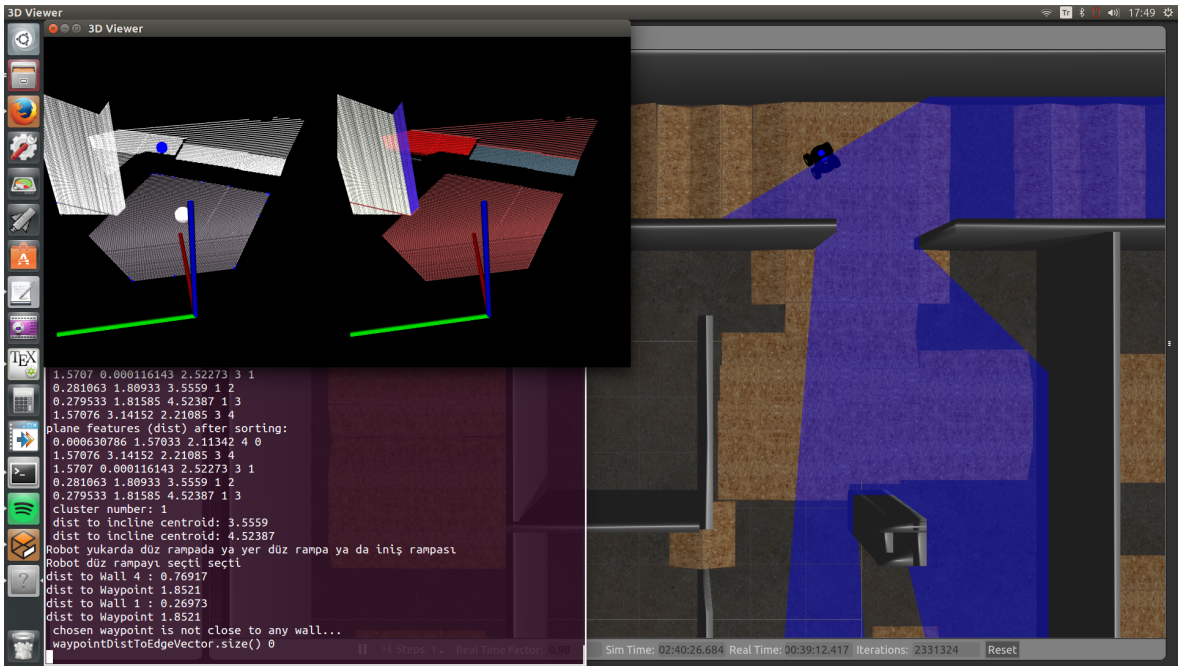
Bu parametrelere ek olarak, rampa algoritması kapsamında kullanılan parametrelerden, $enBuyukMumkunUzaklik$ parametresi 3.5 metre olarak seçilmiştir. Eğimli rampalarda limit değerlerini belirleme kullanılan $egimliRampaBolme$ parametresi 4 olarak belirlenmiştir. Robot ile rampa arasındaki küresel açı farkı $aciFarkiParam$ 45 derece olarak seçilmiştir. Robotun yolnoktası bulma algoritmasını çalıştırmaya karar verilirken kullandıkları yuvarlanma ve yunuslama açılarının 1 dereceden küçük olması gerekmektedir. Son olarak, robotun hedef konuma vardığı kontrol edilirken de robot küresel hedefine 0.5 metre veya yerel hedefine 0.3 metre mesafeden daha yakınsa bu hedefe varıldığına karar verilmektedir.

Bu noktada robotun rampaların bulunduğu afet ortamında önceki bölümde detaylı biçimde açıklanan rampa üzeri yolnoktası bulma algoritmasının çalışmasını gösteren birkaç örnek verilebilir.

Şekil 5.2 incelendiğinde, koyu maviyle gösterilen noktalardan robota yakın olanı ile kırmızıyla gösterilen yolnoktaları, önceki bölümde açıklanmış olan $cikisDuzlemiOnuYolnoktasi$ parametresiyle belirtilen noktaya karşılık gelmektedir. Koyu mavi olan ulaşılabilir, kırmızı olan ulaşılamaz olanını ifade etmektedir. Sarı noktalar iniş ve çıkış rampaları için belirlenen yolnoktalarını, beyaz ise yolnoktaları arasından gitmek üzere yerel hedef olarak seçilenini göstermektedir. Uzakta bulunan açık mavi renkli nokta ise robotun o andaki küresel hedefini ifade etmektedir.

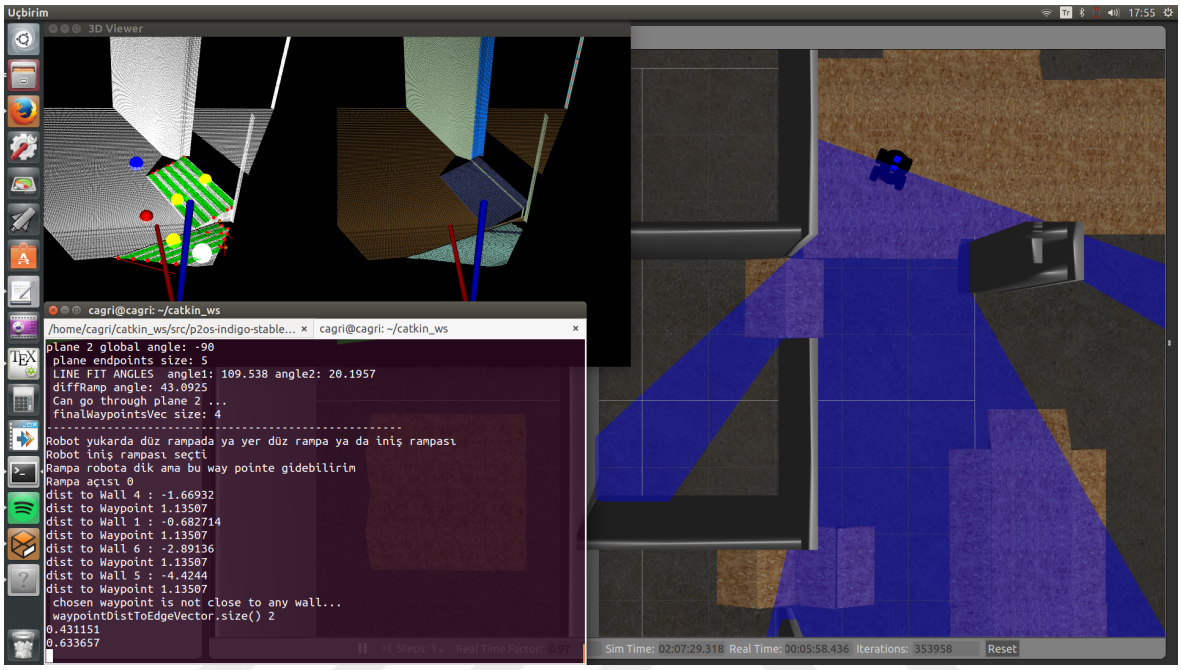


Şekil 5.2: Çıkış Rampası Örneği



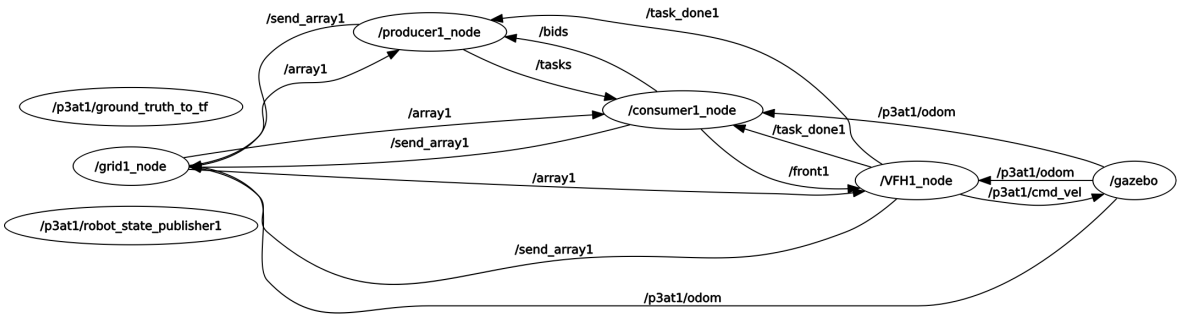
Şekil 5.3: Düz Rampa Örneği

Şekil 5.3'de robotun düz rampa üzerinde bulunduğu bir durum gösterilmiştir. Koyu mavi ile gösterilen nokta küresel hedef iken, beyaz ile gösterilen nokta düz rampa üzerinde seçilen yolnoktasını ifade etmektedir.



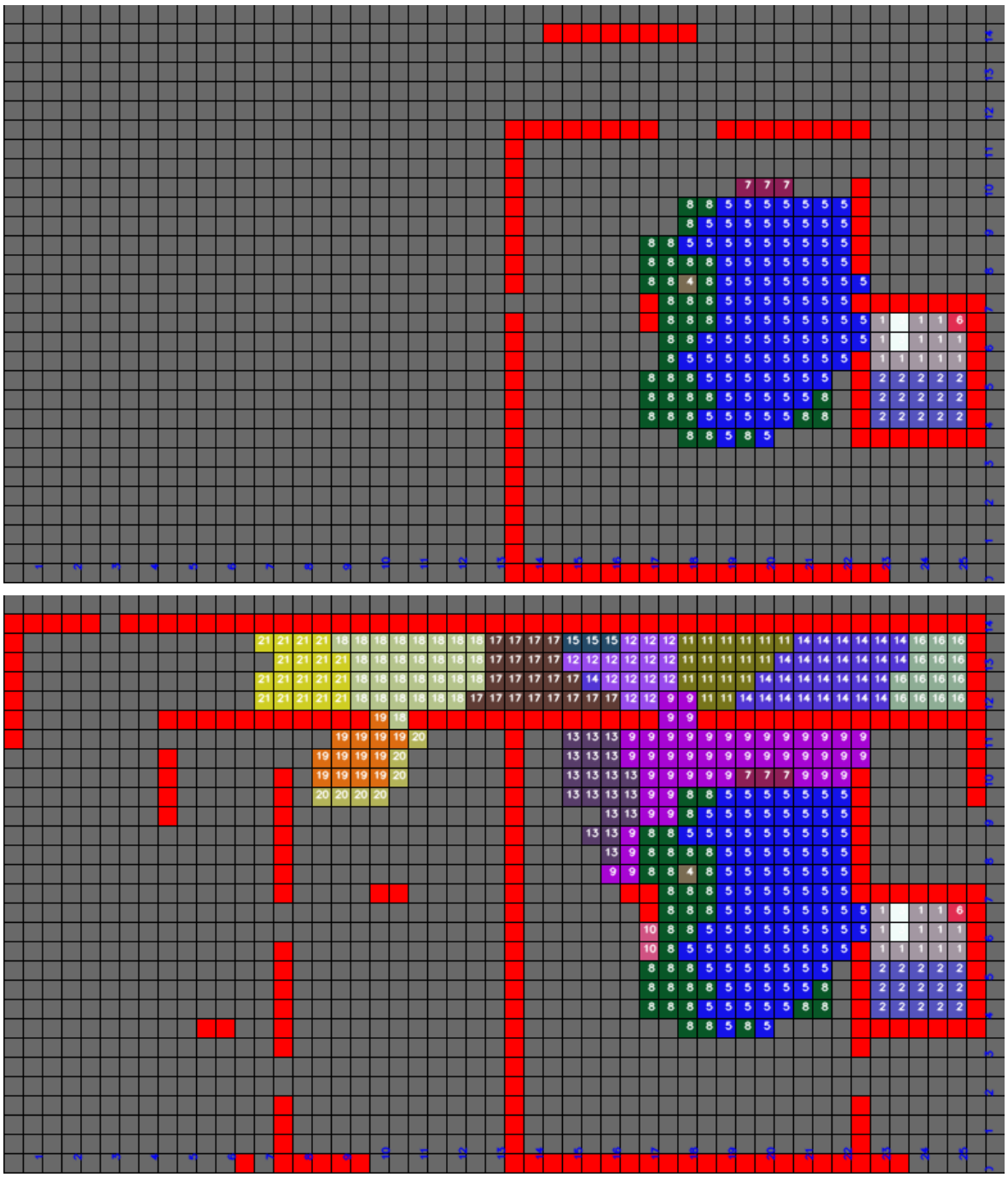
Şekil 5.4: İniş Rampası Örneği

Şekil 5.4'de robotun seçtiği yolnoktası beyazla gösterilmiş ve bir iniş rampası üzerinde olmaktadır. Kırmızı ile gösterilen nokta $cikisDuzlemiOnuYolnoktasi$ parametresi ile belirtilen yolnoktasına karşılık gelmektedir ve ulaşılamaz olarak belirlenmiştir. Sarı noktalar iniş ve çıkış rampaları üzerindeki yolnoktalarını, koyu mavi ise küresel hedef noktasını göstermektedir.

