T.C.
KAHRAMANMARAŞ SÜTÇÜ İMAM ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

OTONOM MOBİL ROBOTLARDA KÜMELEME YÖNTEMİ İLE TAM KAPSAMA PLANLAMA

HAMZA AYDEMİR

YÜKSEK LİSANS TEZİ

Bilişim Sistemleri Anabilim Dalı

KAHRAMANMARAŞ 2021

T.C.
KAHRAMANMARAŞ SÜTÇÜ İMAM ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

OTONOM MOBİL ROBOTLARDA KÜMELEME YÖNTEMİ İLE TAM KAPSAMA PLANLAMA

HAMZA AYDEMİR

YÜKSEK LİSANS TEZİ

Bilişim Sistemleri Anabilim Dalı

KAHRAMANMARAŞ 2021

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada orijinal olmayan her türlü kaynağa eksiksiz atıf yapıldığını bildiririm.

Hamza AYDEMİR

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

OTONOM MOBİL ROBOTLARDA KÜMELEME YÖNTEMİ İLE TAM KAPSAMA PLANLAMA
(YÜKSEK LİSANS TEZİ)

HAMZA AYDEMİR

ÖZET

Elfes, özellikle araştırma ve endüstri bağlamında robotların uygulama ve yayılma alanını genişletmek için bilinmeyen ortamlarda çalışabilecek otonom mobil robot kavramının önemini vurgulamıştır. Bu doğrultuda bir robotun, bilinmeyen bir ortamda herhangi bir noktaya yerleştirildiğinde yalnızca algılayıcılarını kullanarak gezinebilmesinin sağlanması zamanla temel ister haline gelmiştir. Otonom gezinme olarak ifade edilen bu ister, robotun yol planlaması yapması ihtiyacını doğurmuştur.

Yol planlama, robotun bir başlangıç noktasından bir hedefe verimli bir şekilde hareket etmesi için yörünge planlamasıdır. Yol planlaması, otonom mobil robotun görev tanımına göre çalışma alanının tamamını gezinmeyi gerektirebilir. Bu problem alanyazında Tam Kapsama Planlama (TKP) olarak ifade edilir. TKP engellerden kaçınırken bir alanın tüm noktalarından geçen bir yol planlama görevidir.

Bu çalışmada ele alınan problem, alanyazında en yaygın kullanılan bir TKP yöntemi olan Grid Tabanlı Tam Kapsama Planlama (GTTKP)'da kısmen dolu olan hücrenin tamamen dolu olarak işlenmesidir. Bu bağlamda, GTTKP yönteminde karşılaşılan kapsama probleminin ortamın özellikleri dikkate alınarak oluşturulacak kümeleme yönteminin kullanılabilmesi araştırma sorusu olarak belirlenmiştir. Bu doğrultuda, Arthur ve Vassilvitskii tarafından 2007 yılında alanyazına kazandırılan ve yaygın olarak kullanılan bir kümeleme algoritması ve bölümleme tekniği olan K-means++ algoritmasının kullanılabilmesi önerilmiştir.

Önerilen K-means++ Tam Kapsama Planlama (Km++TKP) yöntemi simülasyon ortamında ve gerçek dünyada robot üzerinde yapılan deneyler ile doğrulanmıştır. Aynı zamanda Km++TKP yönteminin başarımı ile GTTKP yönteminin başarımı karşılaştırılmıştır. Buna göre; tüm deneylerde Km++TKP yöntemin iç mekânı kapsama başarımının GTTKP yöntemine göre daha yüksek olduğu hesaplanmıştır.

Deneylerden elde edilen bulgular incelendiğinde, önerilen Km++TKP yönteminin simülasyon ortamında iç mekânın %96,86'sını, gerçek dünya boş iç mekânın %95,97'sini ve

gerçek dünya engel içeren iç mekânın ise %94,32'sini kapsadığı görülmüştür. GTTKP yönteminin simülasyon ortamında iç mekânın %89,76'sını, gerçek dünya boş iç mekânın %82,47'sini ve gerçek dünya engel içeren iç mekânın ise %77,95'ini kapsadığı gözlemlenmiştir.

Anahtar Kelimeler: Grid tabanlı tam kapsama planlama, k-means++ tam kapsama planlama, otonom mobil robot, ROS, tam kapsama planlama

Kahramanmaraş Sütçü İmam Üniversitesi
Fen Bilimleri Enstitüsü
Bilişim Sistemleri Anabilim Dalı, Aralık/2021

Danışman: Prof.Dr. Mehmet TEKEREK
Sayfa Sayısı: 67

COMPLETE COVERAGE PLANNING WITH CLUSTERING METHOD FOR AUTONOMOUS MOBILE ROBOTS
(M.Sc. THESIS)

HAMZA AYDEMİR

ABSTRACT

Elfes emphasized the importance of the concept of autonomous mobile robots that can work in unknown environments to expand the application and spread of robots, especially in the context of research and industry. In this direction, it has become basic requirement over time to ensure that a robot can navigate using only its sensors when placed at any point in an unknown environment. This demand, which is expressed as autonomous navigation, has led to the need for the robot to plan a path.

Path planning is trajectory planning for the robot to move efficiently from a starting point to a destination point. Path planning may require navigating the entire workspace according to the autonomous mobile robot's job description. This problem is referred to as Complete Coverage Planning (CCP) in the literature. CCP is a path planning task that passes through all points of an area while avoiding obstacles.

The problem addressed in this thesis is to process a partially filled cell as completely full in Grid Based Complete Coverage Planning (GBCCP), which is the most widely used CCP method in the literature. In this context, the use of the clustering method, which will be created by taking into account the characteristics of the environment of the coverage problem encountered in the GBCCP method, has been determined as a research question. In this direction, it is suggested to use the K-means++ algorithm, which is a widely used clustering algorithm and partitioning technique introduced in the literature by Arthur and Vassilvitskii in 2007.

The proposed K-means++ Complete Coverage Planning (Km++CCP) method has been validated by experiments on the robot in the simulation environment and the real world. At the same time, the performance of the Km++CCP method and the performance of the GBCCP method were compared. According to this; In all experiments, it has been calculated that the indoor coverage performance of the Km++CCP method is higher than the GBCCP method.

When the findings obtained from the experiments are examined, it is seen that the proposed Km++CCP method covers 96.86% of the indoor in the simulation environment, 95.97% of the real-world obstacle-free indoor, and 94.32% of the real-world indoor with obstacles. It has been observed that the GBCCP method covers 89.76% of the indoor in the simulation environment, 82.47% of the real-world obstacle-free indoor, and 77.95% of the real-world obstacle-free indoor.

Keywords: Autonomous mobile robot, complete coverage planning, grid-based complete coverage planning, k-means++ complete coverage planning, ROS

Kahramanmaraş Sütçü İmam University
Graduate School of Science and Technology
Department of Informatics Systems, December/2021

Supervisor: Prof.Dr. Mehmet TEKEREK

Page Number: 67

TEŞEKKÜR

Bu çalışmanın gerçekleştirilmesinde; tez çalışmamın tüm safhalarında engin bilgi ve birikimlerinden faydalandığım ve çalışmanın etkin bir şekilde ilerleyişi için fedakârlıklardan kaçınmayan değerli danışman hocam **Prof.Dr. Mehmet TEKEREK**'e, mesleki deneyimlerini paylaşarak çalışmama katkıda bulunan sayın hocam **Dr.Öğr.Üyesi Mehmet GÖK**'e, özellikle deney alanın kurulmasındaki desteklerinden dolayı **Öğr.Gör. Arif GÜRLER**'e ve çalışma süreci de dahil olmak üzere hayatımın her evresinde koşulsuz desteğini esirgemeyen ve yanımda olan **Ailem**'e sonsuz teşekkürlerimi sunarım.

Hamza AYDEMİR

İÇİNDEKİLER

Sayfa No

SİMGELER VE KISALTMALAR DİZİNİ

BSD : Berkeley Software Distrubiton - Berkeley Yazılım Dağıtımı

ÇS-KAK : Çoklu Sarmal - Kapsar Ağaç Kapsama - Multiple Spiral - Spanning Tree Coverage

DWA : Dynamic Window Approach - Dinamik Pencere Yaklaşımı

GTTKP : Grid Tabanlı Tam Kapsama Planlama - Grid Based Complete Coverage Planning

İHA : İnsansız Hava Aracı - Unmanned Aerial Vehicle

Km++K : K-means++ Kapsama Algoritması - K-means++ Coverage Algorithm

Km++TKP : K-means++ Tam Kapsama Planlama - K-means++ Complete Coverage Planning

LIDAR : Laser Detection and Ranging - Işık Tespiti ve Uzaklık Tayini

PGM : Portable Gray Map - Taşınabilir Gri Harita

ROS : Robot Operating System - Robot İşletim Sistemi

RViz : ROS Visualization - ROS Görselleştirme

SBC : Single Board Computer - Tek Kart Bilgisayar

SLAM : Simultaneous Localization and Mapping - Eş Zamanlı Konum Belirleme ve Haritalama

SONAR : Sound Navigation and Ranging - Ses Navigasyonu ve Uzaklık Tayini

Spiral-STC : Spiral Spanning Tree Coverage - Sarmal Kapsar Ağaç Kapsama

TKP : Tam Kapsama Planlama - Complete Coverage Planning

YAML : YAML Ain't Markup Language - YAML İşaretleme Dili Değil

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

1. GİRİŞ

Elfes (1989) özellikle araştırma ve endüstri bağlamında robotların uygulama ve yayılma alanını genişletmek için bilinmeyen ortamlarda çalışabilecek otonom mobil robot kavramının önemini vurgulamıştır. Bu doğrultuda bir robotun, bilinmeyen bir ortamda herhangi bir noktaya yerleştirildiğinde yalnızca algılayıcılarını kullanarak gezinebilmesinin sağlanması (Castellanos, Neira ve Tard´os, 2004) zamanla temel ister haline gelmiştir. Otonom gezinme olarak ifade edilen bu ister, robotun yol planlaması yapması ihtiyacını doğurmuştur.

Yol planlama, robotun bir başlangıç noktasından bir hedefe verimli bir şekilde hareket etmesi için yörünge planlamasıdır (Patle, Ganesh, Pandey, Parhi ve Jagadeesh, 2019). Robot, planlanan yörüngeyi kullanarak bir noktadan diğerine gider ve bu sayede gezinme işlemini gerçekleştirilebilir. Yol planlaması, otonom mobil robotun görev tanımına göre çalışma alanının tamamını gezinmeyi gerektirebilir. Bu problem alanyazında Tam Kapsama Planlama (TKP) olarak ifade edilir. TKP engellerden kaçınırken bir alanın tüm noktalarından geçen bir yol planlama görevidir (Galceran ve Carreras, 2013; Zhu, Tian, Sun ve Luo, 2019). Bu görev; otonom zemin temizleme robotları (Prabakaran, Elara, Pathmakumar ve Nansai, 2018; Vega-Heredia vd., 2019), yüzey boyama robotları (Seriani vd., 2015), otonom sualtı keşif araçları (Zhu, Tian, Sun ve Luo, 2019), devriye görevi yapan insansız hava araçları (Guastella vd., 2019), mayın temizleme araçları (Najjaran, Kircanski ve Goldenberg, 2000), çim biçme makineleri (Bosse, Nourani-Vatani ve Roberts, 2007), biçerdöverler (Ollis ve Stentz, 1997) ve pencere temizleyicileri (Farsi vd., 1994) gibi çok çeşitli gerçek dünya robotik uygulamalarında gerçekleştirilmektedir.

TKP yöntemleri robotun ortam hakkında bilgi toplama biçimine göre, çevrim içi ve çevrim dışı olarak iki temel yaklaşım altında sınıflandırılabilir (Choset, 2001). Çevrim içi yaklaşımda robot, gerçek zamanlı algılayıcılarından gelen ölçümleri kullanarak sürekli ortamı anlamlandırır ve kapsama planlar. Çevrim dışında ise, ortamın haritası ve robotun haritadaki konumu kullanılır (Galceran ve Carreras, 2013). Özetle, çevrim dışı yaklaşımda robot ortam bilgisine önceden sahipken çevrim içinde ortam bilgisine sahip değildir. Çevrim içi kapsama yöntemleri, dinamik ortamlarda kullanılmaya daha elverişli olmasına karşın algılayıcı maliyeti ve algılayıcılardan gelen verilerin hatalı olması gibi kısıtları barındırır (Choset, 2001). Çevrim dışı kapsama yöntemleri ise, özellikle statik ortamlarda daha fazla kapsama ve daha optimize edilmiş yol sağlar.

Çevrim dışı kapsama yöntemi ile kapsama planlaması yapan bir otonom mobil robot, gezinmek için bulunduğu ortamın haritasına ve haritaya göre bulunduğu yeri ifade eden konum bilgisine ihtiyaç duyar (Dissanayake, Durrant-Whyte ve Bailey, 2000; Saeedi, Trentini, Seto ve Li, 2016; Zaman, Slany ve Steinbauer, 2011). Harita ve konum bilgisi sayesinde robot, en uygun yolu planlayabilir (Brooks, 1986) ve yol üzerinde karşılaştığı engellerden kaçınırken güzergahını sürekli güncelleyerek hedefe ulaşma ihtimalini artırabilir (Crowley, 1985). Konum belirleme ve haritalamanın birbirine bağımlılığı, problemin karmaşıklığını arttırır ve bu iki sorunun aynı anda çözülmesini gerektirir (Saeedi, Trentini, Seto ve Li, 2016). Bailey ve Durrant-Whyte (2006), Hadji, Kazi, Hing ve Mohamed Ali'nin (2015) tanımlarından hareketle Eş Zamanlı Konum Belirleme ve Haritalama (SLAM - Simultaneous Localization and Mapping)'nın, otonom mobil robotun tutarlı bir haritayı adım adım inşa ederken aynı zamanda konumunu belirlemek için bu haritayı kullanabileceği bir tahmin süreci olduğu söylenebilir. SLAM probleminin çözümü, otonom gezinmenin altyapısını sağlar.

Bu çalışmada ele alınan problem, alanyazında en yaygın kullanılan bir çevrim dışı TKP yöntemi olan Grid Tabanlı Tam Kapsama Planlama (GTTKP)'da kısmen dolu olan hücrenin tamamen dolu olarak işlenmesidir (Almadhoun, Taha, Seneviratne ve Zweiri, 2019). Bu bağlamda, GTTKP yönteminde karşılaşılan kapsama probleminin ortamın özellikleri dikkate alınarak oluşturulacak kümeleme yönteminin kullanılabilmesi araştırma sorusu olarak belirlenmiştir. Bu doğrultuda Arthur ve Vassilvitskii tarafından 2007 yılında alanyazına kazandırılan ve yaygın olarak kullanılan bir kümeleme algoritması ve bölümleme tekniği olan K-means++ algoritması (Arthur ve Vassilvitskii, 2007)'nın kullanılabilmesi önerilmiştir.

TKP üzerine yapılan çalışmaların oldukça büyük bir bölümünde önerilen algoritmalar veya yaklaşımlar, robot ile simüle edilmeden ya da gerçek dünya koşullarında test edilmeden alanyazına sunulmuştur (Khan, Noreen, Ryu, Doh ve Habib, 2017; Yang, Tang, Zhou ve Ma, 2018; Li, Fang, Wang ve Song, 2019). Dolayısıyla; robotun mevcut enerji, ağırlık ve maksimum hız gibi özelliklerini hesaba katmadan yalnızca geometrik kısıtlamaları dikkate alınmakta ve diğer görev gereksinimleri görmezden gelinmektedir (Di Franco ve Buttazzo, 2016). Sidhu (2020) yol planlama konusunda yapılan 151 çalışmayı; deneysel tasarımlarına, metodolojilerine ve analitik tekniklerine göre incelemiştir. İnceleme sonucunda belirli kriterler dikkate alınmadan gerçekleştirilen deneylerin yanıltıcı sonuçlar verdiği ve bu sonuçlar dikkate alınarak geliştirilen algoritmaların özellikle gerçek dünya

uygulamalarında riskler doğuracağını vurgulamıştır. Bu çalışmada önerilen yöntem, simülasyon ve gerçek dünyada yapılan deneylerle araştırmada kullanılan robot üzerinde doğrulanmıştır. Bu bağlamda araştırma probleminin çözümü, yapılacak akademik çalışmalara ve robotik uygulamalara teori ve uygulamada katkı sağlayacaktır.

Sonuç olarak, bir otonom mobil robotun bilinmeyen bir ortamda, çalışma alanının tamamını ortamın özellikleri dikkate alınarak kapsayacak şekilde gezineceği bir TKP yöntemi geliştirilmiştir. Daha sonra bu yöntem alanyazında en yaygın kullanılan GTTKP yöntemi ile kapsama oranı dikkate alınarak hem simülasyon ortamında hem de gerçek dünyada karşılaştırılmış ve incelenmiştir. Kullanılan yöntemler ve önerilen yöntem ile ilgili teorik ve deneysel çalışmalar ilerleyen bölümlerde detaylı olarak anlatılmıştır.

1.1. Tezin Anahattı

Bu tez 5 temel bölümden oluşmaktadır. İlk bölüm giriştir. Konunun arkaplanı, tam kapsama planlama, tezin amacı, motivasyonu, tezde ele alınan problem ve problemin çözümüne yönelik öneri açıklanmıştır. Bölüm 2’de tam kapsama planlama detaylıca anlatılmıştır. Alanyazın araştırması paylaşılmıştır. Bölüm 3’te varolan ve önerilen yöntemlerin uygulaması için kullanılan gereçler ve yöntemler modellenerek açıklanmıştır. Bölüm 4 tezin deneysel kısmıdır. Simülasyon ve gerçek dünya deneyi tasarımları, uygulamaları ve deneyler sonrasında elde edilen bulgular bu bölümde incelenmiştir. Tüm çalışmalar ve sonuçları Bölüm 5'te özetlenerek açıklanmıştır.

2. TAM KAPSAMA PLANLAMA

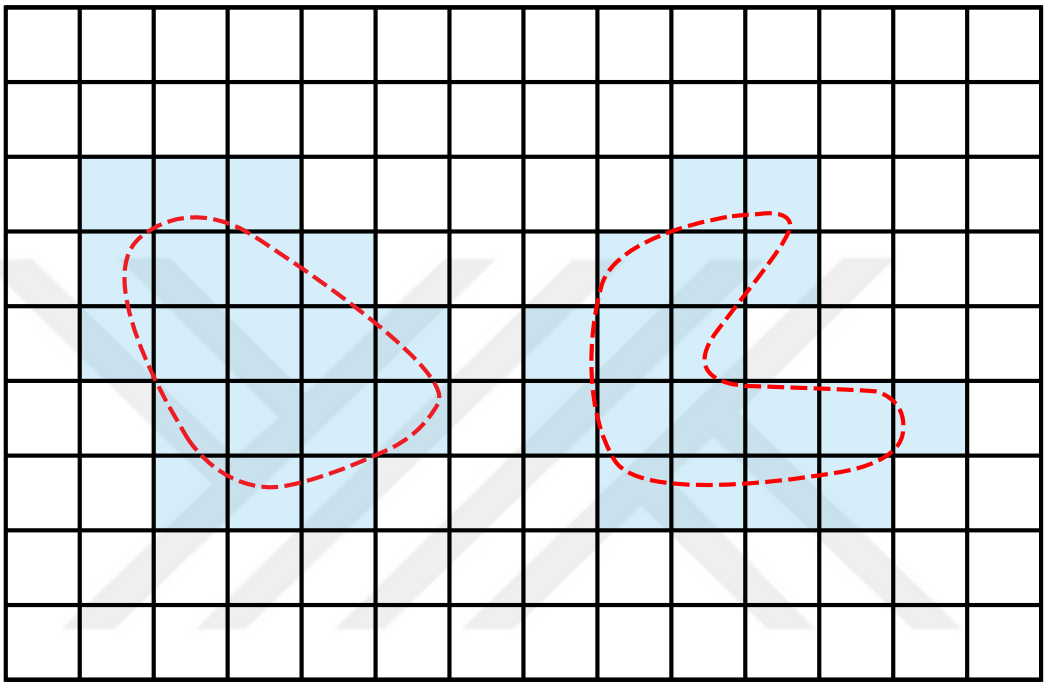
TKP algoritmaları genel olarak alanyazında, boş alanın tam olarak kapsanmasını garanti edip etmediklerine bağlı olarak sezgisel veya tam olarak sınıflandırılmaktadır. Alanyazında TKP ile ilgili yer alan ilk çalışmalardan olan Cao, Huang ve Hall (1988) çalışmasında, bir mobil robotun kapsama işlemi gerçekleştirmek için karşılaması gereken gereksinimleri tanımlanmıştır. Çalışmada bahsedilen gereksinimler şunlardır:

- Robot, hedef bölgeyi tamamen kapsayan tüm noktalardan geçmelidir.
- Robot, gezinme yolları çakışmadan bölgeyi kapsamalıdır.
- Yolların tekrarı olmaksızın sürekli ve sıralı işlem gereklidir.
- Robot tüm engellerden kaçınmalıdır.
- Kontrolde kolaylık için düz çizgiler veya daireler gibi basit hareket yörüngeleri kullanılmalıdır.
- Mevcut koşullar altında optimal bir yol beklenir.

Alanyazında yer alan TKP algoritmaları genel anlamda; hücresel ayrıştırma yaklaşımının temellerini ortaya ilk atan klasik hücresel ayrıştırma, daha genel bir engel sınıfıyla başa çıkabilen mors işlevlerinin kritik noktalarına dayalı olarak hücresel ayrıştırma, doğal yer işaretlerinin tespitine dayalı bir yaklaşım, doğrusal ortamlarda çalışan ve yalnızca temas algılayıcıları ile donatılmış robotlar için bir yaklaşım, ortamın grid temsiline dayanan yöntemler, cadde veya yol gibi grafik olarak gösterilebilen yaklaşım, 3 boyutlu ortamları kapsamak için geliştirilen çeşitli yöntemler ve çok robotlu kapsama yöntemleri gibi başlıklar altında sınıflandırılabilir (Galceran ve Carreras, 2013). Bu yaklaşımlar içerisinde alanyazında en çok kullanılan yöntem, her elemanın bir hücre ile ilişkili doluluk bilgilerini içerdiği bir dizi olarak temsil edilebildiği GTTKP yöntemidir. Bu yöntem ilk olarak Moravec ve Elfes (1985) tarafından bir mobil robota monte edilmiş bir SONAR algılayıcı kullanarak bir iç ortamın haritasını oluşturmak için önerilmiştir.

GTTKP yöntemleri, iç mekân mobil robot operasyonları için uygundur (Galceran ve Carreras, 2013). Genellikle bir grid haritadaki hücreler kare şeklinde ve robot boyutundadır (Oh, Choi, Park ve Zheng, 2004). Grid haritalarda bellek kullanımının üstel büyümesi sorunu vardır. Bunun nedeni harita çözünürlüğünün ortamın karmaşıklığından bağımsız olarak sabit kalmasıdır (Thrun, 1998). Aynı zamanda haritanın tutarlılığını sağlamak için robotun haritadaki konumunun sürekli doğru takip edilmesi gerekir (Thrun, 2003). Ayrıca grid haritada engel olarak işaretlenen hücrelerde aslında engel, hücrenin tamamını kaplamıyor

olabilir. Bu durum ilgili alanın robot tarafından tam kapsanamamasına neden olur. Ayrıca sterilizasyon ve fotogrametri gibi robotik uygulamalarda robotun kapsadığı alan, robotun fiziki ölçülerinden farklı olabilir. Örneğin; robotun kapladığı alan 50 cm^2 iken kapsadığı alan 3 m^2 olabilir. Bu durumda grid harita yönteminde her bir grid'in boyutu robotun 6 katı büyüklüğünde olacağından aslında tam dolu olmayan grid'in dolu olarak işlenme ihtimali de artar. Şekil 2.1'de bir grid harita örneği verilmiştir.



Şekil 2.1 Örnek bir grid harita ve kapsama sorunu (Galceran ve Carreras, 2013)

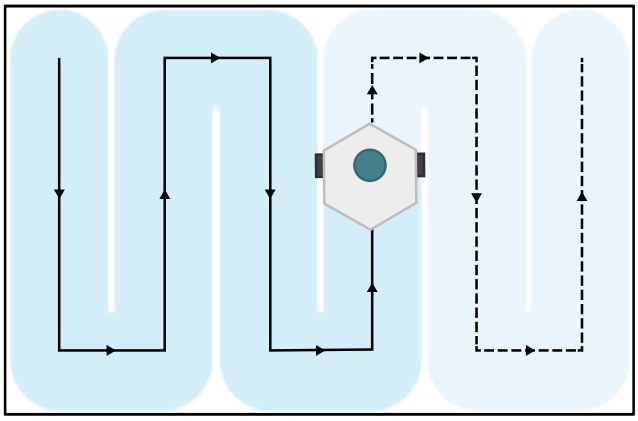
Şekil 2.1'de, noktalardan oluşan kırmızı çizgi ile ifade edilen iki farklı engel ve engelin haritada gösterimi için mavi renkte boyanan hücreler yer almaktadır. Kısmen dolu olan grid'in tamamen dolu olarak işlenmesi problemi verilen Şekil 2.1'de görülebilmektedir.

TKP stratejileri; kapsama oranı, enerji verimliliği, gezinme süresi, sağlamlık, gezinme başarısı ve engellerden kaçınma gibi belirli parametrelere odaklanır (Almadhoun, Taha, Seneviratne ve Zweiri, 2019; Le, Khanh Nhan ve Mohan, 2020). Ayrıca bir TKP algoritmasının performansı; kapsama süresi, kapsanan alanın tekrar kapsanma sayısı ve kapsama sırasında dönüş sayısının azlığı gibi kriterler dikkate alınarak incelenir (Zhou, Sun, Yu, Sun ve Sun, 2018).

2.1. Alanyazın Araştırması

Lumelsky, Mukhopadhyay ve Sun (1990) çalışmasında arazi kapsama problemi üzerine yoğunlaşmıştır. Çalışma kapsamında arazide bulunan engellerin geometrisine

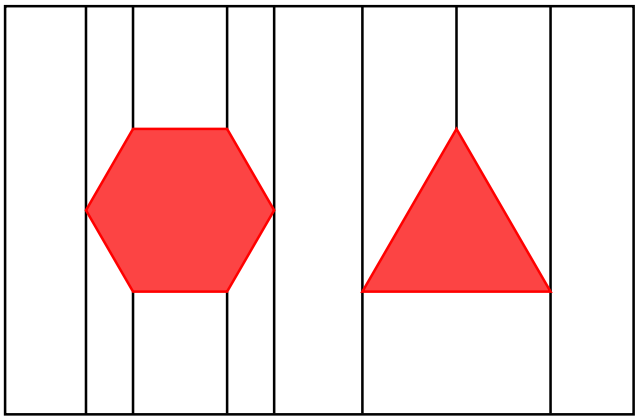
herhangi bir kısıtlama getirmeden rastgele şekillerde engellere sahip düzlemsel araziler elde etmek için iki algoritma geliştirilmiştir. Klasik tam hücresel ayrıştırma yöntemi olarak anılan bu yöntem Şekil 2.2'de görselleştirilmiştir.



Şekil 2.2 Klasik tam hücresel ayrıştırma zigzag yöntemi örneği

Şekil 2.2'de görüldüğü gibi boş alan yani engellerden arındırılmış alan hücre adı verilen basit ve örtüşmeyen bölgelere ayrılır. Hiçbir engelin bulunmadığı bu bölgelerin kapsanması kolaydır ve zigzag gibi basit hareketler kullanılarak robot tarafından kapsanabilir.

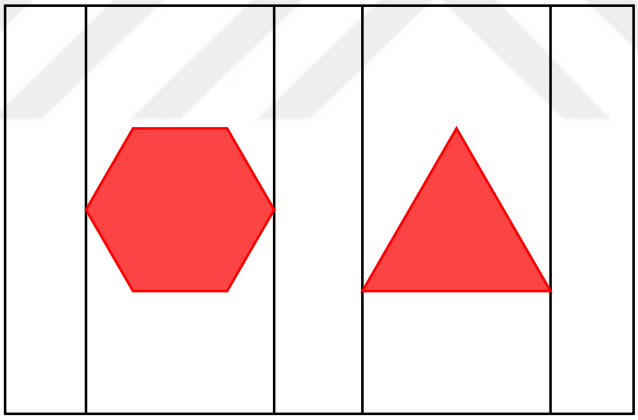
Latombe (1991) en basit tam hücresel ayrıştırma tekniklerinden biri olan yamuk ayrıştırma tekniğini önerdi. Yamuk ayrıştırmasında, Şekil 2.3'te gösterildiği gibi her hücre bir yamuktur. Teknik sayesinde ayrıştırılan her bir hücreyi kapsamak için basit ileri geri hareketler kullanılabilir. Oksanen ve Visala (2009) bu tekniği tarım alanında uygulamıştır. Çalışmanın sonunda tekniğin, makul çözümler sunmadığı durumlar olmakla birlikte tarımsal operasyonlarda kapsama problemini çözmek için daha optimal algoritmalar bulma yolunda önemli bir adım olduğu vurgulanmıştır.