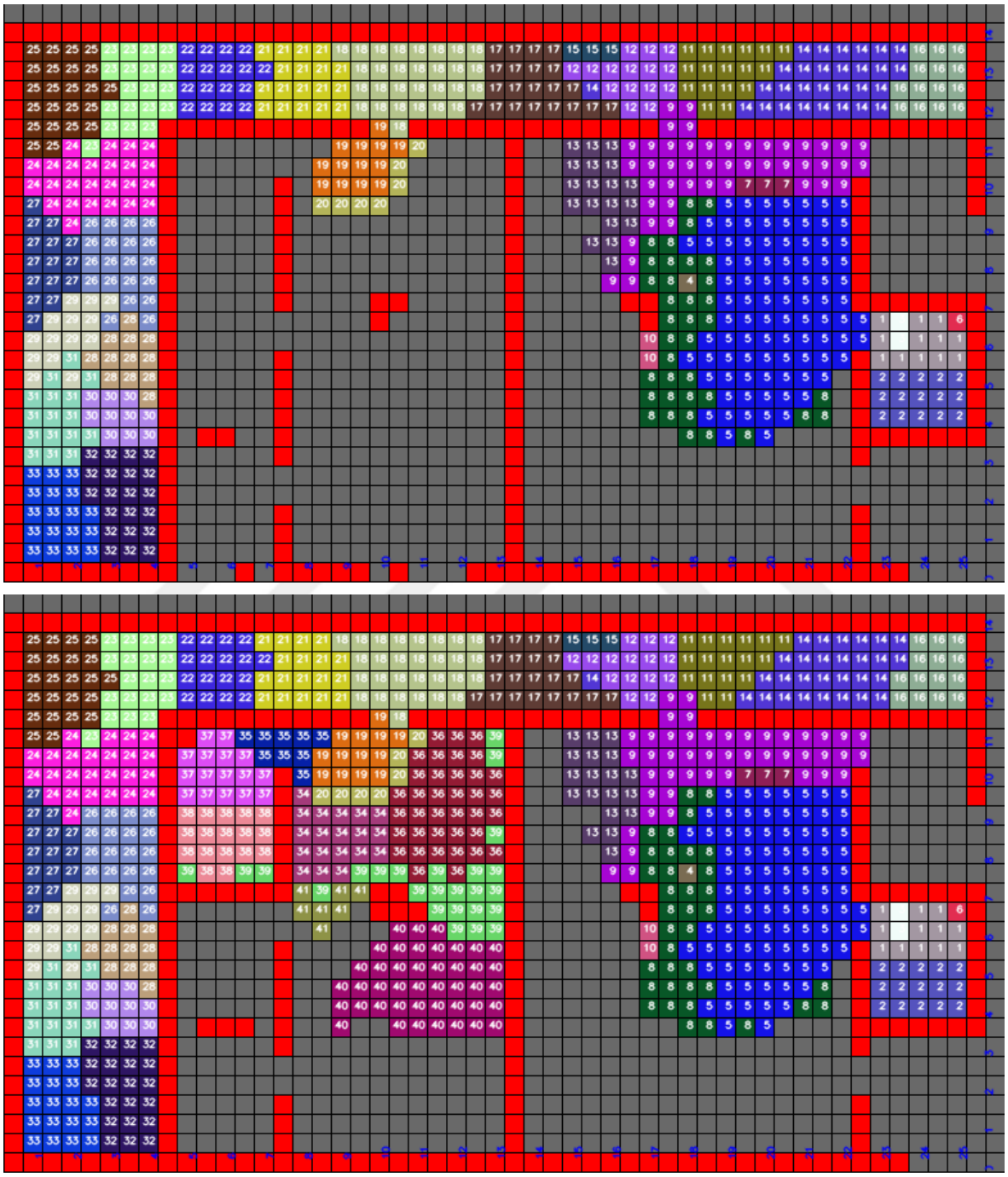


Şekil 5.5: Tek robot ROS Düğümleri

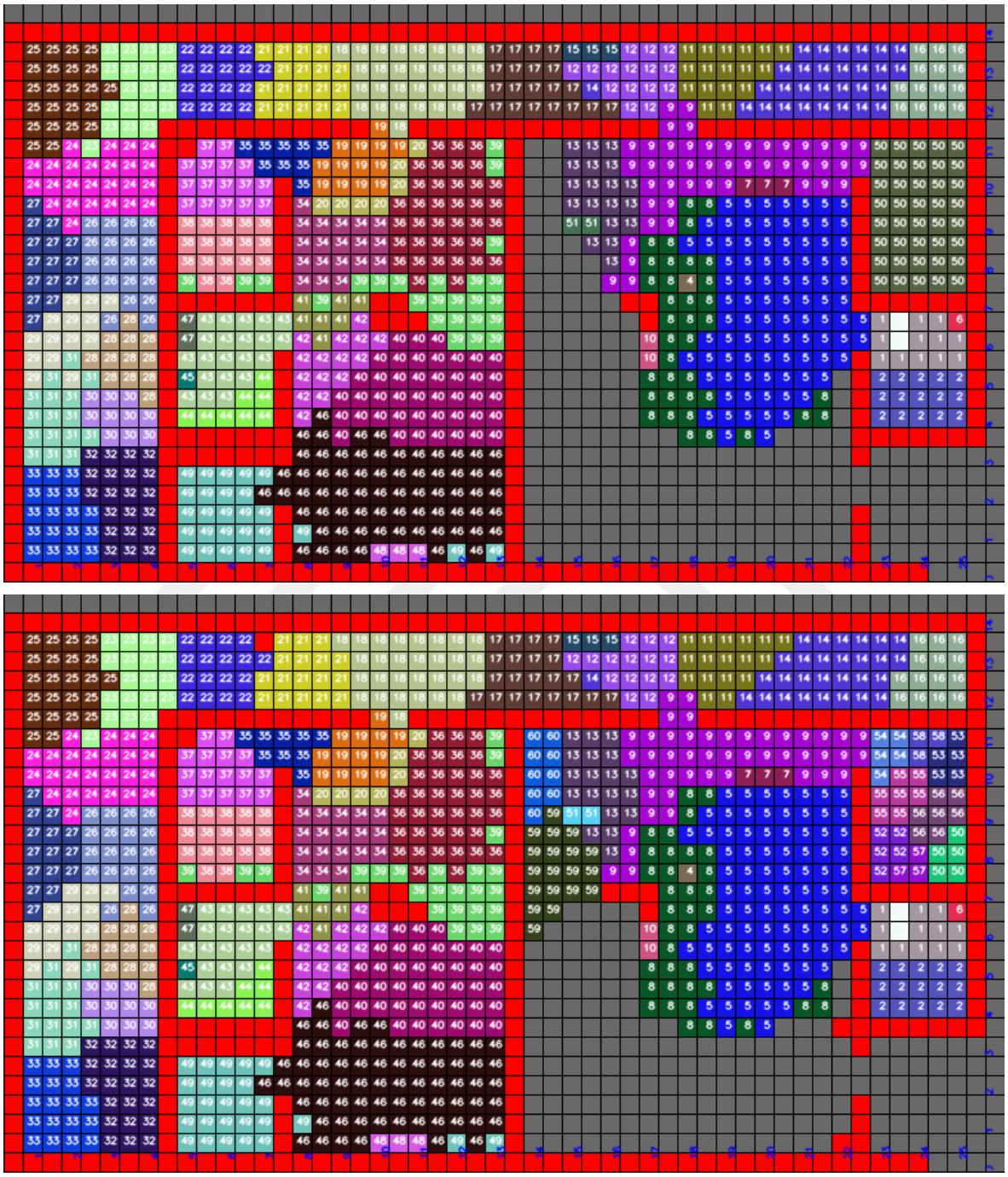
Şekil 5.5'de ROS tabanında tek bir robot üzerinde çalışan düğümler ve bu düğümler arasındaki iletişim kanalları gösterilmektedir. Genel işleyiş bakımından bu düğümlerden "grid1_node" metrik haritanın oluşturulmasından, "consumer1_node" ve "producer1_node" daha önceden harici olarak da belirtilen keşif algoritmaları ve hedef atanmasından, "vfh1_node" ise Vektör Alan Histogramı, çarpışmadan sakınma, rampa algılama gibi seyrüsefer fonksiyonlarından sorumludur.



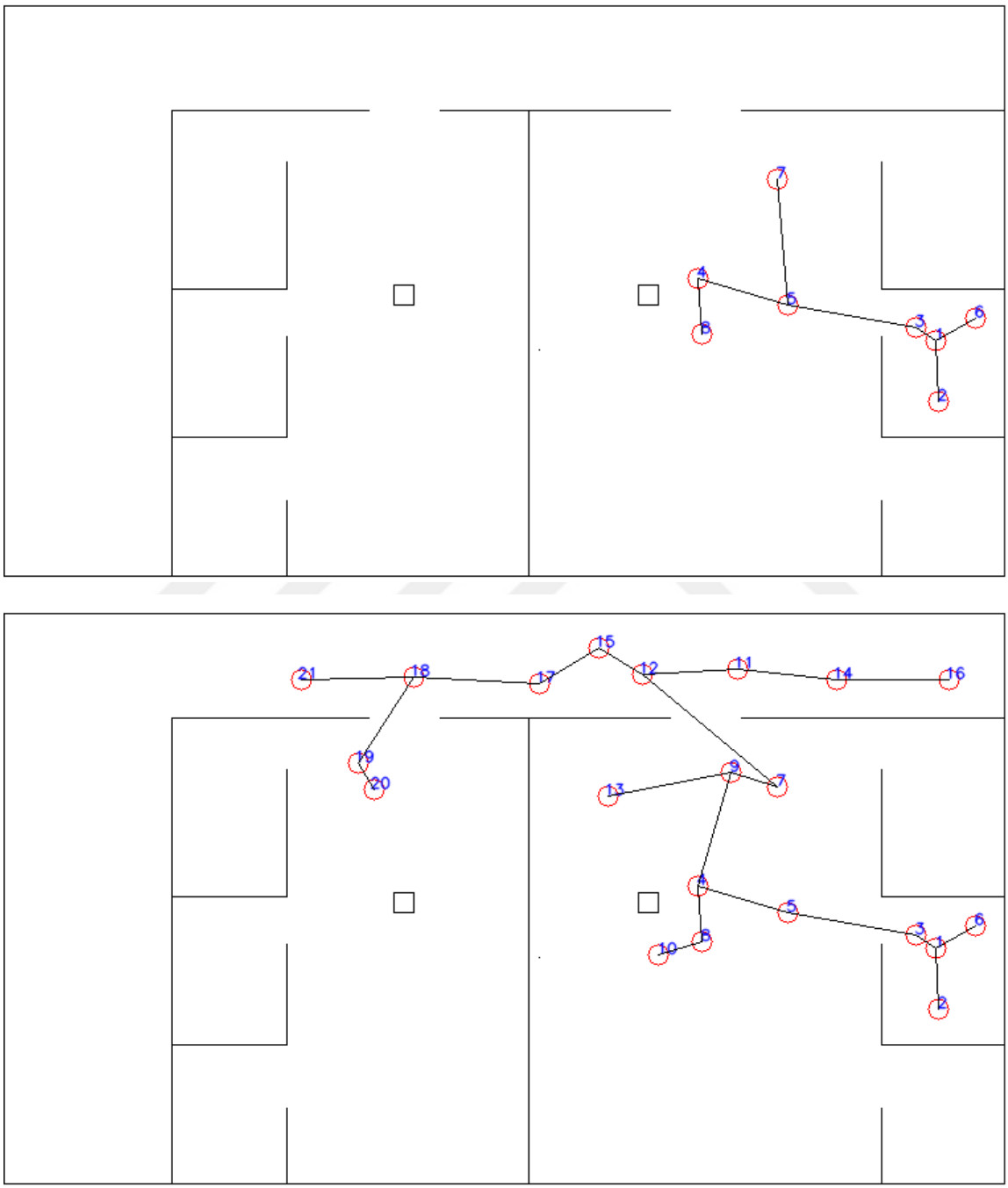
Şekil 5.6: Tek Robot Topolojik Harita - Kümeleme Adımları 1



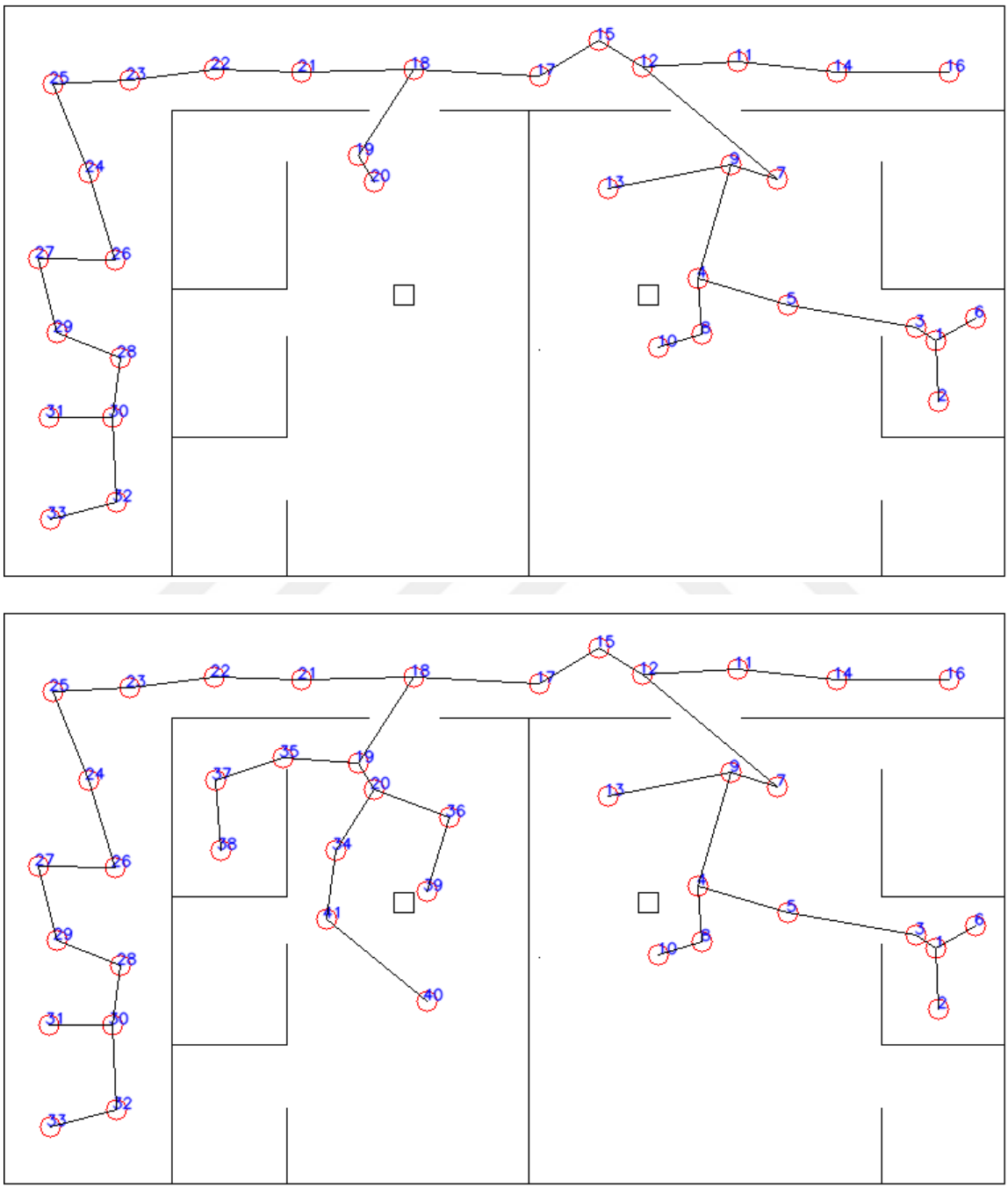
Şekil 5.7: Tek Robot Topolojik Harita - Kümeleme Adımları 2



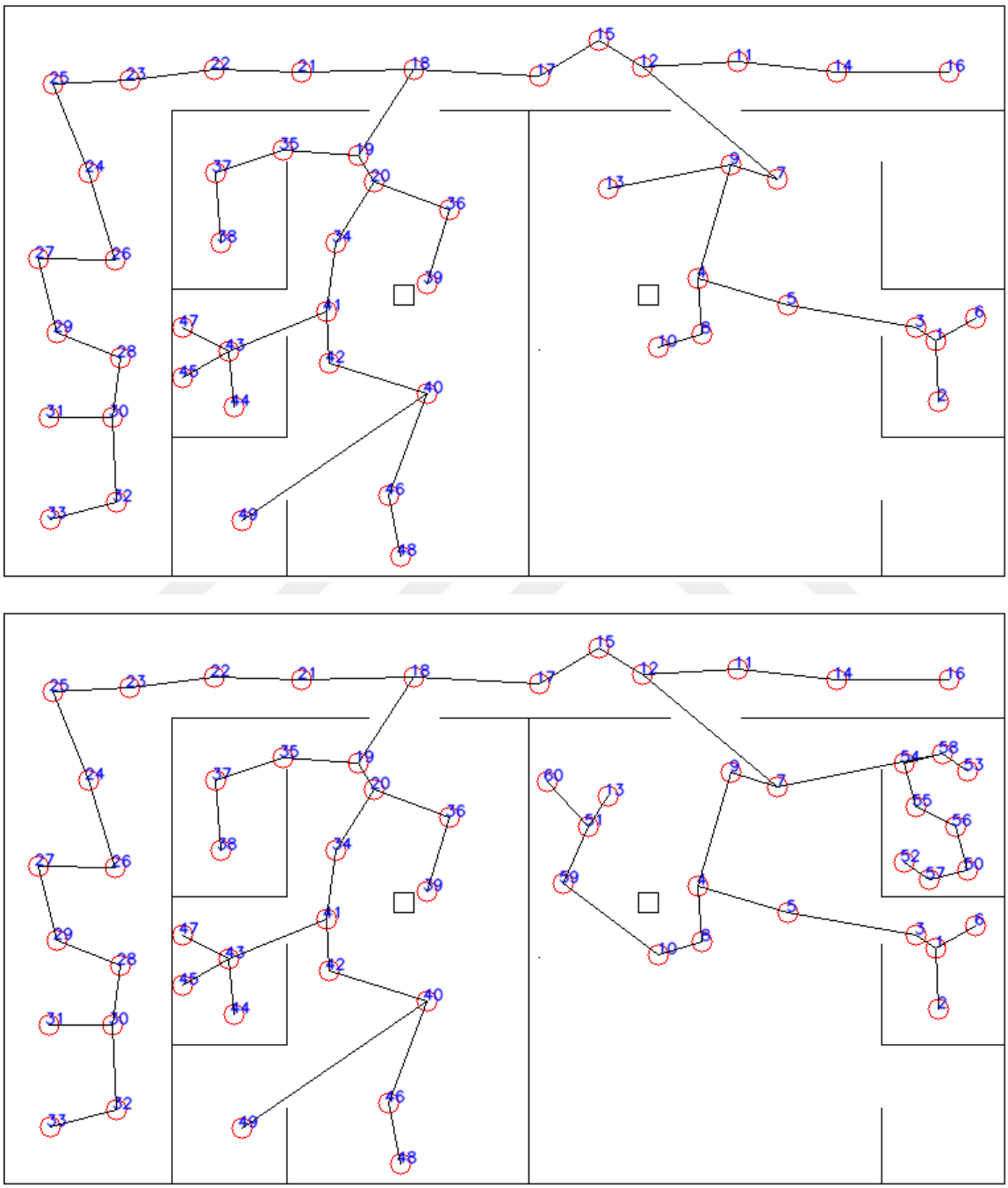
Şekil 5.8: Tek Robot Topolojik Harita - Kümeleme Adımları 3



Şekil 5.9: Tek Robot Topolojik Harita - Düğüm Adımları 1



Şekil 5.10: Tek Robot Topolojik Harita - Düğüm Adımları 2



Şekil 5.11: Tek Robot Topolojik Harita - Düğüm Adımları 3

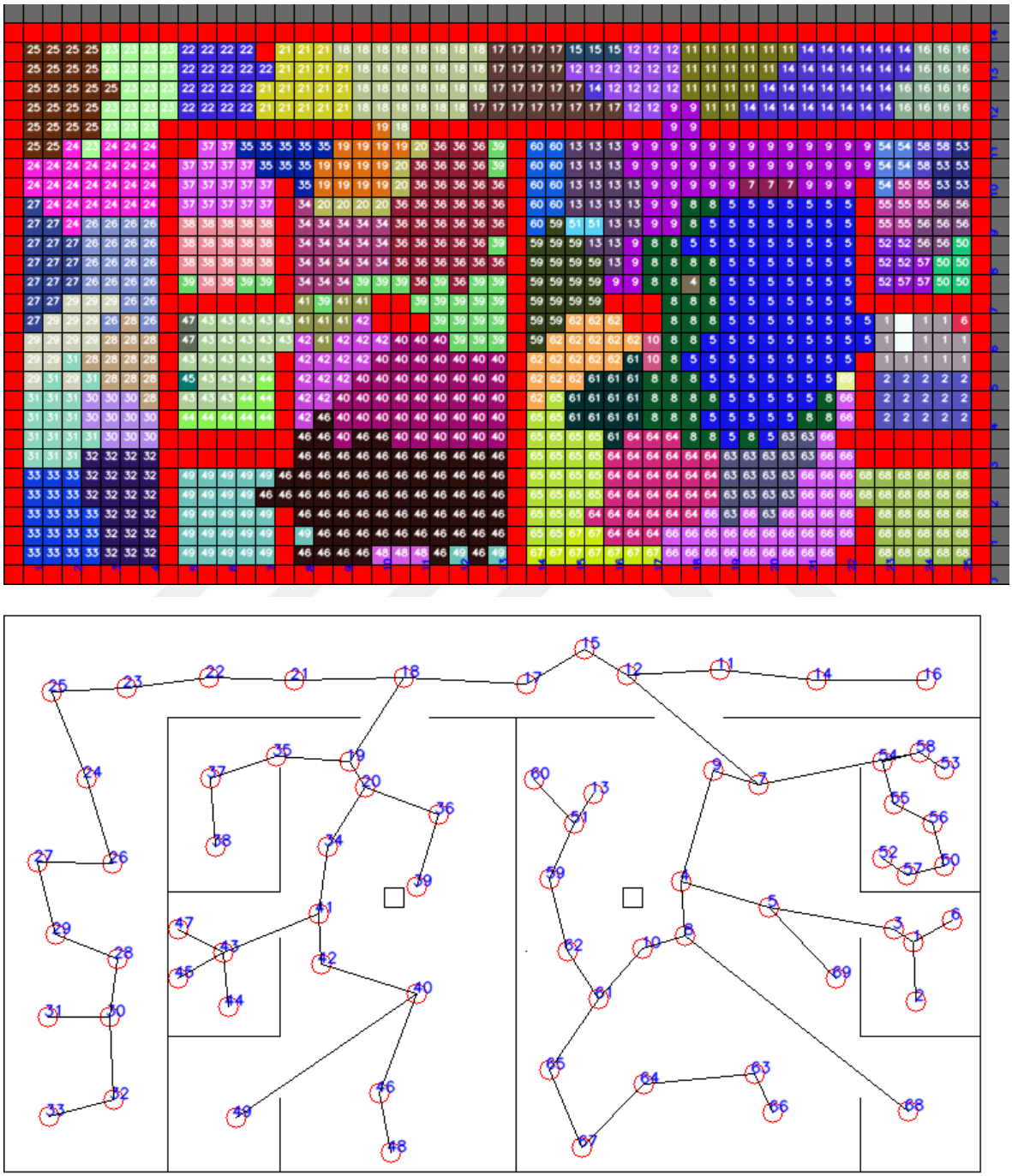
Şekil 5.6, 5.7, 5.8, 5.9, 5.10, ve 5.11'de, tek bir robot kullanılarak başarılı bir şekilde gerçeklenen bir testin adımları incelenebilmektedir. Şekil 5.6, 5.7 ve 5.8'de robotun hareketi süresince çevrimiçi şekilde elde ettiği metrik haritadaki boş hücreler kullanılarak oluşan kümeler, her hücrenin ait olduğu küme numarası belirtilerek gösterilmektedir. Şekil 5.9, 5.10 ve 5.11'de de bu kümeler kullanılarak elde edilen düğümler ve dolayısıyla bu düğümlerin bağlanmasıyla elde edilen topolojik harita gösterilmektedir. Robot bu test kapsamında (24.7, 3.95) konumundan başlatılmış ve bütün ortamı başarılı bir şekilde katettiği görülmüştür.

Şekil 5.12'de robotun hareketini tamamladığında elde edilen kümeler ve düğümler gösterilmektedir. Bu elde edilen topolojik harita ile, robot ortamdaki bir hedefe, VAH algoritmasının da yardımıyla, Dijkstra en kısa yol alggoritmasıyla oluşan yol boyunca seyrüsefer yapabilmektedir. Şekil 5.13 incelendiğinde de robotun aldığı yol görülebilmektedir.