Şekil 2.3 Yamuk ayrıştırma tekniği örneği

Tam kapsama yöntemlerinin büyük çoğunluğu kapsanacak alanı düzlemsel bir yüzey olarak ele alır. Ancak doğadaki bazı yüzeyler 3 boyutludur. Bu yüzeyleri kapsamak için bazı 3 boyutlu kapsama algoritmaları geliştirilmiştir. Hert, Tiwari ve Lumelsky (1996) 2 boyutlu kapsama yöntemine dayanan bir 3 boyutlu kapsama yöntemi önerdi. Çalışma kapsamında, deniz tabanını görüntüleyen otonom bir sualtı aracı kullanarak farklı derinliklerde uzanan ardışık yatay düzlemlerde düzlemsel arazi kapsama algoritması geliştirilmiştir.

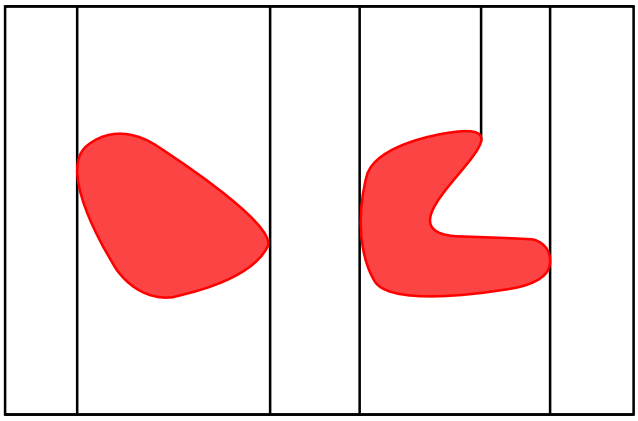
Choset ve Pignon (1998), yamuk ayrıştırmanın daha büyük hücreler oluşturmak için sezgisel olarak birleştirilebilen birçok hücre üretmesi problemini çözmek amacıyla boustrophedon hücresel ayrıştırmasını alanyazına kazandırdı. Geliştirilen teknik, otonom bir mobil robot üzerinde test edilmiştir. Boustrophedon kelimesi eski Yunancadan gelir ve "öküzün yolu" anlamına gelir. Bir öküzün bir saban ile ileri geri sürüklediği modeli belirtir. Bu teknikte, yalnızca dikey bir segmentin tepe noktasının hem üstünde hem de altında uzatılabileceği köşeler dikkate alınır (Şekil 2.4). Boustrophedon ayrıştırması, yamuk ayrıştırmasına göre hücre sayısını etkili bir şekilde azaltır. Bu sayede daha kısa kapsama yolları elde edilir.



Şekil 2.4 Boustrophedon ayrıştırma tekniği örneği

Butler, Rizzi ve Hollis (1999) bilinmeyen doğrusal ortamları çevrim içi olarak kapsayan temas algılama robotları yani menzil algılama yetenekleri olmayan robotlar için hücre ayrıştırma algoritması önerdi.

Acar, Choset, Rizzi, Atkar ve Hull (2002), Mors fonksiyonlarının kritik noktalarına dayanan yeni bir hücresel ayrıştırma yaklaşımı önererek boustrophedon ayrıştırmasını poligonal olmayan engelleri de ayrıştıracak şekilde genelleştirdi (Şekil 2.5).



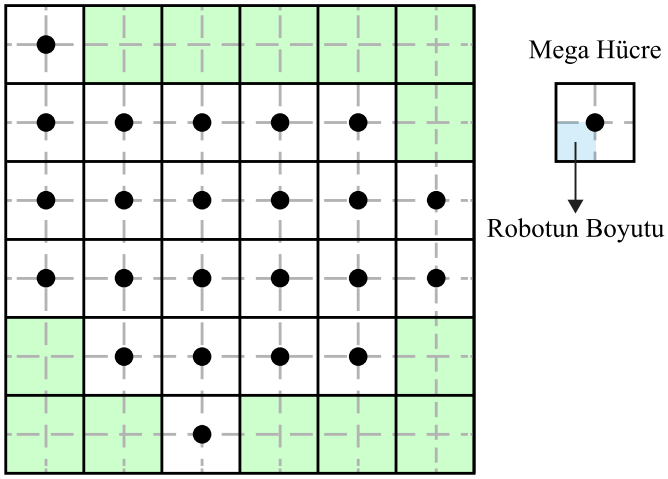
Şekil 2.5 Mors ayrıştırma tekniği örneği

Wong ve MacDonald (2003) mobil robotlar için doğal yer işaretlerinin tespitine dayalı bir çevrim içi topolojik kapsama algoritması sundu. Simülasyon ile desteklenen çalışmada ortamın büyük bir kısmının kapsandığı ortaya konmuştur. Bu yöntem, çokgen, eliptik ve doğrusal engeller de dahil olmak üzere çok çeşitli ortamların ayrıltırılması için kullanılabilir.

Moravec ve Elfes (1985) mobil robota monte edilmiş Ses Navigasyonu ve Uzaklık Tayini (SONAR - Sound Navigation and Ranging) kullanarak bir iç mekânın haritasını oluşturmak için grid yöntemini sundu. Yönteme göre; her grid hücresinin, bir engelin mevcut olup olmadığını veya tam tersi boş olup olmadığını belirten bir değeri vardır. Bu değer, ikili (engel varsa 1, engel yoksa 0) veya olasılıklı olabilir (Elfes, 1987).

Zelinsky, Jarvis, Byrne ve Yuta (1993), Moravec ve Elfes (1985)'in geliştirdiği grid harita yöntemini ilk kez tam kapsama probleminin çözümü için kullanmıştır. Çalışmada sunulan grid tabanlı tam kapsama yöntemine göre çevrim dışı tam kapsama için bir grid gösterimi kullanılır. Grid'e Dalga Cephesi olarak adlandırılan tam kapsama yol planlama algoritması uygulanarak başlangıç noktasından hedef noktaya her grid için sayısal bir değer atanır. Daha sonra başlangıç hücresinden başlayarak ve ziyaret edilmemiş en yüksek etikete sahip komşu hücreyi seçerek bir kapsama yolu hesaplanır.

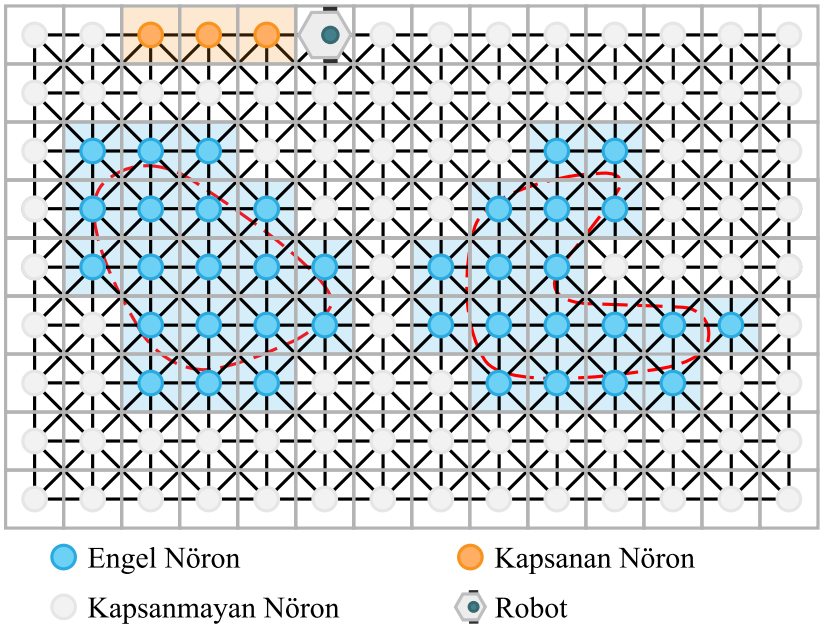
Gabriely ve Rimon (2002) mobil robotlar için çalışma alanını bir grid haritasına bölerek sistematik bir spiral yol izleyen çevrim içi bir yaklaşım olan Sarmal Kapsar Ağaç Kapsama (Spiral-STC - Spiral Spanning Tree Coverage) algoritmasını önermiştir. Bu sarmal yol, robotun yerleşik sensörlerini kullanarak artırımlı olarak oluşturduğu kısmi grid haritasının bir genişleme ağacını takip ederek oluşturulur. Yöntemde iki farklı boyutta grid hücresi kullanılır. Yönteme göre robotla aynı boyutta olan dört grid hücresi birleşerek bir mega grid hücresini oluşturur (Şekil 2.6).



Şekil 2.6 Sarmal kapsar ağaç kapsama (Spiral-STC) yöntemi örneği

Şekil 2.6'da görselleştirildiği gibi robot, her bir grid hücresini tam olarak bir kez kapsayarak tam kapsama planlar. Geliştirilen algoritma mevcut hücreden başlayarak robotun, boş alandaki ilk yeni mega hücreyi saatin tersi yönde seçerek yeni bir gezinme yönü belirler. Daha sonra, mevcut mega hücreden yenisine yeni bir yayılan ağaç kenarı oluşturulur. Algoritma yinelemeli olarak devam eder.

Luo, Yang, Stacey ve Jofriet (2002) bir zemin temizleme uygulamasına yönelik grid tabanlı çevrim içi bir sinir ağı kullanmayı önerdi. Bu yaklaşımda, her grid hücresinin köşegen uzunluğunun robot süpürme yarıçapına eşit olduğu kabul edilir. Aynı zamanda her grid hücresiyle bir nöron ilişkilendirilir. Her nöronun en yakın 8 komşusu ile bağlantıları vardır. Bu kavramlar Şekil 2.7'de görselleştirilmiştir.



Şekil 2.7 Kapsama için sinir ağı yöntemi örneği

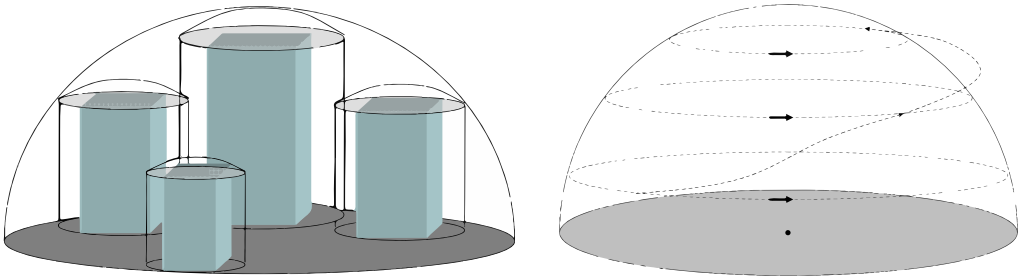
Çalışmada önerilen ve Şekil 2.7'de görselleştirilen modelin aktivite ortamı yani, belirli bir anda tüm nöronların çıktı değeri robotu temiz olmayan alanlara çekerken, robot zaten temizlenmiş alanlar ve engeller tarafından itilir. Robotun bir sonraki konumu, çevre hakkında herhangi bir ön bilgi olmaksızın, robotun mevcut konumu, mevcut konumu ile ilişkili nöronun aktivitesi ile belirlenir. Robotun mevcut konumunun (temiz veya kirli bir alanda veya bir engelin önünde olup olmadığı) duyusal bilgilerle belirlenebileceği varsayılmaktadır. Robotun konumu, sinir ağına bir girdidir. Kullanılan model, sinir ağı tasarım aşamasında geniş bir değer aralığında ayarlanabilen 6 parametreye sahiptir ve bu nedenle herhangi bir öğrenme prosedürü olmadan kapsama sağlanmaktadır. Bu yöntemin bir avantajı, durağan olmayan ortamların yani dinamik olarak değişen engellerin üstesinden gelebilmesidir.

Gonzalez, Alvarez, Diaz, Parra ve Bustacara (2005) Spiral-STC ile mobil robotlara yönelik çevrim içi bir yaklaşım olan Geri İzleme Sarmal algoritmasını önerdi. Bu algoritma duvarı takip eden bir prosedür kullanarak sadece boş hücreleri değil, kısmen dolu olanları da kapsayacak bir yenilik getirdi. Algoritma kısmen dolu hücrelerin spiral yolun dış halkasının bir parçası olduğu fikrine dayanmaktadır. Çalışma kapsamında geliştirilen algoritma simülasyon deneyleri ile doğrulanmıştır.

Özbek (2010) çalışmasında çoklu robotların tam kapsama gerçekleştirmeleri için gezinme yollarının planlanması problemini ele almıştır. Bu doğrultuda algılayıcı tabanlı gezgin çok robotlu statik çalışma alanının tam kapsanması için yeni ve diğer yöntemlere göre avantajları söz konusu olan Çoklu Sarmal - Kapsar Ağaç Kapsama (ÇS-KAK) algoritmasını geliştirmiştir. Aynı zamanda çalışma kapsamında geliştirilen algoritma alanyazında en çok kullanılan algoritmalar ile GTTKP yöntemi kullanılarak test edilmiştir.

Xu (2011) sokak veya yol ağı gibi grafik olarak gösterilebilen ortamlar için kapsama algoritması sundu. Çalışma kapsamında geliştirilen algoritma, bir grafik olarak sağlanan önceki harita bilgilerinin eksik olabileceğini ve trafikteki belirli yönlerdeki kısıtlamalar gibi (örneğin tek yönlü bir sokak) ortamdaki çevresel kısıtlamaları hesaba katar.

Cheng, Keller ve Kumar (2008) 3 boyutlu kentsel yapıların İHA'lar ile kapsanması için zaman açısından en uygun yörüngeleri planlayan bir çevrim dışı yaklaşım sundu. Şekil 2.8'de yöntem görselleştirilmiştir.