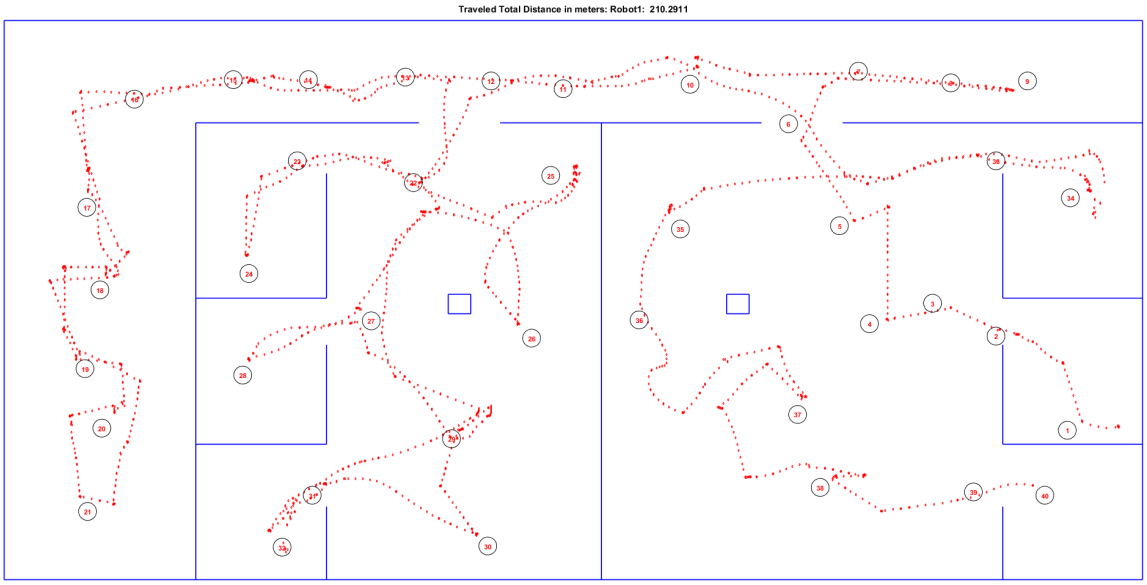
Testler için oluşturulan bir diğer ortam da Şekil 5.14'de gösterilmiştir. Bu ortamdaki testlerde, iki robot kullanılmıştır. Robotların hareket sürelerinin ve aynı anda iki robotun çalışmasıyla oluşan işlem yükünün bir miktar azaltılabilmesi için ortam rampaların yoğunluğu açısından sadeleştirilmiştir. Robotlar birbirleriyle haberleşme içerisinde olup, birbirlerinin belirli bir mesafe (testler sırasında 5m.) komşuluğunda olduklarında, o ana kadar elde ettikleri kendilerine ait metrik ve topolojik haritaları birbirleriyle paylaşmaları sağlanmaktadır. Buna ek olarak, yine robotların birbirlerine parametrik olarak belirlenen bir mesafe kadar yaklaşmaları (test kapsamında 2m.) durumunda, daha önce anlatılan çarpışmadan kaçınma algoritmasının çalıştırılması sağlanmıştır.

Şekil 5.15, önceden tek robotta olduğu gibi ROS tabanında her iki robot üzerinde çalışan düğümleri ve aralarındaki iletişim kanallarını göstermektedir. Düğümlerin işlevleri genel olarak tek robottakine benzer şekilde olmakla birlikte, görülebileceği üzere robotlar aynı zamanda kendi aralarında da haberleşebilmektedir.

Şekil 5.16,5.17,5.18 ve 5.19 ile Şekil 5.20,5.21,5.22 ve 5.23 incelendiğinde, tek robot testine benzer şekilde, çalışan her iki robot için seyrüsefer süresince elde edilen kümeler ve topolojik haritalar gösterilmiştir. Robotların birbirlerinin menzilleri içerisine girmesiyle, oluşturulan topolojik haritaların nasıl birleştiği görülebilmektedir. Burada, her robot, önce diğer robottan gelen metrik haritayı, daha sonra da birbirine bağlı düğümlerden oluşan topolojik haritayı kendisininkiyle birleştirmektedir. Eğer robot, diğer robottan gelen düğümleri o an içerisinde kendi düğümleriyle bağlayamıyor ise, metrik haritalar düğümlerin bağlanmasına elverişli konuma gelene kadar, düğümlerin bağlanması



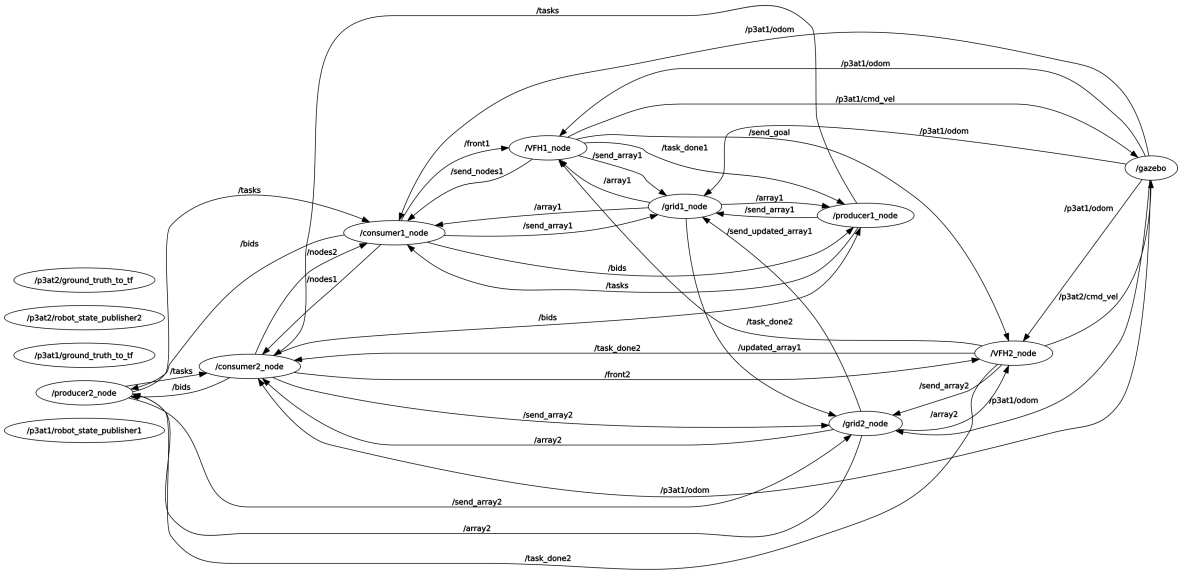
Şekil 5.12: Tek robot testi sonucu elde edilen topolojik harita



Şekil 5.13: Tek robot testi sonunda robotun izlediği yol

Şekil 5.14: Çift robot testlerinde kullanılan gazebo test ortamı

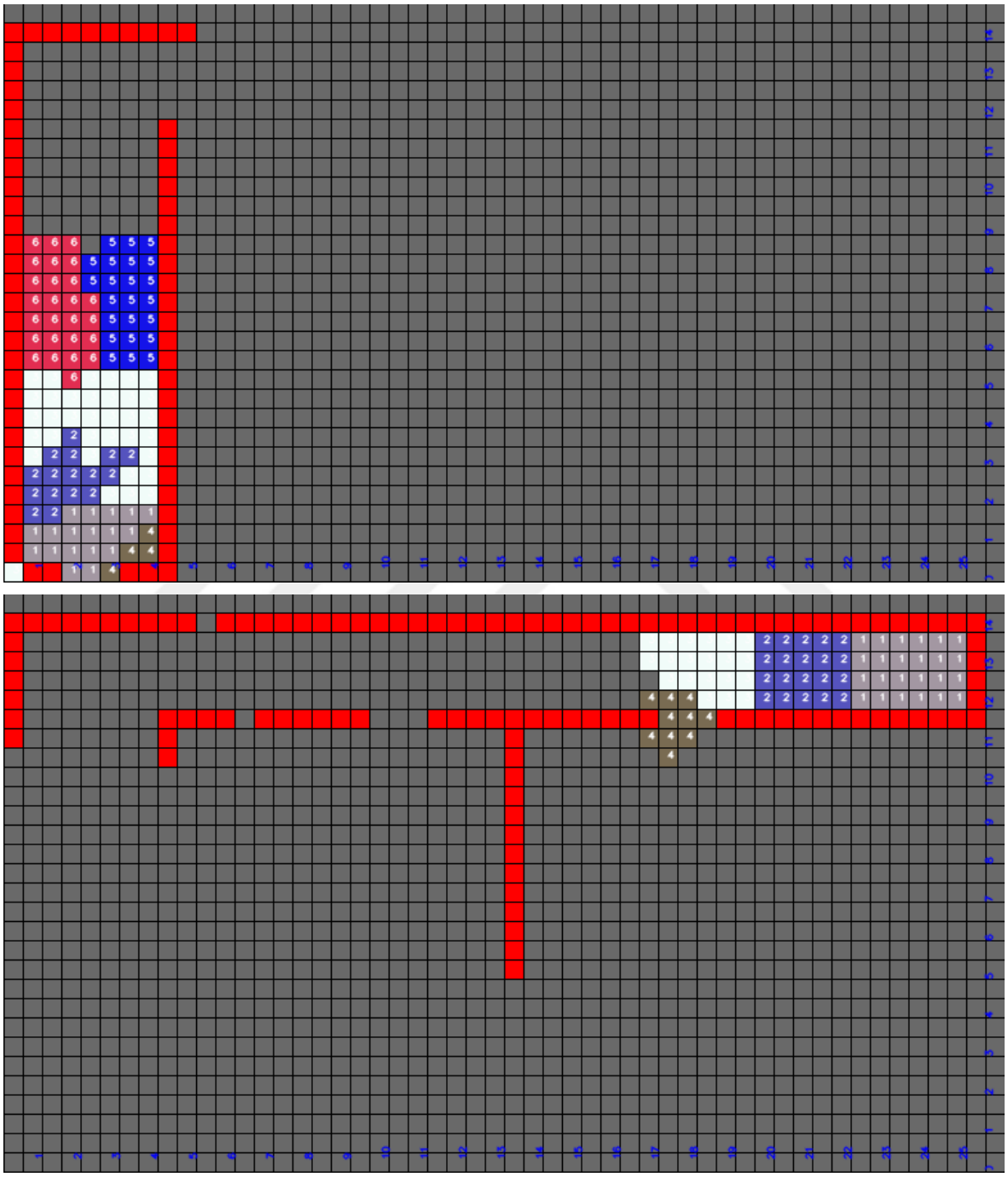
ertelenmekte ve bu sürede robotlar kendi işlerine devam etmektedir. Seyrüseferin tamamlanmasıyla oluşan topolojik harita ve düğümler Şekil 5.24'de gösterilmektedir.