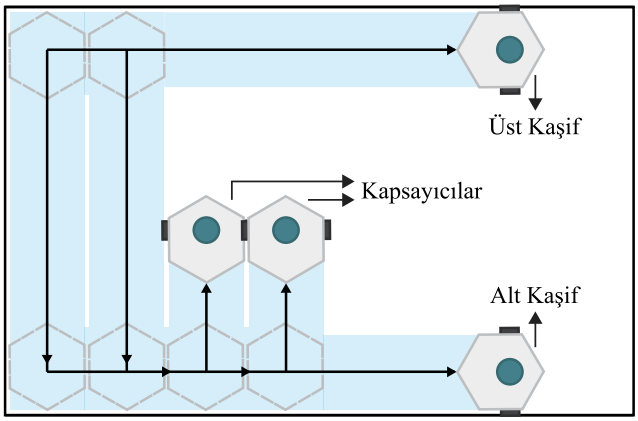
(a) (b)

Şekil 2.8 (a) 3 boyutlu kentsel yapıları kapsama ve (b) kapsama yolu örneği

Bu yaklaşımda, ilk olarak kapsanacak yapılar, yani binalar yarım küreler ve silindirler şeklinde temel geometrik cisimlere basitleştirilir (Şekil 2.8(a)). Daha sonra bu daha basit yüzeyleri kapsayacak şekilde yörüngeler planlanır (Şekil 2.8(b)). Önerilen bu yöntem, simülasyon ortamında doğrulanmıştır.

Luo ve Yang (2002) kapsama görevleri için biyolojik olarak esinlenilmiş sinir ağı yaklaşımının, robotların birbirlerini hareketli engeller olarak gördüğü çoklu robot senaryolarına basit bir uyarlamasını sundu.

Rekleitis, New, Rankin ve Choset (2009) bilinmeyen bir ortamda mobil çoklu robotlar kullanarak tam kapsama problemi için yöntem önerdi. Yöntem, Şekil 2.9'da görselleştirilmiştir.



Şekil 2.9 Çoklu robot kapsama örneği

Şekil 2.9'da görüldüğü gibi yöntemde robotlar iki rol üstlenir. Kaşifler olarak adlandırılan üyeler mevcut hedef hücrenin sınırlarını kapsarken, kapsayıcılar olarak adlandırılan diğer üyeler basit ileri geri hareketler gerçekleştirir.

Nedjati, Izbirak, Vizvari ve Arkat (2016) çalışmasında çoklu İnsansız Hava Aracı (İHA) kullanarak deprem sonrası hasar değerlendirme ve müdahale sistemi sundu. Bu sistemde, deprem bölgesinden görüntüleri toplamak ve faydalı bilgileri çıkarmak için bir

yanıt haritası oluşturulur. Ortama birden fazla İHA konuşlandırılır ve İHA'lardaki kameralardan gelen görüntüler kullanılır.

Xu, Ding ve Luo (2021) çalışmasında ilk kez bir insansız yüzey aracının yol planlaması için eksiksiz bir kapsama sinir ağı algoritması önerdi. Çalışmada, kapsama verimliliğini artırmak ve yolu daha düzenli hale getirmek için kapsama yönü terimi ile birlikte optimal bir sonraki konum karar formülü oluşturulmuştur. Sonuçlar, önerilen kapsama sinir ağı algoritması tarafından oluşturulan kapsama yolunun daha az dönüş sayısına, kilitlenmelere ve hesaplama süresine sahip olduğunu göstermektedir.

Apuroop, Le, Elara ve Sheu (2021) modüler yeniden yapılandırılabilir robotları bir iç mekânı tam kapsamak için kullanmıştır. Bu robotlar, alan gereksinimlerine göre farklı şekillerde yeniden yapılandırılarak tüm alanın kapsanmasını sağlayabilir. Robotun enerji tüketimini en aza indirirken alan kapsamını en üst düzeye çıkarmasını sağlayan bir yönteme sahip olmak hayati önem taşımaktadır. Çalışma, pekiştireçli öğrenme tekniğine dayalı, petek şeklinde bir döşeme robotu olan değiştirilmiş hTrihex için eksiksiz bir alan kapsama planlama modülü önermektedir. Çalışma kapsamında, uzun kısa süreli bellek katmanına sahip bir evrişimli sinir ağı, aktör-eleştirel deneyim tekrarı pekiştirmeli öğrenme algoritması kullanılarak eğitilmiştir. Önerilen yöntemin, daha kısa sürede minimum maliyetle kapsama planladığı simülasyon deneyleri ile doğrulanmıştır.

3. GEREÇ VE YÖNTEM

Bu bölümde; ROS, SLAM, Gmaping SLAM algoritması, GTTKP yöntemi, Km++TKP yöntemi, TurtleBot3 robot ve gezinme hakkında detaylı açıklamalara ve uygulamalara yer verilmiştir.

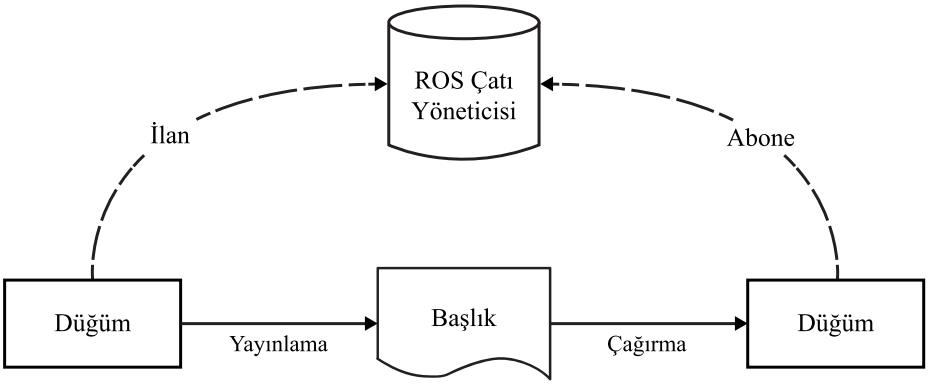
3.1. Robot İşletim Sistemi (ROS)

SLAM ve TKP algoritmaları Robot İşletim Sistemi (ROS - Robot Operating System) ile çalıştırılabilir. ROS, robot sistemlerinin etkin bir şekilde geliştirilmesine olanak sağlayan Berkeley Yazılım Dağıtımı (BSD - Berkeley Software Distrubiton) lisansı altında lisanslanan açık kaynak bir meta-işletim sistemidir. ROS; robot uygulamaları için tasarlanmış kütüphaneler, çeşitli robot platformları için geliştirilmiş paketler, SLAM algoritma paketleri, navigasyon paketleri, 3B robot simülasyon ortamları, yüksek performanslı fizik motorları ve ölçüm verilerine gürültü eklenmesi seçeneğine sahip algılayıcı eklentilerini içerir (Takaya, Asai, Kroumov ve Smarandache, 2016).

ROS ile tümleşik veya bağımsız olarak geliştirilen SLAM ve TKP algoritmaları, robot simülasyon ortamlarında simüle edilerek test edilebilir. Maliyet ve zaman tasarrufu sağlayan robot simülasyon ortamları, yeni kavramların, stratejilerin ve algoritmaların hızlı ve etkin bir şekilde test edilmesi için robotik araştırmalarında kullanılır (Qian vd., 2014). Buna ek olarak simülasyon ortamları, geliştirme döngüsünü azaltır ve farklı ortamlar için çok yönlü olarak uygulanabilir (Takaya, Asai, Kroumov ve Smarandache, 2016). ROS tabanlı Gazebo robot simülasyon ortamı robotik uygulamaların test edilmesi için güçlü bir araçtır (Qian vd., 2014).

3.1.1. ROS Yapısı

ROS yapısı, yayın yapma ve abone olma yaklaşımına sahiptir (Şekil 3.1). Bu yapı Şekil 3.1'de diyagram biçiminde gösterilmiştir.



Şekil 3.1 ROS yapısı (Li ve Shi, 2018)

Çeşitli alt görevleri yapmak üzere yazılan düğümler birbiri arasında yayın yapma ve abone olma yaklaşımını kullanarak haberleşir. Düğümler hesaplama işlemi yapabilen ve elde ettikleri verileri başlıklar altında yayınlayabilen veya başka bir düğümün yayınladığı verilere, başlıklara abone olarak erişebilen birimlerdir. Düğümlerin haberleşmesini, ROS çatı yöneticisi olarak ifade edilen ağ tabanlı bir sunucu yazılımı sağlar.

3.1.1.1. Düğüm (Node)

ROS'ta alt görevler düğümler ile gerçekleşir. Örneğin, bir düğüm kameradan gelen verileri okuyup işlerken başka bir düğüm işlenen bu verileri kullanarak motorları kontrol edebilir. Çalışan tüm düğümler `rosnode list` komutu ile listelenir. Çalışan bir düğüm hakkındaki bilgiye ise `rosnode info` *`düğüm_ismi`* komutu ile erişilir. Bu komut sayesinde ilgili düğümün abone olduğu ve yayın yaptığı başlıklar izlenebilir.

3.1.1.2. Mesaj (Message)

Düğümler arasındaki haberleşme mesajlar ile gerçekleşir. Mesaj hakkında detaylı bilgilere `rosmsg info` *`mesaj_adı`* komutu ile ilgili ulaşılabilir.

3.1.1.3. Başlık (Topic)

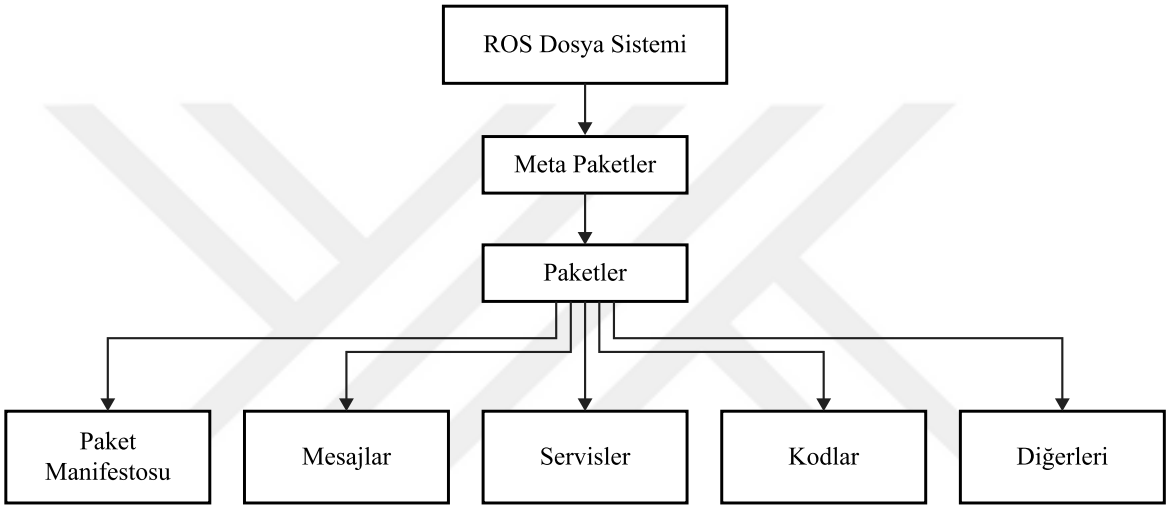
Başlık, düğümler arasındaki haberleşmeyi mesajlaşma yoluyla sağlar. Çalışan tüm başlıklar `rostopic list` komutu ile listelenir. İlgili başlığa abone olan ve başlığı yayınlayan düğüm ya da düğümler `rostopic info` *`başlık_ismi`* komutu ile görüntülenebilir. İlgili başlığın yayınladığı mesaja `rostopic echo` *`başlık_ismi`* komutu ile erişilir.

3.1.1.4. ROS Çatı Yöneticisi (ROS Master)

ROS'ta düğümler birbirleri arasında haberleşmek için önce ROS çatı yöneticisine kaydolur. Kayıt işlemi ile düğüm aktifleşir. ROS çatı yöneticisi `roscore` komutu ile başlatılır.

3.1.2. ROS Dosya Sistemi

Bir işletim sistemine benzer şekilde ROS dosyaları da sabit diskte belirlenmiş bir dosya yapısı ile depolanır. ROS dosya sistemi Şekil 3.2'de görselleştirilmiştir.



Şekil 3.2 ROS dosya sistemi (ROS Wiki, 2014)

ROS'ta en temel birim paketlerdir. Şekil 3.2'de görüldüğü üzere paketler, düğümleri, manifesto dosyasını ve yapılandırma dosyalarını içerir. Belirli görevleri yerine getirmek için oluşturulur ve kullanılırlar. Paketler rospack list komutu ile listelenir. Yeni bir paket `roscreate-pkg` *`paket_ismi`* komutu ile oluşturulur ve oluşturulan paket `rosmake` komutu ile derlenir. Paketin içinde yer alan `manifest.xml` dosyası; paketin yazarı, kütüphane bağımlılıkları ve lisansı gibi bilgileri içerir. Özelleşmiş amaçlarla hazırlanan paketlerin bir araya gelmesi ile yığın oluşur. Robotun gezinmesini sağlayan ROS navigasyon yığını buna bir örnektir.

3.2. Eş Zamanlı Konum Belirleme ve Haritalama (SLAM)

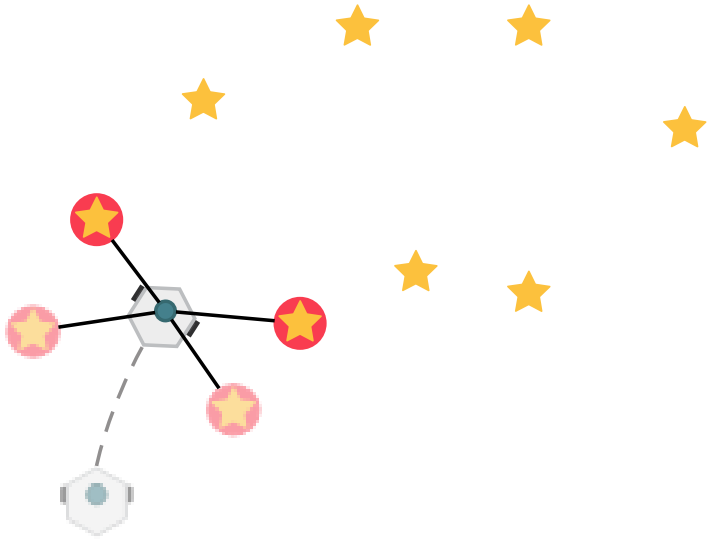
Önerilen yöntemde iç mekânın tam kapsanması için GTTKP yöntemlerinde olduğu gibi mekânın 2B harita bilgisi gerekir. Bu bilgi, alanyazında SLAM olarak ifade edilen problemin çözümüyle elde edilir. SLAM, otonom mobil robotun tutarlı bir haritayı adım

adım inşa ederken aynı zamanda konumunu belirlemek için bu haritayı kullanabileceği bir tahmin süreci olarak tanımlanabilir (Bailey ve Durrant-Whyte, 2006; Castellanos, Neira ve Tard´os, 2004; Hadji, Kazi, Hing ve Mohamed Ali, 2015; Lemaire, Lacroix ve Sol`a, 2005). SLAM, temelde işaret öğesi gözlemleme, veri ilişkilendirme, konum tahmini ve konum güncellemesi olmak üzere 4 adımda çalışır. Bu işlem adımlarını bir örnekle açıklamak gerekirse Şekil 3.3'te robotun ilk iki işaret öğesini gözlemlediği görülmektedir.

Şekil 3.3 İlk iki işaret öğesinin gözlemlenmesi

Şekil 3.3'te robot, kırmızı daire içerisindeki sarı yıldız şeklinde görselleştirilen iki işaret öğesi gözlemledi. Daha sonra bu işaret öğelerine göre konum tahmini yaptı.

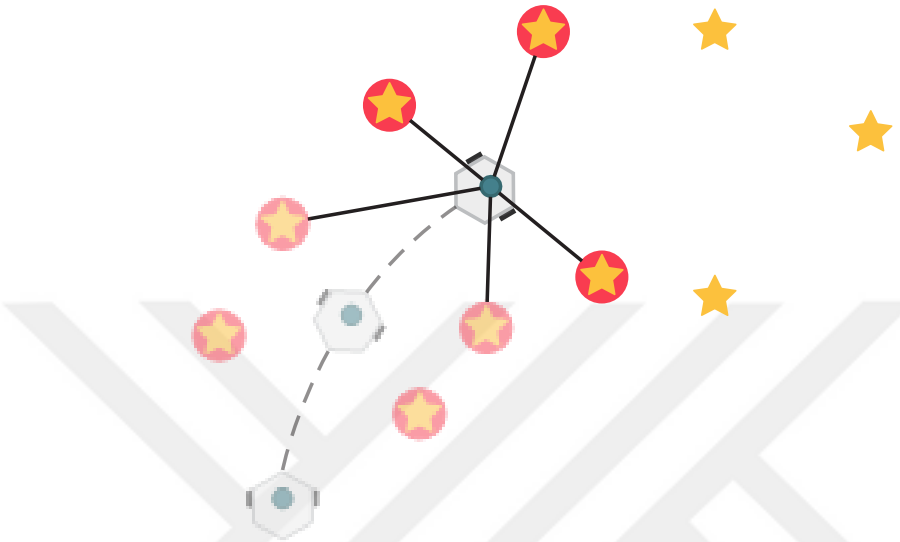
Şekil 3.4'te iki yeni işaret öğesi gözlemlendi.



Şekil 3.4 İki yeni işaret öğesinin gözlemlenmesi

Şekil 3.3'te gözlemlenen veriler ile Şekil 3.4'te görselleştirilen yeni gözlem ilişkilendirildi ve buna göre konum tahmini ve konum güncellendi. Aynı anda ilk iki işaret öğesinin yeniden gözlemlenmesi hem robot hem de işaret öğesi için konum tahminini iyileştirir.

Şekil 3.5'te üç yeni işaret öğesi daha gözlemlendi.



Şekil 3.5 Üç yeni işaret öğesinin daha gözlemlenmesi

Şekil 3.5'te görselleştirilen bu gözlem, önceki iki işaret öğesinin yeniden gözlemlenmesi ile ilişikilendirildi. Buna göre, konum tahmini ve konum güncellendi. Bu örnekten de anlaşılacağı üzere eş zamanlı konum belirleme ve haritalama bu adımların sürekli çalıştırılmasıyla gerçekleşen bir tahmin sürecidir.

3.2.1. Gmapping SLAM algoritması

SLAM probleminin robotik uygulamalarda çözümünün gerçekleştirilmesi için Gmapping, Hector SLAM ve Karto SLAM gibi algoritmalar geliştirilmiştir (Xuexi, Guokun, Genping, Dongliang ve Shiliu, 2019). Bu algoritmalar içerisinde Gmapping, dünya çapında SLAM işlemi için en güçlü olduğu düşünülen ve en yaygın kullanılan SLAM algoritmasıdır (Santos, Portugal ve Rocha, 2013; Guimarães, Oliveira, Fabro, Becker ve Brenner, 2016). Bu algoritmanın tercih edilmesinde, yaygın olarak kullanılması ve yapılandırılma kolaylığı etkili olmuştur.

Gmapping, her partikülün mekânın ayrı bir haritasını taşıyan Rao-Blackwellized parçacık filtresi temelli bir SLAM algoritmasıdır. Bu algoritma ile robot, algılayıcısı ile parçacıkları örnekleyerek mekânın haritasını oluşturur. Robotun mekânda ne kadar ilerlediğinin bilgisi ise odometri algılayıcısı kullanılarak elde edilir. SLAM için Rao-

Blackwellized parçacık filtresinin temelinde; gözlemleri $z_{1:t}$ ve robotun odometri ölçümleri $u_{0:t}$ göz önüne alındığında, robotun potansiyel $x_{1:t}$ yörüngeleri hakkında bir p ($x_{1:t}$ $z_{1:t}$, $u_{0:t}$) tahmin yapmak ve öncül harita ve yörüngeler üzerinden bir öncül değeri Eşitlik 1 ile hesaplama vardır (Stachniss, Grisetti ve Burgard, 2005).

$$p(x_{1:t}, m \vee z_{1:t}, u_{0:t}) = p(m \vee x_{1:t}, z_{1:t}) p(x_{1:t} \vee z_{1:t}, u_{0:t}) \quad (1)$$

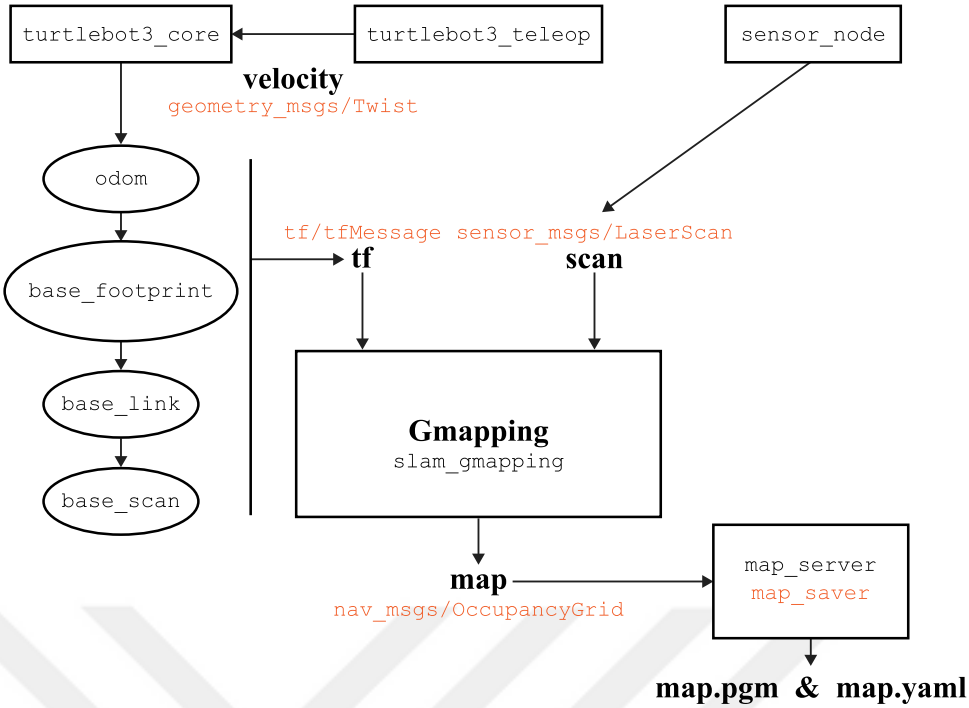
Gmapping, robotun son gözlem bölgesine göre olasılıksal dağılım yöntemlerini kullanarak bir yayılım örneği oluşturur. Bu sayede belirsiz durumlar giderilerek doğru bir harita elde etmeye daha da yaklaşılır. Yöntemde kullanılan Rao-Blackwellized parçacık filtresi ile her güncellemede yeniden örnekleme yapılarak robotun yörüngesi ve haritayı temsil eden örnekler seti güncellenir. Sıralı şekilde; örnekleme, önem ağırlığına göre atanma; yeniden örnekleme, sonuçlara göre atanma ve her bir parçacık gözlemine karşılık gelen harita tahmini yapılır. Sonuçta düşük ağırlıklı parçacıklar kaybolmuş ve yerine yüksek tahminli parçacıklar güncellenerek tahmin doğruluğu artmıştır.

Doucet, Freitas ve Gordon'un (2001) önerdiği yöntem ile yeniden örnekleme basamağı Gmapping algoritmasında kullanılmıştır. Bu yönteme göre, yeniden örneklenme işlemi için parçacıklar arasında önemli ağırlık değerine sahip olan parçacıklar dikkate alınarak seçici bir uyarmalı teknik uygulanır. Eşitlik 2'de Ne_{ff}, parçacığın öncül yörüngede ne kadar iyi bir ağırlığa sahip olduğunu ifade eder.

$$Ne_{ff} = \frac{1}{\sum_{i=1}^{N} (\widetilde{\omega}^{(i)})^2} \quad (2)$$

Eşitlik 2'de verilen $\widetilde{\omega}^{(i)}$, i parçacığının normalize edilmiş ağırlığıdır. Ne_{ff}'in parçacık sayısının, $N/2$ sayısının yarısının altına düştüğü her durumda bir yeniden örnekleme adımı gerçekleştirilir. Böylece parçacık tükenme riski azalırken daha doğru bir haritanın oluşturulması sağlanır.

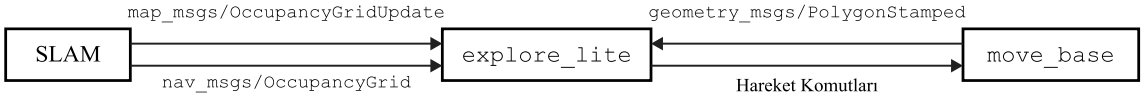
Gmapping algoritmasının ROS ile TurtleBot3 üzerinde çalıştırılması için `turtlebot3_slam` paketi kullanılır. Bu paketin diyagramı Şekil 3.6'da verilmiştir.



Şekil 3.6 `turtlebot3_slam` paketi şeması (Pyo, Cho, Jung ve Lim, 2017, s. 327)

Şekil 3.6'da görüldüğü üzere Gmapping paketi; `tf` ve `scan` adlı iki düğüme abonedir. Scan düğümü, algılayıcıdan gelen verileri ifade eder. Tf düğümü ise robotun konum dönüşümünü ifade eder. Bu iki düğümden gelen veriler Gmapping algoritması ile hesaplanır ve `map` adlı başlık yayınlanır. Bu başlık, `map_server` paketi ile okunarak harita bilgisi kaydedilebilir.

Robotun mekânın haritasını oluşturması için mekânda gezinmesi gerekir. Gezinme işlemi rastgele yer değiştirme hareketleri yapılarak ya da bir operatör tarafından kumanda edilerek sağlanabileceği gibi otonom şekilde de gerçekleştirilebilir. Bunun için çalışma kapsamında `explore_lite` adlı ROS paketi kullanılmıştır. Bu paketin tercih edilmesinde, yapılandırılma kolaylığı ve düşük kaynak tüketimi etkili olmuştur (Hörner, 2016). Hörner (2016) tarafından geliştirilen `explore_lite` paketi ile robot, açgözlü sınır temelli keşif yapar. Açgözlü sınır temelli keşif, robotun hiçbir sınır bulunmayana kadar çevresini keşfe devam edeceğini ifade eder. Şekil 3.7'de `explore_lite` paketinin diyagramı verilmiştir.



Şekil 3.7 `explore_lite` paketinin diyagramı

Şekil 3.7'deki gibi paket, Gmapping algoritması tarafından yayınlanan, `nav_msgs/OccupancyGrid` ve `map_msgs/OccupancyGridUpdate` mesajlarına abone olur. Bu sayede Gmapping tarafından oluşturulan harita takip edilerek haritada keşfedilmeyen bölgeler belirlenir. Paket, `move_base` (`<move_base>/global_costmap/costmap`) tarafından yayınlanan harita ya da SLAM ile oluşturulan haritayı kullanabilir. SLAM ile oluşturulan haritalar daha az gürültü içermesinden dolayı araştırma kapsamında `explore_lite` paketi için SLAM algoritması (Gmapping) ile oluşturulan harita kullanılmıştır.

Otonom gezinme işlemi robotun keşfetmediği alanlara sınır ataması ile gerçekleşir. Robot kendisine en yakın olan sınıra giderek sıra ile hiç sınır kalmayana kadar gezinir. Hiç sınırın kalmaması, robotun mekânın tamamını keşfettiği anlamına gelir. Bu durumda haritalama işlemi tamamlanır.

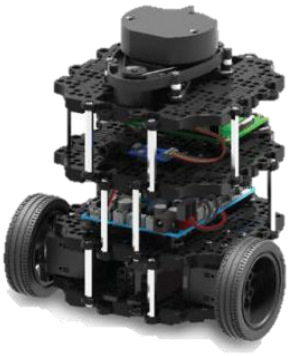
Gmapping SLAM algoritması ile oluşturulan harita, `map_server` paketi kullanılarak kaydedilir. Haritanın kaydedilmesiyle, harita bilgisini içeren `.yaml` ve `.pgm` uzantılı iki dosya oluşur. YAML İşaretleme Dili Değil (YAML - YAML Ain't Markup Language), tüm programlama dilleri için insan dostu bir veri serileştirme dilidir (YAML, 2001). YAML dosyası metin formatında harita verilerini içerirken, Taşınabilir Gri Harita (PGM - Portable Gray Map) dosyası resim formatında piksel bilgilerini içerir. Harita bilgisini metin formatında içeren YAML dosyası; `image`, `resolution`, `origin`, `occupied_thresh`, `free_thresh` ve `negate` adlı değişkenleri içerir.