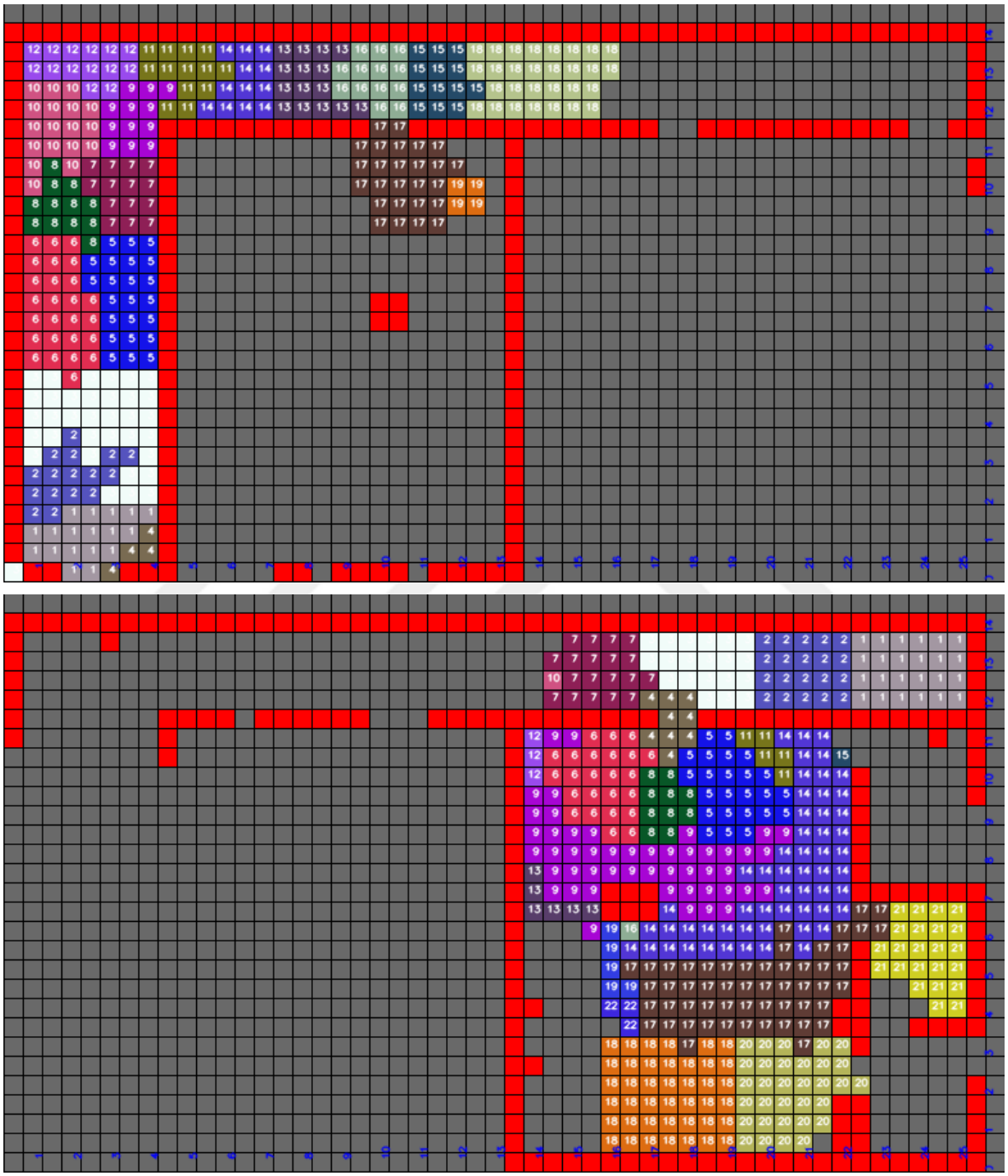
Şekil 5.15: Çift Robot ROS Düğümleri

5.25'de yine tek robot'a benzer şekilde, robotların hareketleri süresince aldıkları yol ifade edilmektedir. Bu test için, robotların biri (0.6, 0.6) konumundan, diğeri de (25, 12) konumundan başlatılmıştır. Robotların izledikleri kendilerine ait yolları kırmızı ve siyah olarak farklı renklerle gösterilmiştir. Şekildeki 66 numaralı yeşil yuvarlak ile belirtilen hedef, her iki robota da bu hedefin geldiği ve aynı zamanda robotların birbirlerinin çarpışma menzilinde olduklarını ifade etmektedir. Bu durumda, çarpışmadan sakınma algoritması gereğince, siyah renk ile belirtilen ikinci robot, siyah kare şeklinde ifade edilen 33 numaralı konumda, ilk robot 66 numaralı hedefe varana kadar beklemiştir. Benzer şekilde, 21 ve 15 numaralı üçgen şekli ile ifade edilen hedefler, robotların çarpışma önleme protokolüne girdiğini, birbirlerinin hedeflerine kendi hedeflerinden daha yakın olduklarını ve dolayısıyla hedef değiştirdiklerini ifade etmektedir. Seyrüsefer sonucunda ilk robotun yaklaşık 317m., diğer robotun da yaklaşık 280m. yol katettiği görülmüştür. Sonuç olarak, iki robot da ortamdaki hareketini başarılı bir biçimde tamamlamıştır.

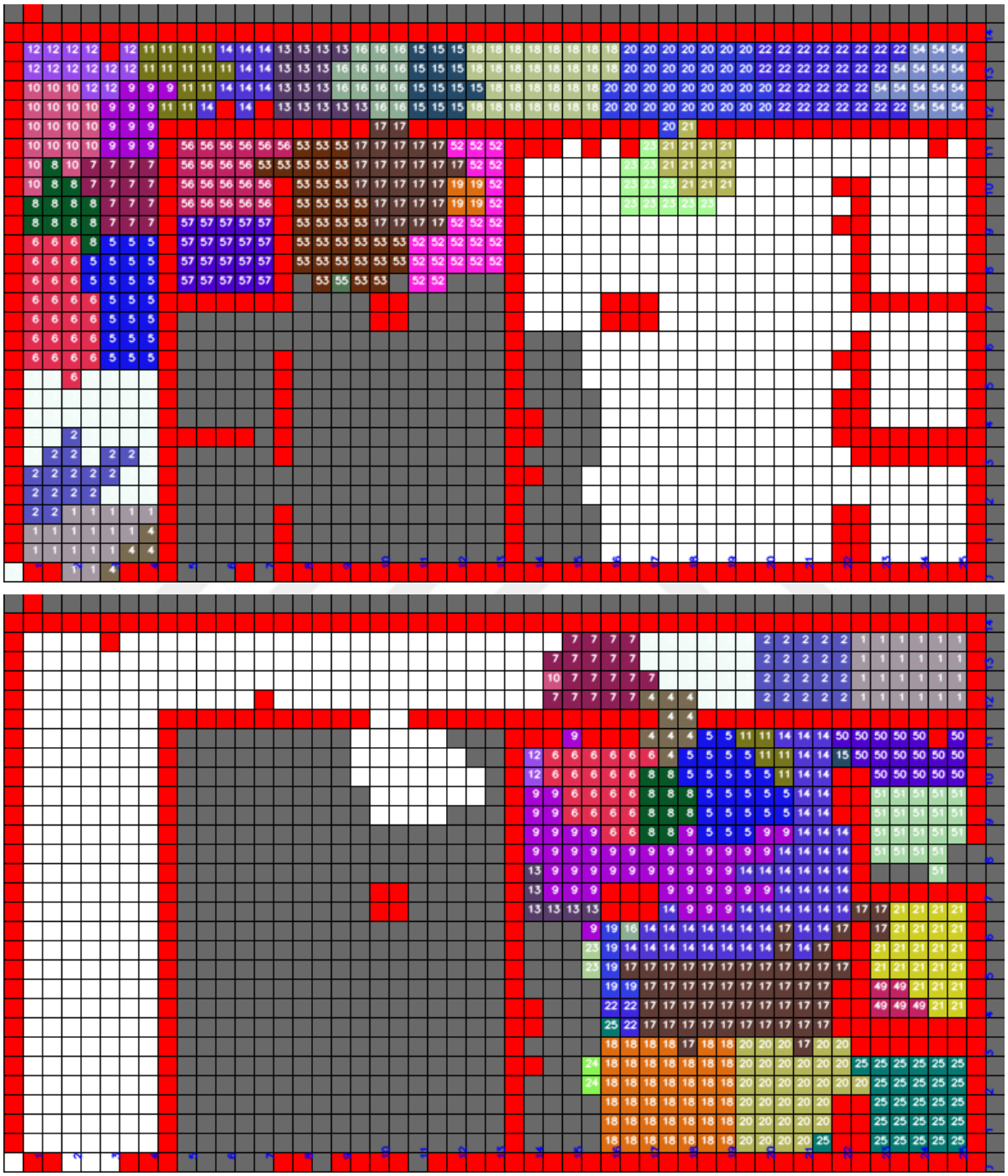
Testler gözlemlendiğinde ve elde edilen sonuçlar incelendiğinde, olası bazı problemlerle karşılaşılabildiği gözlemlenmiştir. Bu problemlerden ilkiyle metrik harita oluşturma aşamasında karşılaşılabilmektedir. Robotun açısal hız değerinin, testler sırasında kullanılan $0.40 rad/sn$ gibi bir değerden büyük olması durumunda, metrik haritada boş olarak gösterilmesi gereken bazı hücrelerin dolu olarak gösterildiği gözlemlenebilmektedir. Buna ek olarak, robot rampalardan geçisinin yumuşak olmaması ve fazla sallanmaya sebep olması da hatalı hücrelerin oluşmasına neden olabilir. Oluşturulan ortam örnek olarak düşünüldüğünde, kapıların genişliği 2 hücreye karşılık gelmektedir. Bu hücrelerden birinin dolu olarak işaretlenmesi, elde edilecek düğümlerin bağlanmasını zorlaştırabilir veya hiç



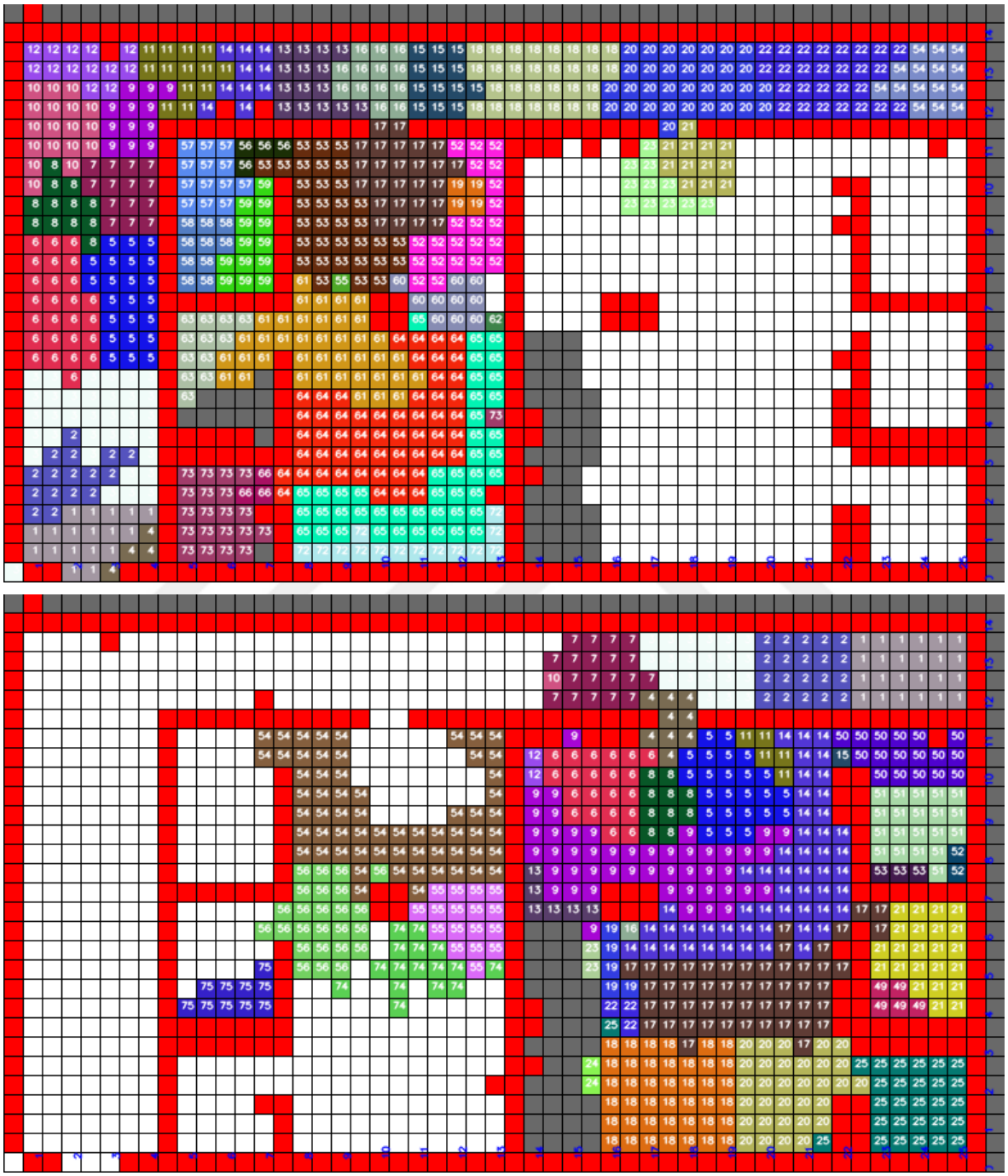
Şekil 5.16: İki Robot Topolojik Harita - Kümeleme Adımları 1



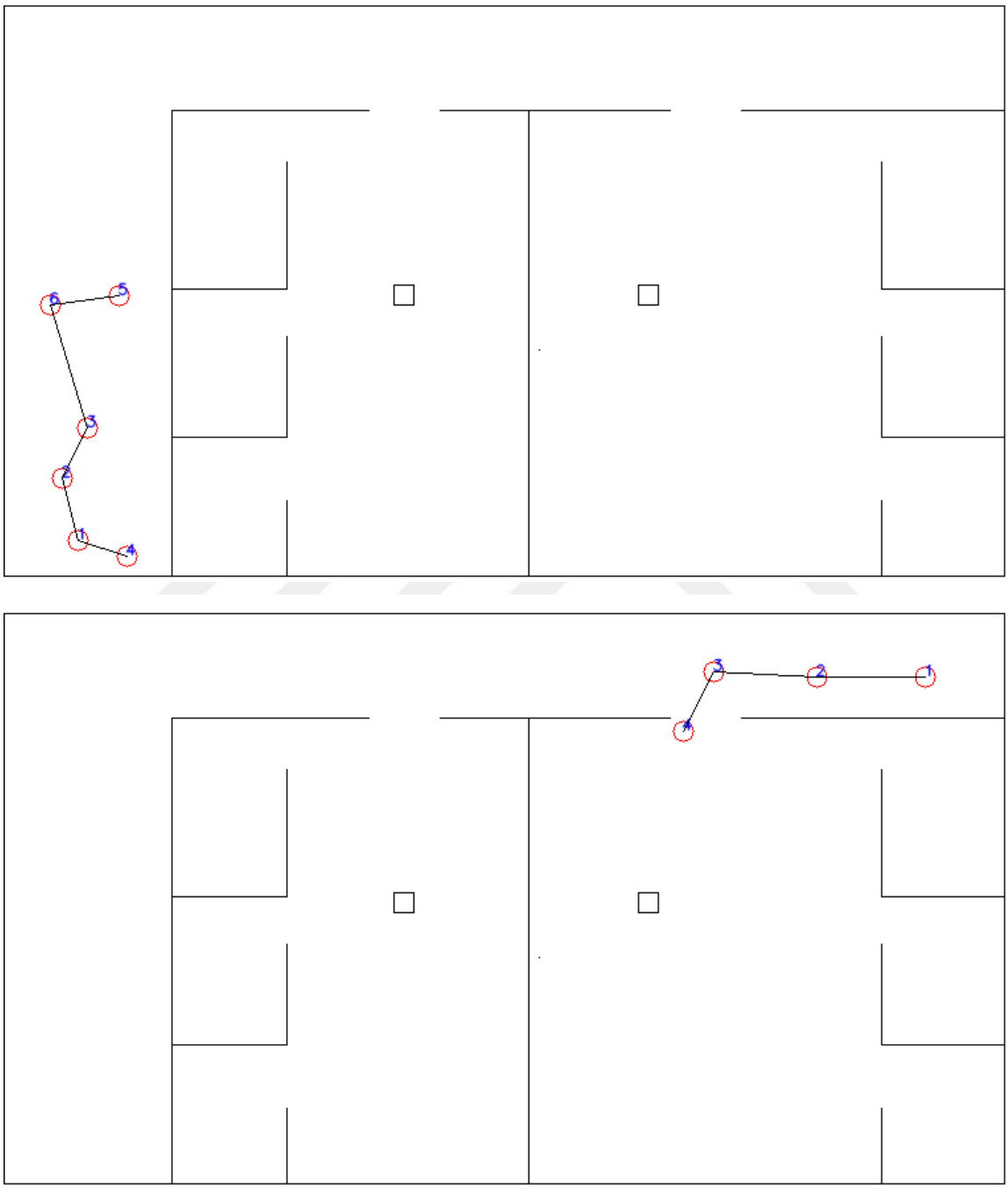
Şekil 5.17: İki Robot Topolojik Harita - Kümeleme Adımları 2