Tam kapsamanın yapılacağı mekânın resim formatındaki haritasının ölçeği bu dosyada yer alan `resolution` adlı değişkende tutulur. Resolution varsayılan olarak 0.050000 değerini alır (Mishra ve Javed, 2018). Bu değer resim dosyasındaki her 1 pikselin uzunluğunun 5 cm olduğu anlamına gelir (ROS Wiki, 2019; Wonnacott, Karhumaa, Walker, Önder, 2012). İç mekânın ölçüsünün hesaplanması için piksellerin kare biçiminde olmasından yola çıkılarak harita görselindeki 1 pikselin $5cm \times 5cm = 25\ cm^2$ olduğu değeri türetilir. Bu değer, ilerleyen başlıklarda K-means++ Kapsama (Km++K) algoritmasına girdi olarak kullanılacaktır.

3.3. TurtleBot3 Robot

Gmapping algoritması kullanılarak harita oluşturmak için robot ve robotun gezinebileceği bir mekân gerekir. Çalışmada eğitim ve araştırma amaçlı kullanılabilen ROS

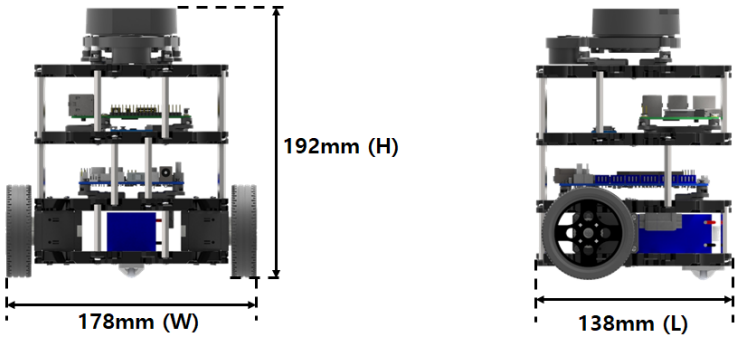
tabanlı açık kaynak bir mobil robot olan TurtleBot3 Burger kullanılmıştır. Robotun görseli Şekil 3.8'de verilmiştir.



Şekil 3.8 TurtleBot3 Burger robot

Simülasyon ortamlarının yanı sıra gerçek dünya testleri için SLAM algoritmalarını kullanan temel teknolojisi navigasyon olan TurtleBot3 mobil robot platformu geliştirilmiştir (Ackerman, 2013). ROS tabanlı açık kaynak bir mobil robot olan TurtleBot3; 2 boyutlu bir Işık Tespiti ve Uzaklık Tayini (LIDAR - Light Detection and Ranging), bir Tek Kart Bilgisayar (SBC - Single Board Computer), mikrodenetleyici, algılayıcılar ve hareketliliği sağlayan bileşenlerden oluşur (Singh, Trivedi, Sharma ve Niranjan, 2018).

TurtleBot3 Burger robotun tercih edilmesinde, SLAM ve navigasyon uygulamaları için yeterli teknik yapıya sahip olması, ROS ile uyumlu olması, açık kaynak olması ve ekonomik olması etkili olmuştur. TurtleBot3 Burger'in ölçüleri Şekil 3.9'da verilmiştir.



Şekil 3.9 TurtleBot3 Burger robotun ölçüleri

Şekil 3.9 da görüldüğü üzere, TurtleBot3 Burger 17,8 cm genişliğinde, 13,8 cm uzunluğunda ve 19,2 cm yüksekliğindedir. Bu ölçüler dikkate alındığında robotun gezinirken tek seferde yaklaşık 25 cm^2 alanı kapladığı hesaplanmıştır. Bu değer, ilerleyen başlıklarda KmK algoritmasına girdi olarak kullanılacaktır. Robotun teknik özellikleri ise Çizelge 3.1'de verilmiştir.

Çizelge 3.1 TurtleBot3 Burger robotun teknik özellikleri

Özellikler	Açıklama
Maksimum Öteleme Hızı:	0,22 metre/saniye
Maksimum Dönme Hızı:	162,7 derece/saniye
Ağırlık:	1 kg
Çalışma Süresi:	2 saat 30 dakika
Şarj Süresi:	2 saat 30 dakika
Motor:	Dynamixel XL430-W250-T
SBC:	Raspberry Pi 3 Model B
Gömülü Denetleyici:	OpenCR (32-bit ARM Cortex-M7)
Algılayıcılar:	LIDAR (HLS-LFCD2) 3 Eksenli Jiroskop 3 Eksenli İvmeölçer 3 Eksenli Mıknatısölçer

TurtleBot3 Burger'in gezinme için kullanılan maksimum öteleme hızı 0,22 m/s, dönme hızı ise 162,7 d/s'dir. OpenCR gömülü denetleyici ve Raspberry Pi 3 SBC robotun varsayılan kartlarıdır.

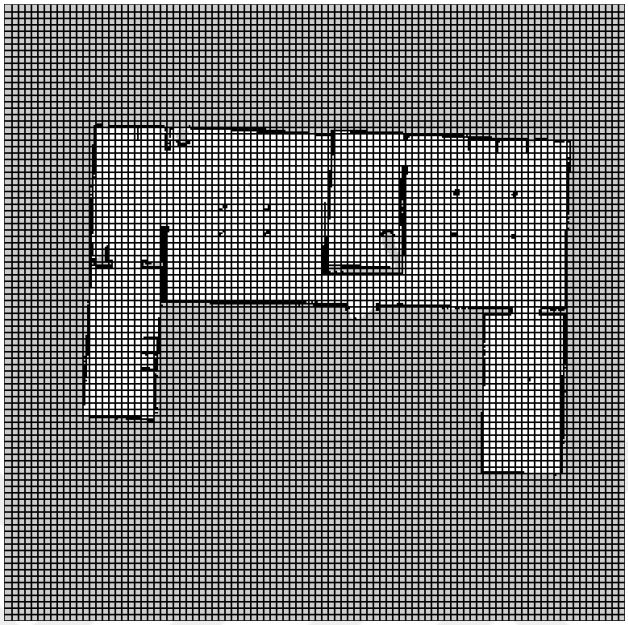
3.4. Grid Tabanlı Tam Kapsama Planlama (GTTKP)

TKP için alanyazında en çok kullanılan yöntem, her elemanın bir hücre ile ilişkili doluluk bilgilerini içerdiği bir dizi olarak temsil edilebildiği GTTKP yöntemidir (Almadhoun, Taha, Seneviratne ve Zweiri, 2019; Galceran ve Carreras, 2013). Bu yöntem ilk olarak Moravec ve Elfes (1985) tarafından bir mobil robota monte edilmiş bir SONAR algılayıcı kullanarak bir iç ortamın haritasını oluşturmak için önerilmiştir. Yönteme göre; her grid hücresinin, bir engelin mevcut olup olmadığını veya tam tersi boş olup olmadığını belirten bir değeri vardır. Bu değer, ikili (engel varsa 1, engel yoksa 0) veya olasılıklı olabilir (Elfes, 1987).

Çalışmada ele alınan problem, GTTKP yönteminde kısmen dolu olan hücrenin tamamen dolu olarak işlenmesidir (Almadhoun, Taha, Seneviratne ve Zweiri, 2019). Bu yöntemin bir diğer sınırlılığı ise robotun tek seferde kapsadığı alanın robotun fiziki ölçüsü ile aynı olmasını ve kare olmasını gerektirmesidir (Oh, Choi, Park ve Zheng, 2004).

GTTKP yönteminde kapsama; iç mekânın haritasının, robotun uzun kenarının ölçüsünde eşit kare grid'lere bölünmesiyle sağlanır. GTTKP yönteminin uygulamasında Ubuntu Linux dağıtımının 18.04 (Bionic Beaver) sürümüne yönelik yayımlanan ROS'un Melodic Morenia sürümü kullanılmıştır. Yazılım geliştirme sürecinde ise Python

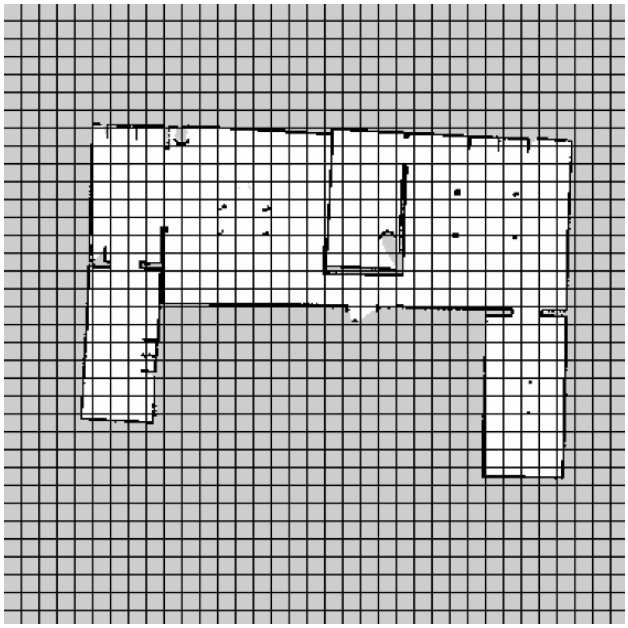
programlama dili kullanılmıştır. Yöntemin uygulanış biçiminin anlaşılması için iç mekânın, robotun ölçüsü dikkate alınarak oluşturulan grid haritası Şekil 3.10'da örnek olarak verilmiştir.



Şekil 3.10 Gerçek boyutlara göre iç mekânın grid haritası

Şekil 3.10'da verilen haritadaki her bir grid robotun boyutundadır. GTTKP yönteminde bu harita kullanılarak kapsama planlanır.

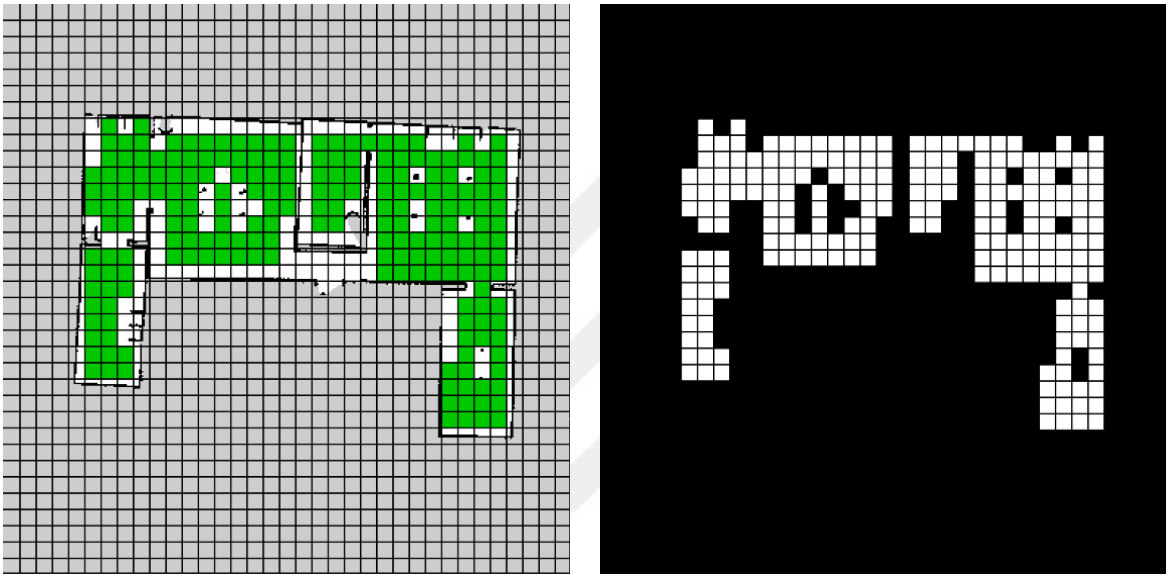
Gerçek boyutlara göre oluşturulan grid harita görüntüsünün okunurluğunu arttırmak amacıyla robotun boyutu 1x4 oranında büyütülmüş olarak oluşturulan haritası Şekil 3.11'de verilmiştir.



Şekil 3.11 Robotun boyutunun 1x4 oranında büyütüldüğü grid harita

Çalışma kapsamında işlemler gerçek boyutlarda yapılmıştır. Şekil 3.11'deki görüntü görselleştirme amacıyla kullanılmıştır ve gerçek boyutları ifade etmemektedir.

Grid harita işleminden sonra haritanın doluluk bilgisi hesaplanmıştır. Hesaplama yapılırken her grid'in içindeki pikseller okunmuş ve engel içermeyen grid'ler Şekil 3.12(a)'da görüldüğü gibi yeşile boyanarak görselleştirilmiştir. Yeşile boyanan alanlar gezinilebilir alanı oluşturmaktadır. Gezinilebilir alanın daha net okunabilmesi için alanyazında yaygın olarak kullanılan monokrom gösterim yapılmıştır. Buna göre; haritada engeller siyah, gezinilebilir alan ise beyaz renkte Şekil 3.12(b)'de görselleştirilmiştir.



(a) (b)

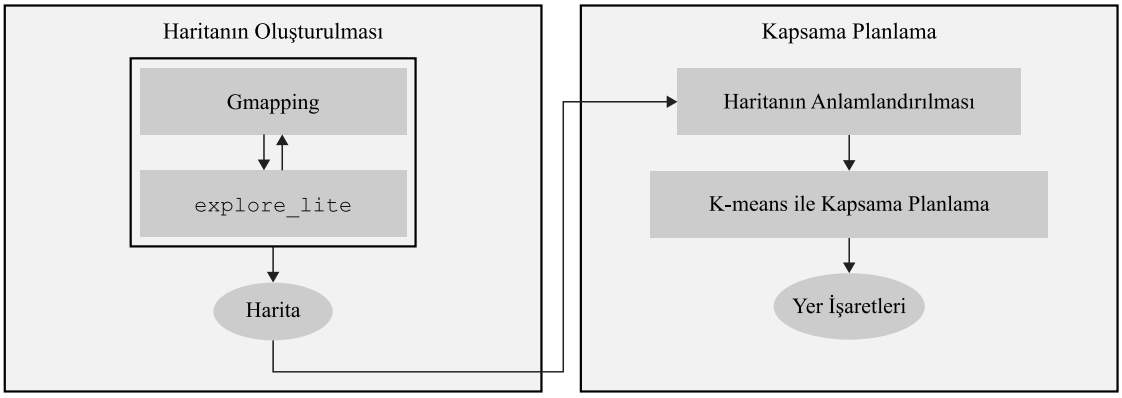
Şekil 3.12 (a) Harita üzerindeki gezinilebilir alanın yeşil; (b) monokrom gösterimle engellerin siyah, gezinilebilir alanın ise beyaz renk ile görselleştirilmesi

Şekil 3.12'de görüldüğü gibi robotun ölçüleri büyüdükçe kısmen dolu olan hücrenin tamamen dolu olarak işlenme ihtimali de artmaktadır. Aynı zamanda, GTTKP yöntemi ile hesaplanan haritanın 3 bölüme ayrıldığı görülmektedir. Bu bölümler arasında robotun geçebileceği kadar mesafe bulunmasına karşın olası bir gezinme senaryosunda robot hangi bölümden başlatılırsa yalnızca o bölümde kalan grid'leri gezinecektir.

3.5. K-means++ Tam Kapsama Planlama (Km++TKP)

Bu çalışmada, bir mobil robotun gezineceği iç mekânın haritasındaki gezinilebilir alanın K-means++ algoritması kullanılarak kümelendiği ve ağırlık merkezlerinin yer işareti olarak kullanılabileceği bir çevrim dışı TKP yöntemi olan Km++TKP yöntemi önerilmiştir.

Km++TKP yönteminin uygulanması; iç mekânın haritasının oluşturulması ve kapsamanın planlanması olmak üzere iki aşamadan oluşmaktadır (Şekil 3.13).



Şekil 3.13 Km++TKP yöntemi uygulama aşamaları blok diyagramı

Şekil 3.13'te verilen blok diyagramına göre öncelikle haritanın oluşturulması aşamasında Gmapping algoritması ve `explore_lite` paketi kullanılarak iç mekânın haritası oluşturulur. Bir sonraki aşama olan kapsamanın planlanması, haritada gezinilebilir alanın belirlenmesi ve planlaması sürecidir. Burada gezinilebilir alandaki pikseller K-means++ algoritması ile kümelenir ve hepsalanan ağırlık merkezleri gezinme için yer işaretlerine dönüştürülür. Oluşturulan yer işaretleri navigasyon yığınına gönderilerek robotun planlanan kapsamaya göre gezinmesi sağlanabilir.

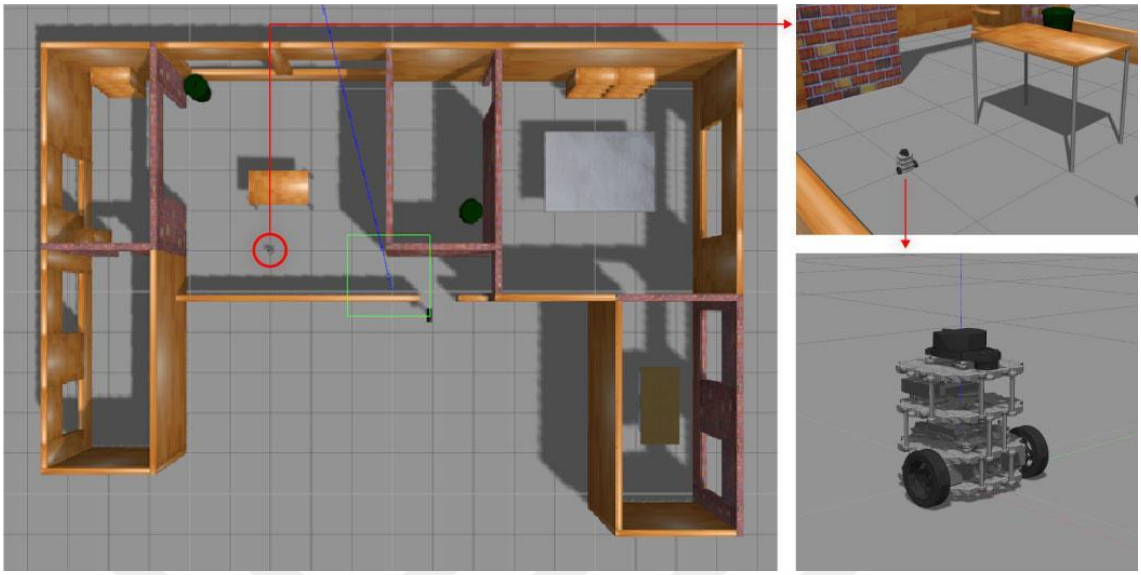
Km++TKP yönteminin uygulamasında Ubuntu Linux dağıtımının 18.04 (Bionic Beaver) sürümüne yönelik olarak yayınlanan ROS'un Melodic Morenia sürümü kullanılmıştır. Buna ek olarak Gazebo robot simülasyon ortamı ve RViz (ROS Visualization - ROS Görselleştirme) görselleştirme araçları da sisteme dahil edilmiştir. Yazılım geliştirme sürecinde ise Python programlama dili kullanılmıştır. Takip eden alt bölümlerde, Km++TKP yönteminin gerçekleştirimi hakkında bilgiler yer almaktadır.

3.5.1. Haritanın Oluşturulması

Bilinmeyen bir iç mekâna ve bu mekânda bilinmeyen bir noktaya bırakılan robot tarafından otonom olarak harita oluşturulması için ROS `explore_lite` paketi ile Gmapping algoritması kullanılmıştır.

Çalışmada TurtleBot3 Burger robot kullanılmıştır. Robotun simülasyon ortamında gezinme mekânı olarak TurtleBot3 House adlı bir iç mekân seçilmiştir. Bu simülasyon mekânı, karmaşık görevlerin performans testlerinin yapılması için geliştirici firma tarafından

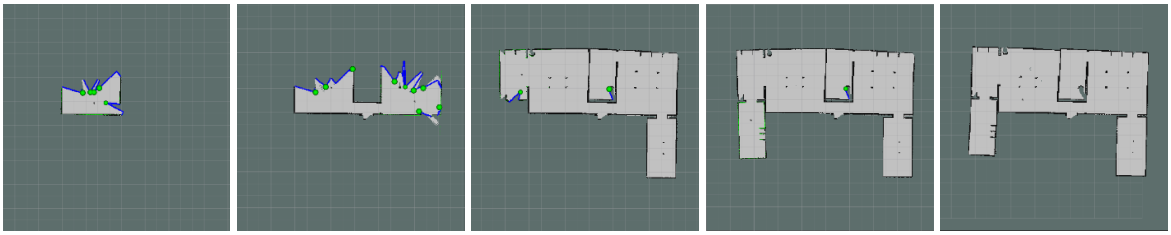
hazırlanmış ve Gazebo için ROS paketi haline getirilerek yayınlanmıştır. Mekânın ve robotun Gazebo robot simülasyon ortamındaki görünümü Şekil 3.14'te gösterilmiştir.



Şekil 3.14 TurtleBot3 House adlı mekânın ve robotun, Gazebo robot simülasyon ortamındaki görünümleri

Şekil 3.14'te robotun iç mekândaki bulunduğu noktayı vurgulamak için robot, kırmızı daire içerisine alınmıştır.

ROS `explore_lite` paketi ile Gmapping algoritması kullanılarak iç mekânın adım adım haritasının oluşturulması gerçekleştirimin gösterimi RViz ile görselleştirilmiştir. RViz'den eşit zaman aralıklarıyla alınan gerçekleştirim görüntüleri Şekil 3.15'te verilmiştir.



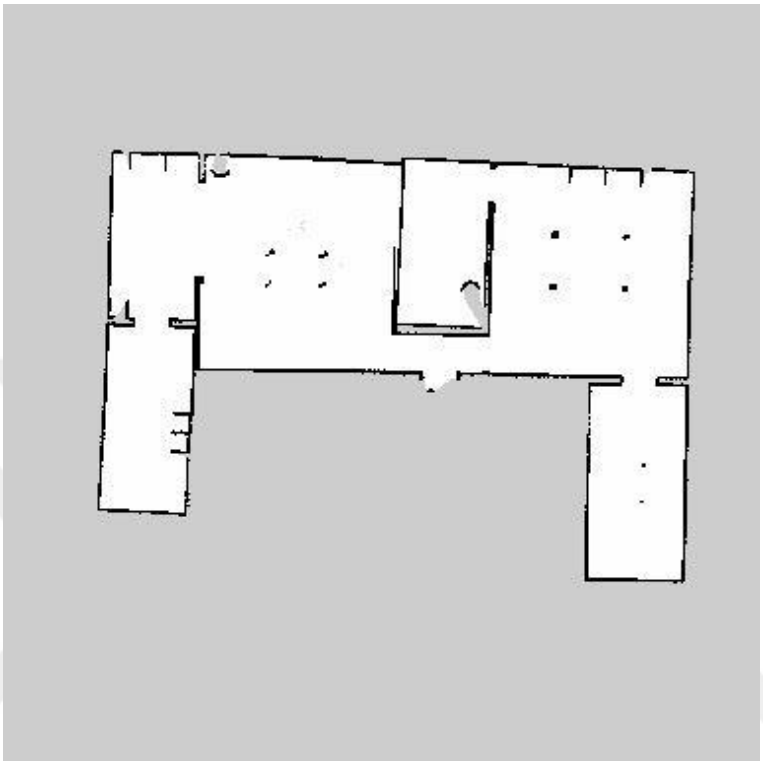
(a) (b) (c) (d) (e)

Şekil 3.15 Mekânın haritasının otonom gezinilerek adım adım oluşturulması

Şekil 3.15'te verilen harita üzerinde yer alan küre biçimindeki yeşil noktalar ROS `explore_lite` paketi tarafından oluşturulan, robot üzerinde bulunan LIDAR'ın ulaştığı son nokta verisini temsil eden sınır noktalarıdır. Robotun başlangıç pozisyonundayken belirlediği sınır noktaları (Şekil 3.15(a)), gezinmeye başlamasıyla sınır noktalarının iç mekânda yayılması (Şekil 3.15(b)), farklı bölgelerin oluşması (Şekil 3.15(c)), son kalan tek

sınır noktası (Şekil 3.15(d)) ve robotun son kalan sınır noktasına giderek mekânın haritasını oluşturma işlemini tamamlaması Şekil 3.15(e)'de görülmektedir.

Tamamlanan harita, ROS `map_server` paketi kullanılarak kaydedilir. Haritanın kaydedilmesiyle oluşan resim dosyası Şekil 3.16'da verilmiştir.



Şekil 3.16 TurtleBot3 House simülasyon ortamının Gmapping algoritması ile oluşturulan haritası

Şekil 3.16'da yer alan haritada ROS `map_server` paketi ile PGM formatında kaydedilen görselde keşfedilmeyen alanlar gri, engellerin sınırı siyah ve robot için gezinilebilir alan beyaz renkte ifade edilir (Şekil 3.16).

Çalışma kapsamında robotun haritalamaya başlamadan önce bulunduğu koordinatlar `odom` yayını aracılığıyla poz bilgisi olarak kaydedilmiştir. Bu bilgi, robotun harita oluşturma işlemini tamamladıktan sonra başladığı noktaya dönmesi için kullanılmıştır. Daha sonra sınır noktası sayısı okunmuş ve sınır noktası sayısı 0 olduğunda robot navigasyon yığını ile başlangıç pozisyonuna gönderilmiştir. Bu şekilde robotun haritadaki konum bilgisi tutulmuş olunur ve haritanın oluşturulması aşaması tamamlanır.

3.5.2. K-means ve K-means++ Algoritması

En iyi bilinen bir kümeleme algoritması olan K-means algoritması J. MacQueen tarafından 1967 yılında alanyazına kazandırılmıştır (MacQueen, 1967). K-means algoritması ilk olarak *k* adet noktayı rastgele seçer. Bu noktalar, ilk belirlenen kümelerin merkezlerini temsil eder. Daha sonra her nokta, en yakın olduğu merkez noktanın bulunduğu kümeye dahil edilir. Son olarak oluşan kümelerin elemanlarının ağırlıklı ortalamaları hesaplanarak yeni küme merkezleri belirlenir. Belirlenen merkezlerin değeri daha sonraki kümeleme işlemlerinde, bulunduğu kümeyi temsil eder. K-means algoritmasının uygulama adımları aşağıdaki şekilde özetlenebilir:

Adım 1. İlk küme merkezlerini belirlemek için rastgele *k* adet merkez seçilir.

Adım 2. Her noktanın seçilen merkezlere uzaklığı hesaplanır. Elde edilen sonuçlara göre tüm noktalar *k* kümeden kendilerine en yakın olan kümeye dahil edilir.

Adım 3. Oluşan kümelerin yeni merkezleri, kümedeki tüm kayıtların ortalaması alınarak yeniden hesaplanır.

Adım 4. Küme merkezleri değişmeyene kadar Adım 2 ve Adım 3 tekrarlanır.

Kümeleme algoritmaları için başlangıçta küme merkezlerinin seçimi algoritmanın performansını etkileyen kritik bir işlemdir. Bu işlem için; Maximin, Katsavounidis ve K-means++ gibi çeşitli yöntemler geliştirilmiştir (Çelebi ve Kingravi, 2015). Üstünel (2018) çalışmasında, K-means algoritmasıyla birlikte farklı metotlar kullanarak verileri; doğru, etkin ve hızlı bir şekilde kümelemiş ve sonuçları analiz etmeyi amaçlamıştır. Çalışmanın sonunda, K-Means++ yönteminin diğer yöntemlere göre hata, toplam süre ve iterasyon anlamında daha başarılı sonuçlar verdiği bulgusuna ulaşmıştır. Bu bağlamda çalışmada, Arthur ve Vassilvitskii (2007) tarafından geliştirilen K-means++ algoritması kullanılmıştır. K-means++ algoritmasının tercih edilmesinde, yaygın olarak kullanılan bir kümeleme algoritması olması ve daha önce kapsama planlama için kullanılmamış olması etkili olmuştur. K-means algoritmasının ilk uygulama adımında da görüldüğü gibi algoritmanın orijinalinde küme merkezleri rastgele belirlenir. K-means++ yönteminde ise sadece birinci merkez rastgele belirlenir. Geri kalan (*k* - 1) tüm merkezler birinci merkez dikkate alınarak ve olasılık hesaplamalarından faydalanılarak hesaplanır. Yöntemin uygulama adımları aşağıda verilmiştir.