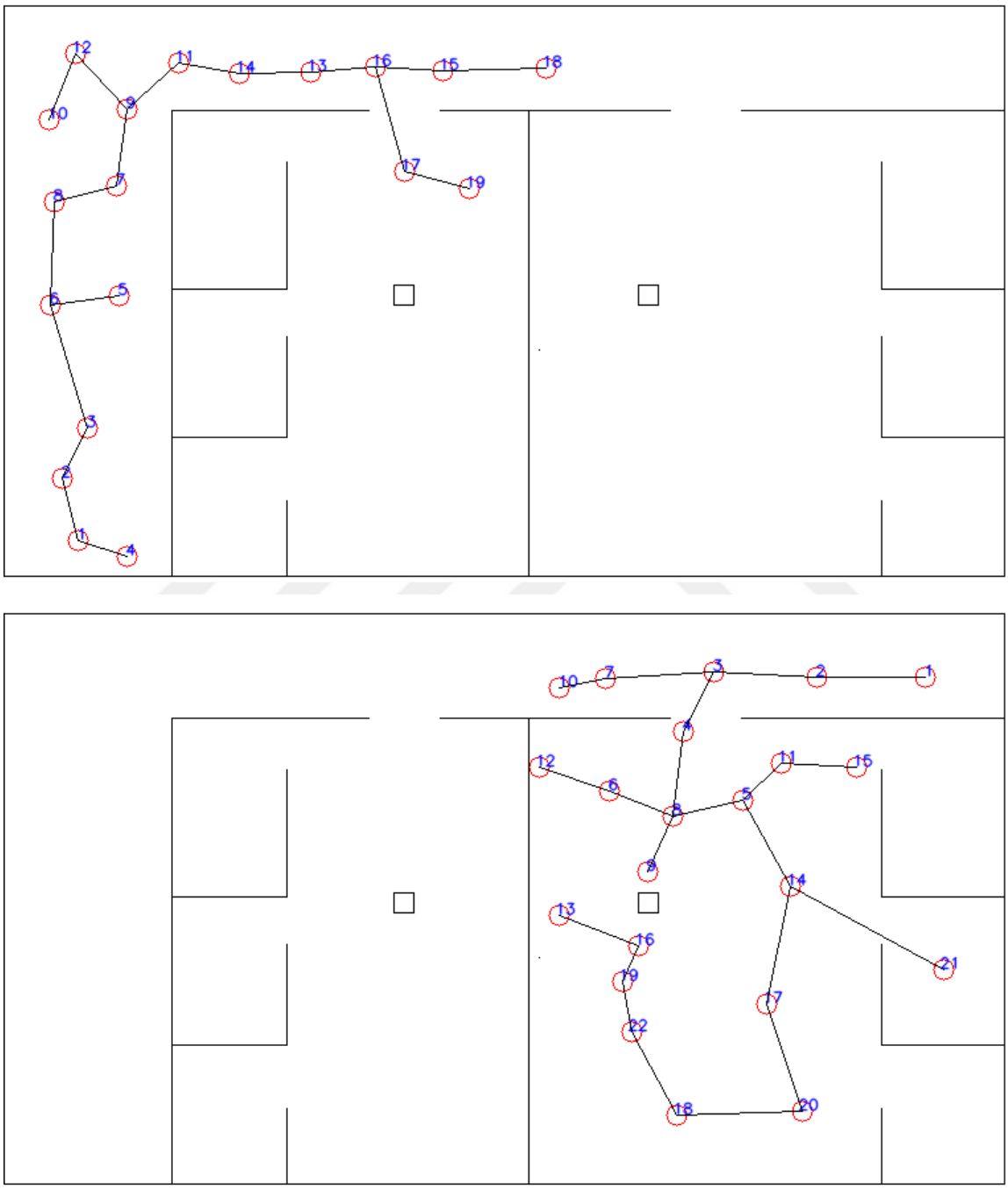
Şekil 5.18: İki Robot Topolojik Harita - Kümeleme Adımları 3



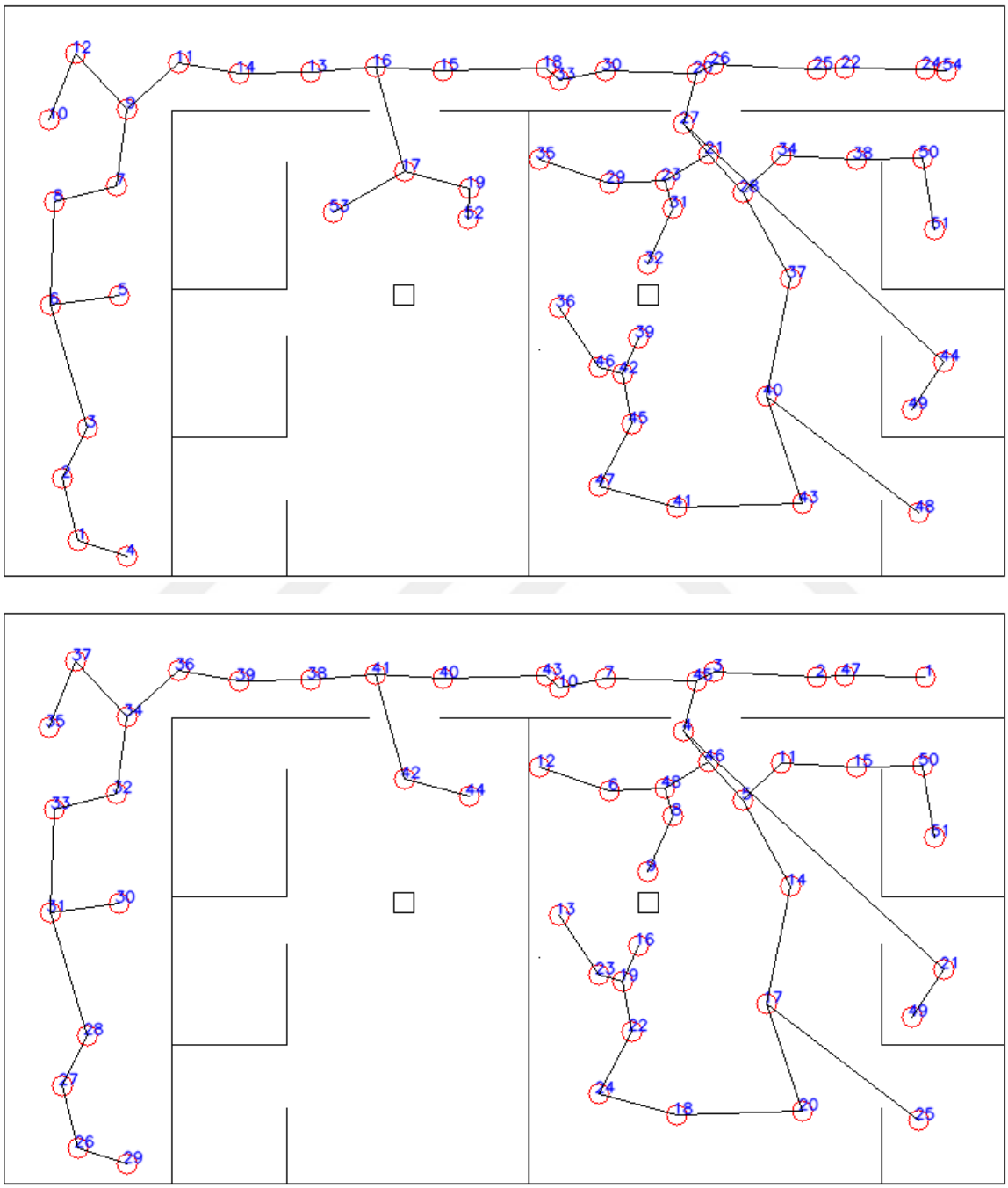
Şekil 5.19: İki Robot Topolojik Harita - Kümeleme Adımları 4



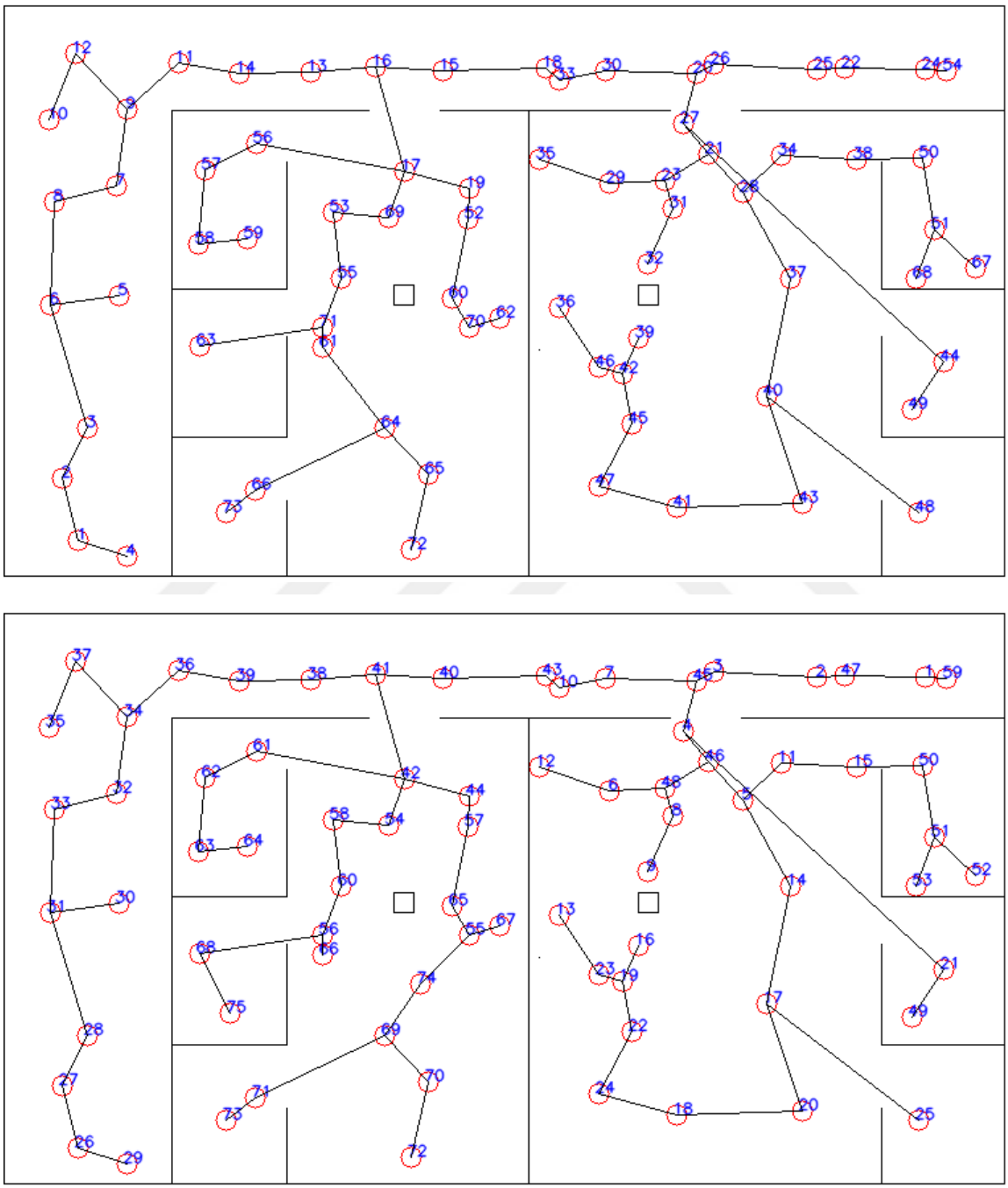
Şekil 5.20: İki Robot Topolojik Harita - Düğüm Adımları 1



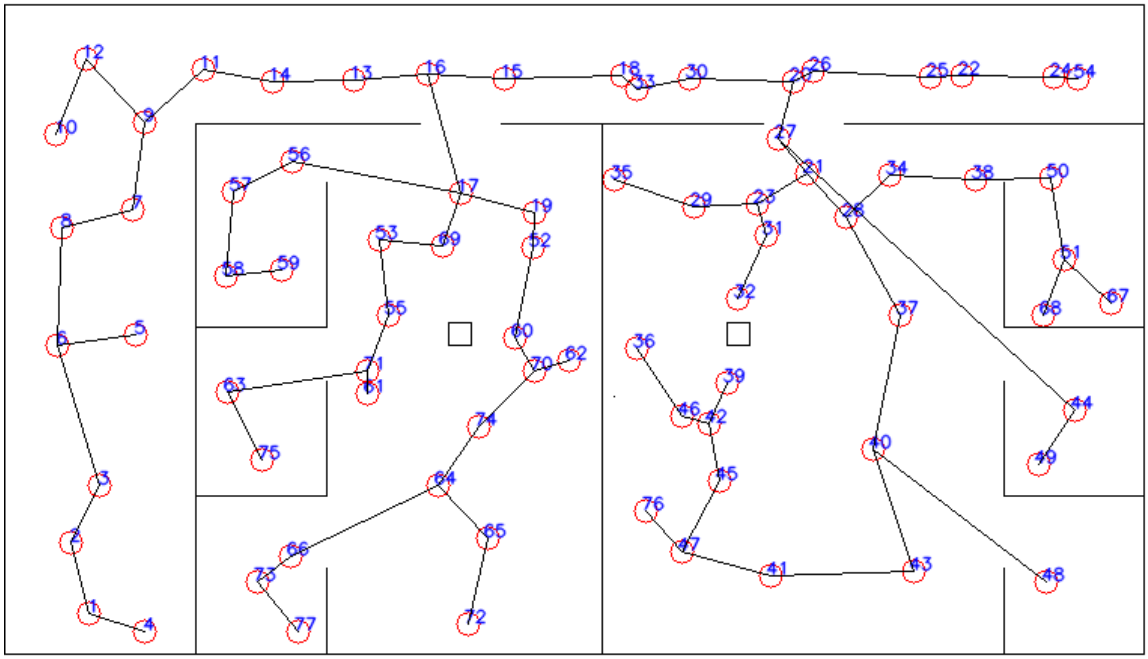
Şekil 5.21: İki Robot Topolojik Harita - Düğüm Adımları 2



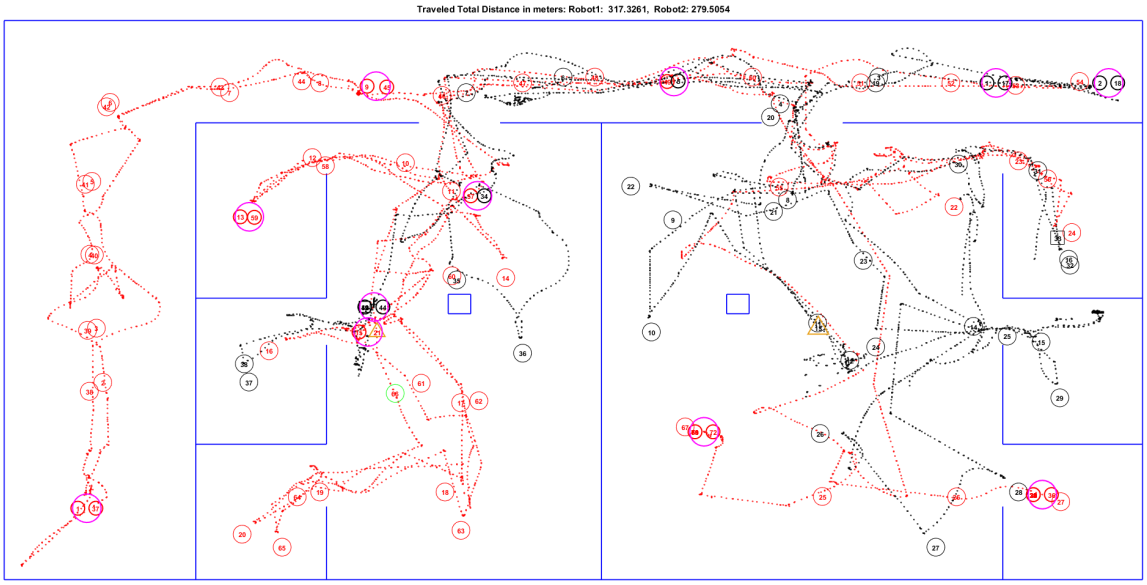
Şekil 5.22: İki Robot Topolojik Harita - Düğüm Adımları 3



Şekil 5.23: İki Robot Topolojik Harita - Düğüm Adımları 4



Şekil 5.24: İki robot testi ile oluşan topolojik harita



Şekil 5.25: İki robot testi sonucunda robotların aldıkları yol

bağlanamamasına sebep olabilir. Bu durumla karşılaşılmaması için hem açısal hem de doğrusal hız değerlerinin uygun bir şekilde ayarlanması gerekmektedir. Bu kapsamda, örnek bir gelecek çalışma olarak, metrik haritadaki hataların saptanıp düzeltilebileceği algoritma veya yöntemler düşünülebilir.

Olası sıkıntılardan bir diğerine de topolojik harita oluşumu aşamasında rastlanabilmektedir. Robotun hareketi sırasında oluşturduğu metrik haritadaki boş hücrelerden, artırımlı bir şekilde adım adım faydalanıldığına daha önceden değinmiştik. Bu elde edilen boş hücrelerin dağınık gelmesi, kümeleme yapılmasını zorlaştırabilmekte, dolayısıyla tek bir küme merkeziyle ile ifade edilebilecek bir bölgenin, birden fazla küme merkeziyle ifade edilmesi sonucu, düğüm sayısında artışa sebep olabilmektedir. Bu durumda robot çok daha sık duraklamak durumunda kalacak, çalışma süresi ve işlem yoğunluğu artacaktır.

Topolojik haritayla ilgili sıkıntılardan bahsederken, akla gelen bir diğer problem de elde edilen düğümlerin bağlanmasıyla ilgilidir. Gerçeklenen yöntemde, düğümlerin bağlanabilmesi için en küçük kapsayan ağaç algoritmasından faydalanılmaktadır. Bu algoritma ile düğümler, aralarındaki kenarların ağırlıklarının toplamı en küçük olacak ve çizgede döngüler olmayacak biçimde bağlanmaktadır. Dolayısıyla, topolojik harita için elde edilen çizge kullanılarak Dijkstra algoritmasıyla bir noktadan diğerine giderken, tek bir olası yol planı elde edilmektedir. Bu durumda, robot yakınındaki bir hedefe giderken, düğümlerin yerleşimi ve bağlanış şeklinden ötürü, uzunca bir yol planı izlemek durumunda kalabilmektedir. Yine gelecek çalışmalar düşünülerek, bu durumun çözümü için farklı bir çizge oluşturma algoritmasından faydalanılması düşünülebilir. Eğer mümkün olan bütün düğümlerin birbirine tam olarak bağlı olduğu bir çizge kullanılırsa, robotun hareketi kolaylaşabilir.

Vektör alan histogramı algoritmasının kullanımında da bazı sıkıntıların oluşabildiği gözlemlenmiştir. İlk olarak, maksimum ve minimum doğrusal ve açısal hızlar gibi değerlerin elde edilmesini etkileyen parametrelerin iyi ayarlanması gerekmektedir. Gereğinden fazla küçültüğünde, çarpışmadan sakınma konusunda yavaş kalabilmekte, çok büyütüldüğünde ise önceden bahsedildiği gibi hem hızlı hareketten ötürü metrik haritayı bozabilmekte, hem de robotun engebeli ortamda çok sarsılmasına neden olabilmektedir. Bunlara ek olarak, hedefe giderken nadir durumlarda robotun sıkışmasına neden olduğu gözlemlenmiştir. Robot seyrüseferi sırasında, yol planı için oluşan düğümleri ve Dijkstra algoritmasını kullanmakla birlikte, ilk düğüme gitmek ve son düğümden hedefe gitmek için VAH algoritmasından faydalanmaktadır. Eğer oluşan son düğüm ile hedef arasında bir engel olursa, robotun bu durumda sürekli hedefe gitmeye çalışarak sıkışıp kalabileceği gözlemlenmiştir. Bu durumun kötü konumlanmış veya bağlanamayan düğümlerden doalyı

olması durumunda, testlere baştan başlamak durumunda kalınmıştır. Öte yandan, birden fazla robot olduğu durumlarda, robotlar bu süreçte birbirlerine haberleşecek kadar yakınlaşırlarsa, sıkışan robotun metrik ve topolojik harita güncellemeleriyle kurtulabildiği de gözlemlenmiştir.

Son olarak, rampa bulma algoritmasıyla ilgili de olası sıkıntılar bulunabilmektedir. Bunlardan ilki, derinlik sensörünün görüş açısındaki kısıttan kaynaklanmaktadır. Robotun rampaya çok yakın olması veya rampanın görüş alanı dışında kalması, yine benzer şekilde robotun eğimli bir konumda bulunması gibi nedenlerle rampalar algılanamayabilmektedir. Bu durum robotun rampalara kontrolsüz biçimde girmesine neden olabilmektedir. Rampa bulma algoritmasıyla ilgili gelecek çalışmalar düşünülecek olursa, ilk akla gelecek önerilerden birisi bulunan yolnoktalarının sadece anlık olarak kullanılması yerine, her adımda yeni eklemeler ve güncellemeler yapılacak şekilde kaydedilerek kullanılmasıdır. Buna ek olarak, elde edilen yolnoktalarına örneğin ait oldukları rampa tipleri düşünülerek anlamsal sınıflar da atanabilir. Bu sınıflar kullanılarak da yerel yol planları oluşturulabilir, bu yol planları da topolojik haritadaki düğümlerle birlikte kullanılabilir.

Özetleyecek olursak, hem tek robot hem de iki robot için yapılan testlerde, robotun çoğunlukla başarılı bir şekilde seyrüseferini tamamlayabildiği gözlemlenmiştir. Elde edilen sonuçlar gösterilmiş, gerçeklenen algoritmaların başarısı gözlemlere göre yorumlanmış, olası problemler not edilmiş ve gelecek çalışmalar kapsamında bu problemlerin çözülerek seyrüsefer işinin gürbüz bir hale getirilmesi için yapılabilecekler düşünülmüştür. Gelecek çalışmalar konusunda ilk olarak kullanılan platform incelenebilir. Bir afet ortamının kullanılan test ortamına göre çok daha engebeli ve düzensiz bir zemine ve çevreye sahip olabileceği akla getirilerek, tekerlek yerine paletli bir robot tercih edilebilir. Burdan yola çıkarak, rampa bulma işinde kullanılan derinlik algılayıcısının robotun eğimli zeminlerdeki açısından etkilenmeyecek bir şekilde montelenmesi düşünülebilir. Kullanılan algoritmalar, bahsedilen olası problemlerden etkilenmeyecek hale getirilebilir veya alternatif yöntemlerin başarısı denenebilir.

KAYNAKLAR DİZİNİ

Akçakoca, M., Kılıç, I., Yayan, U., Akar, B. ve Yazıcı, A., 2014, Akıllı Tekerlekli Sandalye için Algılayıcı Tabanlı Kontrol Gerçeklemesi, *Otomatik Kontrol Ulusal Toplantısı - TOK'14*, 14–19.

Akin, H. L., Ito, N., Jacoff, A., Kleiner, A., Pellenz, J. ve Visser, A., 2012, Robocup rescue robot and simulation leagues. *AI magazine* Vol. 34.No. 1, 78.

Aladren, A, Lopez-Nicolas, G, Puig, L. ve Guerrero, J. J., 2014, Navigation assistance for the visually impaired using RGB-D sensor with range expansion.

ASUS, 2018, *https://www.asus.com/3D-Sensor/Xtion_PRO_LIVE/*. URL: https://www.asus.com/3D-Sensor/Xtion_PRO_LIVE/.

Bachrach, A., Prentice, S., He, R., Henry, P., Huang, A. S., Krainin, M., Maturana, D., Fox, D. ve Roy, N., 2012, Estimation, planning, and mapping for autonomous flight using an RGB-D camera in GPS-denied environments. *The International Journal of Robotics Research* Vol. 31.No. 11, 1320–1343.

Balakirsky, S., Carpin, S., Kleiner, A., Lewis, M., Visser, A., Wang, J. ve Ziparo, V. A., 2007, Towards heterogeneous robot teams for disaster mitigation: Results and performance metrics from robocup rescue. *Journal of Field Robotics* Vol. 24.No. 11-12, 943–967.

Baltes, J. ve Anderson, J., 2003, Flexible binary space partitioning for robotic rescue, *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*. Vol. 4. IEEE, 3144–3149.

KAYNAKLAR DİZİNİ (devam)

Bennewitz, M. ve Burgard, W., 2000, A probabilistic method for planning collision-free trajectories of multiple mobile robots, *14th European Conference on Artificial Intelligence*.

Biswas, J. ve Veloso, M., 2012, Depth camera based indoor mobile robot localization and navigation, *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 1697–1702.

Bondy, J. A., Murty, U. S. R., vd., 1976, *Graph theory with applications*. Vol. 290. Citeseer.

Borenstein, J. ve Koren, Y., 1991, The vector field histogram-fast obstacle avoidance for mobile robots. *Robotics and Automation, IEEE Transactions on* Vol. 7.No. 3, 278–288.

Bruemmer, D. J., Dudenhoeffer, D. D. ve Marble, J. L., 2002, Dynamic-Autonomy for Urban Search and Rescue., *AAAI Mobile Robot Competition*, 33–37.

Brunskill, E., Kollar, T. ve Roy, N., 2007, Topological mapping using spectral clustering and classification, *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 3491–3496.

Buschka, P. ve Saffiotti, A., 2002, A virtual sensor for room detection, *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*. Vol. 1, 637–642 vol.1.

Calisi, D., Farinelli, A., Iocchi, L. ve Nardi, D., 2005, Autonomous navigation and exploration in a rescue environment, *Safety, Security and Rescue Robotics, Workshop, 2005 IEEE International*. IEEE, 54–59.

KAYNAKLAR DİZİNİ (devam)

Calisi, D., Farinelli, A., Iocchi, L. ve Nardi, D., 2007, Multi-objective exploration and search for autonomous rescue robots. *Journal of Field Robotics* Vol. 24.No. 8, 763–778.

Casper, J. ve Murphy, R. R., 2003, Human-robot interactions during the robot-assisted urban search and rescue response at the World Trade Center. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* Vol. 33.No. 3, 367–385.

Choi, J., Choi, M., Nam, S. ve Chung, W., 2011, Autonomous topological modeling of a home environment and topological localization using a sonar grid map. English. *Autonomous Robots* Vol. 30.No. 4, 351–368. ISSN: 0929-5593.

Choset, H., 1996, *Sensor Based Motion Planning: The Hierarchical Generalized Voronoi Graph*.

Colas, F., Mahesh, S., Pomerleau, F., Liu, M. ve Siegwart, R., 2013, 3D path planning and execution for search and rescue ground robots, *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 722–727.

Doroodgar, B., Ficocelli, M., Mobedi, B. ve Nejat, G., 2010, The search for survivors: Cooperative human-robot interaction in search and rescue environments using semi-autonomous robots, *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2858–2863.

Duong, T., JianGang, Y. ve Truong, V., 2014, Application of min cut algorithm for optimal location of FACTS devices considering system loadability and cost of installation. *International Journal of Electrical Power & Energy Systems* Vol. 63, 979–987.

KAYNAKLAR DİZİNİ (devam)

Endres, F., Hess, J., Engelhard, N., Sturm, J., Cremers, D. ve Burgard, W., 2012, An evaluation of the RGB-D SLAM system, *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 1691–1696.

Fischler, M. A. ve Bolles, R. C., 1981, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* Vol. 24.No. 6, 381–395.

Fischler, M. A. ve Bolles, R. C. vspace0mm, 1987, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, *Readings in computer vision*. Elsevier, 726–740.

Gazebo, 2016, *http://gazebosim.org/*. URL: `http://gazebosim.org/`.

Gebhardt, S, Payzer, E, Salemann, L, Fettinger, A, Rotenberg, E ve Seher, C, 2009, Polygons, point clouds, and voxels, a comparison of high-fidelity terrain representations, *Simulation Interoperability Workshop and Special Workshop on Reuse of Environmental Data for Simulation—Processes, Standards, and Lessons Learned*.

Greer, D., McKerrow, P. ve Abrantes, J., 2002, Robots in urban search and rescue operations, *Proceedings of the 2002 Australasian Conference on Robotics and Automation*. Citeseer, 25–30.

Henry, P., Krainin, M., Herbst, E., Ren, X. ve Fox, D., 2010, RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments, *In the 12th International Symposium on Experimental Robotics (ISER)*. Citeseer.

KAYNAKLAR DİZİNİ (devam)

Henry, P., Krainin, M., Herbst, E., Ren, X. ve Fox, D., 2012, RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. *The International Journal of Robotics Research* Vol. 31.No. 5, 647–663.

Huang, A. S., Bachrach, A., Henry, P., Krainin, M., Maturana, D., Fox, D. ve Roy, N., 2011, Visual odometry and mapping for autonomous flight using an RGB-D camera, *International Symposium on Robotics Research (ISRR)*. Vol. 2.

Jager, M. ve Nebel, B., 2001, Decentralized collision avoidance, deadlock detection, and deadlock resolution for multiple mobile robots, *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*. Vol. 3. IEEE, 1213–1219.

Kaleci, B., 2016, İç ortamlarda anlamsal tabanlı keşif algoritmalarının geliştirilmesi.

Kaleci, B., Senler, Ç. M., Parlaktuna, O. ve Gürel, U., 2015, Constructing Topological Map from Metric Map Using Spectral Clustering, *Tools with Artificial Intelligence (ICTAI), 2015 IEEE 27th International Conference on*. IEEE, 139–145.

Kerl, C., Sturm, J. ve Cremers, D., 2013, Robust odometry estimation for RGB-D cameras, *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 3748–3754.

Khatib, O., 1985, Real-time obstacle avoidance for manipulators and mobile robots, *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*. Vol. 2, 500–505.

Koenig, N. ve Howard, A., 2004, Design and use paradigms for gazebo, an open-source multi-robot simulator, *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*. Vol. 3. IEEE, 2149–2154.

KAYNAKLAR DİZİNİ (devam)

Lee, Y. H. ve Medioni, G., 2011, A RGB-D camera based navigation for the visually impaired, *RGB-D: Advanced Reasoning with Depth Camera Workshop*.

Leishman, R. C., McLain, T. W. ve Beard, R. W., 2014, Relative navigation approach for vision-based aerial GPS-denied navigation. *Journal of Intelligent & Robotic Systems* Vol. 74.No. 1-2, 97–111.

Liu, F., Narayanan, A. ve Bai, Q., 2000, Real-time systems.

Liu, M., Colas, F. ve Siegwart, R., 2011, Regional topological segmentation based on mutual information graphs, *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 3269–3274.

Liu, Y. ve Nejat, G., 2013, Robotic urban search and rescue: A survey from the control perspective. *Journal of Intelligent & Robotic Systems* Vol. 72.No. 2, 147–165.

Lucas, B. D., Kanade, T., vd., 1981, An iterative image registration technique with an application to stereo vision.

Luxburg, U., 2007, A Tutorial on Spectral Clustering. *Statistics and Computing* Vol. 17.No. 4, 395–416.

Maier, D., Hornung, A. ve Bennewitz, M., 2012, Real-time navigation in 3D environments based on depth camera data, *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. IEEE, 692–697.

Mihelich, P., 2018, *OpenNI ROS Package*. URL: `http://wiki.ros.org/openni_launch`.

KAYNAKLAR DİZİNİ (devam)

Mozos, O. ve Burgard, W., 2006, Supervised Learning of Topological Maps using Semantic Information Extracted from Range Data, *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, 2772–2777.

Murphy, R. R., 2004, Trial by fire [rescue robots]. *Robotics & Automation Magazine, IEEE* Vol. 11.No. 3, 50–61.

Murphy, R. R. ve Burke, J. L., 2005, Up from the rubble: Lessons learned about HRI from search and rescue, *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Vol. 49. No. 3. SAGE Publications, 437–441.

Nagatani, K., Okada, Y., Tokunaga, N., Kiribayashi, S., Yoshida, K., Ohno, K., Takeuchi, E., Tadokoro, S., Akiyama, H., Noda, I., vd., 2011, Multirobot exploration for search and rescue missions: A report on map building in robocuprescue 2009. *Journal of Field Robotics* Vol. 28.No. 3, 373–387.

Open Perception, W. W., 2018, *PCL ROS package*. URL: `http://wiki.ros.org/perception_pcl`.

openCV, 2016, *http://opencv.org/*. URL: `http://opencv.org/`.

p2os, 2018, *http://wiki.ros.org/p2os*. URL: `http://wiki.ros.org/p2os`.

Peasgood, M., Clark, C. M. ve McPhee, J., 2008, A complete and scalable strategy for coordinating multiple robots within roadmaps. *IEEE Transactions on Robotics* Vol. 24.No. 2, 283–292.

KAYNAKLAR DİZİNİ (devam)

Point Cloud Library (PCL), 2016, *http://pointclouds.org/about/*. URL: http://pointclouds.org/about/.

Point Cloud Library (PCL), 2016, *http://pointclouds.org/documentation/tutorials/convex_hull_2d.php*. URL: http://pointclouds.org/documentation/tutorials/convex_hull_2d.php.

Point Cloud Library (PCL), 2016, *http://www.pointclouds.org/documentation/tutorials/cluster_extraction.php*. URL: http://www.pointclouds.org/documentation/tutorials/cluster_extraction.php.

Point Cloud Library (PCL), 2016, *http://pointclouds.org/documentation/tutorials/voxel_grid.php*. URL: http://pointclouds.org/documentation/tutorials/voxel_grid.php.

Qhull, 2016, *http://www.qhull.org/*. URL: http://www.qhull.org/.

Robocup Rescue Competitions Description. http://www.robocup.org/about-robocup/a-brief-history-of-robocup/. Accessed: 2016-05-18.

ROS.org, 2018, *http://www.ros.org/about-ros/*. URL: http://www.ros.org/about-ros/.

Rusu, R., 2009, Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments. 2009. *Disertation thesis*.

Shah, B. ve Choset, H., 2004, Survey on urban search and rescue robots. *Journal of the Robotics Society of Japan* Vol. 22.No. 5, 582–586.

KAYNAKLAR DİZİNİ (devam)

Shi, J. ve Malik, J., 2000, Normalized Cuts and Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* Vol. 22.No. 8, 888–905.

Shi, J. vd., 1994, Good features to track, *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*. IEEE, 593–600.

Simon, J. ve Kohlbrecher, S., 2018, *Hector nist gazebo ros package*. URL: `http://wiki.ros.org/hector_nist_arenas_gazebo`.

Suarez, J. ve Murphy, R. R., 2012, Using the kinect for search and rescue robotics, *Safety, Security, and Rescue Robotics (SSRR), 2012 IEEE International Symposium on*. IEEE, 1–2.

Tapus, A. ve Siegwart, R., 2006, A cognitive modeling of space using fingerprints of places for mobile robot navigation, *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, 1188–1193.

Thrun, S., 1998, Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence* Vol. 99.No. 1, 21–71.

Tomasi, C. ve Kanade, T., 1991, Detection and tracking of point features.

Ulrich, I. ve Borenstein, J., 1998, VFH+: reliable obstacle avoidance for fast mobile robots, *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*. Vol. 2, 1572–1577 vol.2.

KAYNAKLAR DİZİNİ (devam)

Valenti, R. G., Dryanovski, I., Jaramillo, C., Ström, D. P. ve Xiao, J., 2014, Autonomous quadrotor flight using onboard RGB-D visual odometry, *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 5233–5238.

Valgren, C. ve Lilienthal, A., 2008, Incremental spectral clustering and seasons: Appearance-based localization in outdoor environments, *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, 1856–1861.

Wang, Z., Liu, H., Qian, Y. ve Xu, T., 2012, Real-time plane segmentation and obstacle detection of 3D point clouds for indoor scenes, *European Conference on Computer Vision*. Springer, 22–31.

Wegner, R. ve Anderson, J., 2006, Agent-based support for balancing teleoperation and autonomy in urban search and rescue. *International Journal of Robotics & Automation* Vol. 21.No. 2, 120.

Weiss, Y., 1999, Segmentation using eigenvectors: a unifying view, *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*. Vol. 2, 975–982 vol.2.

Zhang, X., Wu, M., Peng, J. ve Jiang, F., 2009, A rescue robot path planning based on ant colony optimization algorithm, *Information Technology and Computer Science, 2009. ITCS 2009. International Conference on*. Vol. 2. IEEE, 180–183.

Zhao, J., Zhu, L., Liu, G., Liu, G. ve Han, Z., 2009, A modified genetic algorithm for global path planning of searching robot in mine disasters, *Mechatronics and Automation, 2009. ICMA 2009. International Conference on*. IEEE, 4936–4940.