Adım 1. Noktalar arasından rastgele bir ilk merkez c_1 seçilir.

Adım 2. Geri kalan k - 1 merkez Eşitlik 3 kullanılarak olasılık hesaplamayla seçilir.

$$\frac{D(\hat{x})^2}{\sum_{x \in X} D(x)^2} \tag{3}$$

Eşitlik 3'te $D(x)$, bir $x \in X$ noktasının en yakın ağırlık merkezine (küme merkezi) olan uzaklığını göstermek üzere ve c_i $(1 \leq i \leq k - 1)$ olmak üzere $\hat{x} \in X/\{c_i\}$ için $c_i = \hat{x}$ şeklinde seçilir.

Adım 3. Adım 2, k tane merkez oluşana kadar tekrarlanır.

K-means++ algoritmasında uzaklık hesaplaması için Öklid uzaklık ölçüsü kullanılır. Bu uzaklık ölçüsü alanyazında en yaygın kullanılan ölçülerden biridir. Aynı zamanda Öklid uzaklık ölçüsü boyutları benzer ölçeklere sahip verilerde kullanılmak için uygundur (Singh, Yadav ve Rana, 2013). İki nokta arasındaki Öklid mesafesi, karşılık gelen değerler arasındaki farkların karelerinin toplamının karekökünün hesaplanmasıyla bulunur. Öklid uzaklık ölçüsü formülü Eşitlik 4'te verilmiştir.

$$d_{ij} = \sqrt{\sum_{k=1}^{m} \left(x_{ik} - x_{jk}\right)^2} \tag{4}$$

Eşitlik 4'te m boyutlu gözlem uzayında i ve j veri göstergesi uzaklıkları kareleri toplamının karakökü alınarak hesaplanır.

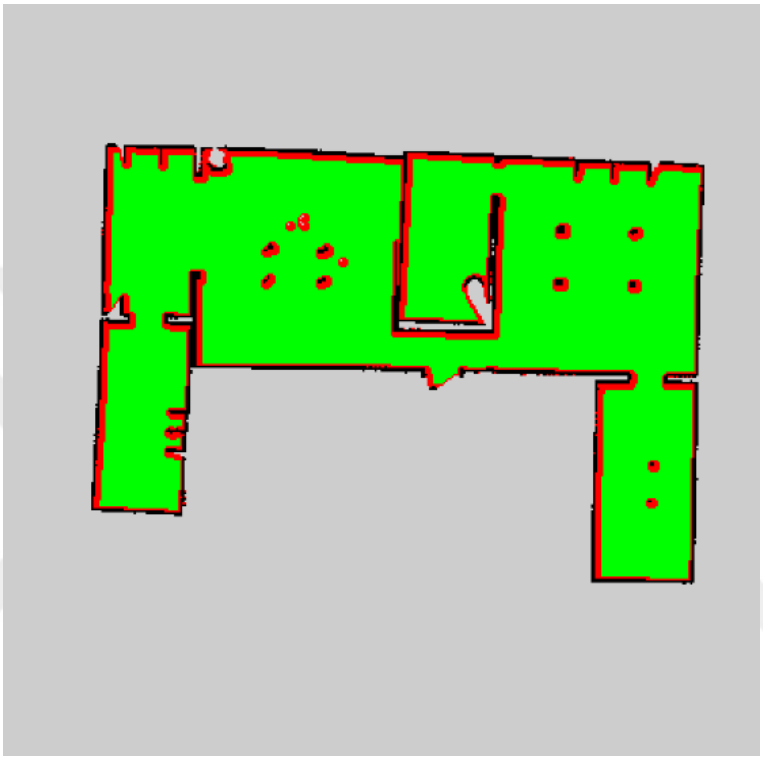
3.5.3. Kapsamanın Planlanması

Km++TKP yönteminde kapsama, iç mekânın haritasındaki gezinilebilir alandaki piksellerin K-means++ algoritması kullanılarak kümelenmesi ve ağırlık merkezlerinin hesaplanmasıyla sağlanır. Hesaplama işlemi için öncelikle gezinilebilir alan belirlenmelidir. Bunun için, çalışma kapsamında geliştirilen K-means++ Kapsama (Km++K) algoritması ile harita anlamlandırılarak gezinilebilir alanın ölçüsü hesaplanmıştır. Ardından K-means++ algoritması ile pikseller kümelenerek ağırlık merkezleri hesaplanmıştır.

3.5.3.1. Haritanın Anlamlandırılması

Km++K algoritması ile iç mekânın haritasındaki her piksel değeri okunarak sayı bilgisi tutulmuştur. Daha sonra engel sınırını ifade eden siyah piksellerin çevresinde robot için güvensiz alanlar belirlenmiştir. Güvensiz alanlar, gezinilebilirdir. Buna karşın engele

yakın bölgeler olduklarından robotun çarpma ve takılma riski vardır. Çalışma kapsamında bu riski en aza indirmek için gezinilebilir alan ve güvensiz alanlar ayrımına gidilmiştir. Gezinilebilir alan ve güveniz alanlar arasındaki ilişkinin anlaşılması için önce haritadaki gezinilebilir alandaki pikseller yeşil renge, güvensiz alanlar kırmızı renge boyanarak görselleştirilmiştir. Bu uygulama (Km++K algoritması) sonucunda Şekil 3.17'de verilen harita görüntüsü elde edilmiştir.



Şekil 3.17 İç mekândaki güvensiz alanların kırmızı, gezinilebilir alanın yeşil renk ile harita üzerinde gösterilmesi

Şekil 3.17'de verilen harita üzerindeki gri pikseller robotun hakkında hiç bilgisi olmadığı keşfedilmemiş alanları, siyah pikseller engel sınırını, kırmızı pikseller güvensiz alanları ve yeşil pikseller gezinilebilir alanı göstermektedir.

İlgili alanlar sınıflandırıldıktan sonra keşfedilmemiş, engel sınırı, güvensiz alanlar ve gezinilebilir alanın ölçüleri hesaplanmıştır. Ölçüler hesaplanırken, Gmapping algoritmasının 1 pikseli *5cm x 5cm* boyutlarında oluşturduğu bilgisi doğrultusunda her pikselin 25 cm^2 olduğu değeri kullanılmıştır.

Çalışma kapsamında haritanın anlamlandırılması için geliştirilen Km++K algoritmasının sözde kodu Çizelge 3.2'de verilmiştir.

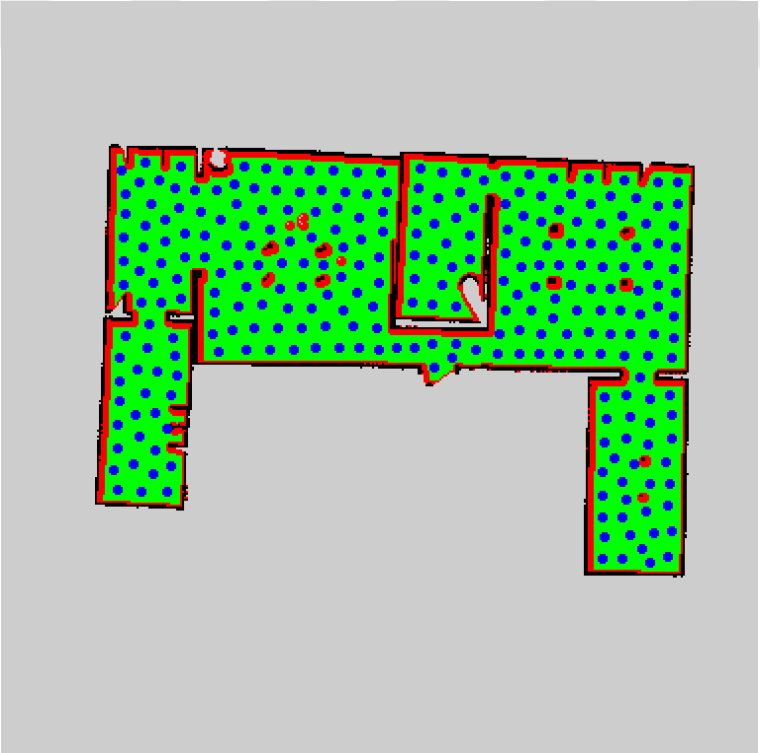
Çizelge 3.2 Çalışma kapsamında geliştirilen Km++K algoritmasının sözde kodu

KmK Algoritması	
	Girdi: Harita (PGM), Robot Kapsama Alanı ve Güvensiz Alan Öteleme Uzaklığı
	Çıktı: Alan (Keşfedilmemiş, Engel Sınırı, Güvensiz ve Gezinilebilir) Ölçüleri, Gezinebilir Alandaki Piksellerin Koordinatları ve *k* değeri.
1:	INPUT Harita
2:	INPUT robotKapsama
3:	INPUT guvensizAlanOtele
4:	FOR i (0 → HaritaGenisligi)
5:	FOR j (0 → HaritaYuksekligi)
6:	IF Harita[i , j] == Beyaz
7:	pikselTamam = True
8:	FOR α (0 → 360)
9:	xOtele = guvensizAlanOtele * Cosinus(Radyan(α))
10:	yOtele = guvensizAlanOtele * Sinus(Radyan(α))
11:	FOR iOtele (i → xOtele)
12:	FOR jOtele (j → yOtele)
13:	IF Harita[iOtele , jOtele] != Beyaz
14:	pikselTamam = False
15:	FOR iOtele (i → xOtele)
16:	FOR jOtele (yOtele → j)
17:	IF Harita[iOtele , jOtele] != Beyaz
18:	pikselTamam = False
19:	FOR iOtele (xOtele → i)
20:	FOR jOtele (j → yOtele)
21:	IF Harita[iOtele , jOtele] != Beyaz
22:	pikselTamam = False
23:	FOR iOtele (xOtele → i)
24:	FOR jOtele (yOtele → j)
25:	IF Harita[iOtele , jOtele] != Beyaz
26:	pikselTamam = False
27:	IF pikselTamam == True
28:	gezinilebilirPikseller[] = Harita[i, j]
29:	ELSE
30:	guvensizPikseller[] = Harita[i, j]
31:	ELSE IF Harita[i , j] == Siyah
32:	engelliPikseller[] = Harita[i, j]
33:	ELSE
34:	kesfedilmemisPikseller[] = Harita[i, j]
35:	k = (LEN(gezinilebilirPikseller) * 5^2 / 10000) / (robotKapsama)

Çizelge 3.2'de paylaşılan sözde kodda öncelikle harita bilgisi, robotun tek seferde kapsama yaptığı alan ölçüsü ve güvensiz alan öteleme uzaklığı girdi olarak alınır. Daha sonra haritadaki her pikselin beyaz, siyah ve gri renkte olup olmama durumu sınanır ve pikseller ilgili renk adında oluşturulan bir diziye atanır. Bu işlem gerçekleştirilirken güvensiz alan ötelemesi için sadece beyaz piksellerin çevresindeki pikseller taranır. Bu sayede güvensiz alanlar oluşturulur. Son olarak, gezinilebilir yani beyaz renkte olan piksellerin sayısı kullanılarak gezinilebilir toplam alan robotun tek seferde kapsadığı alana bölünerek *k* değeri elde edilir.

3.5.3.2. K-means++ Algoritması ile Kümeleme ve Ağırlık Merkezlerinin Oluşturulması

Çalışma kapsamında K-means++ algoritması için Sklearn Python kütüphanesinin Cluster sınıfı kullanılmıştır. K-means++ algoritmasının başlangıç parametresi olan *k* değeri, gezinilebilir alanın ölçüsünün robotun tek seferde kapsadığı alanın ölçüsüne bölünmesiyle hesaplanmıştır. Gezinilebilir alanı ifade eden yeşil pikseller, K-means++ algoritması ile *k* adet kümeye ayrılmış ve ağırlık merkezleri oluşturulmuştur. Oluşan ağırlık merkezleri mavi daire biçiminde Şekil 3.18'de harita üzerinde gösterilmiştir.



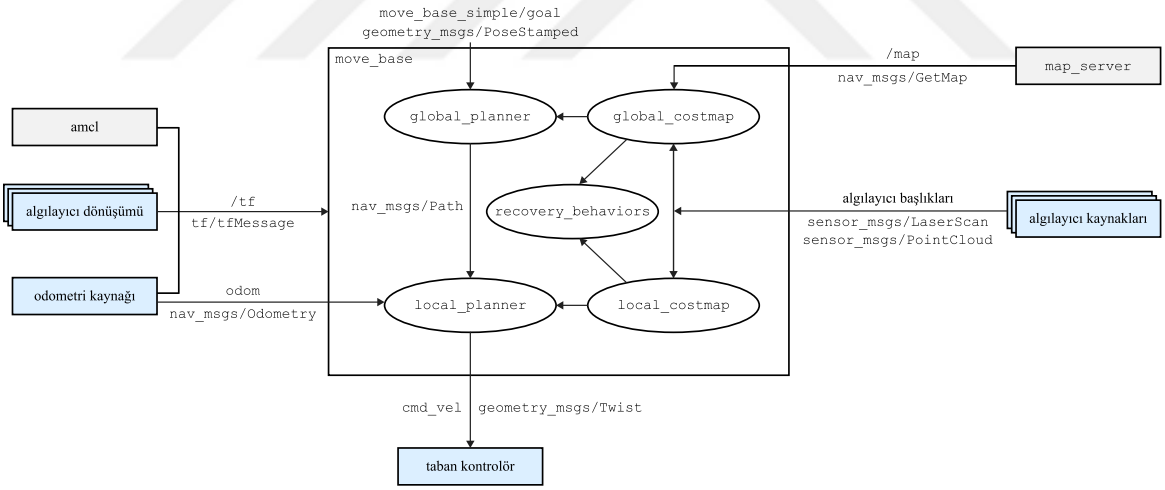
Şekil 3.18 Gezinilebilir alandaki piksellerin K-means++ algoritması ile kümelenmesi ve ağırlık merkezlerinin oluşturularak mavi daire biçiminde görselleştirilmesi

Özet olarak mavi daireler, şekilde görülen yeşil alandaki tüm piksellerin K-means++ algoritması ile kümelenmiş *k* ağırlık merkezini ifade eder. Bu işlem aslında centroid olarak adlandırılan ağırlık merkezi hesaplama yöntemidir. Buna göre, haritadaki mavi noktalar robotun tam kapsama yaparken gezineceği muhtemel rotanın bir parçasıdır. Bu bağlamda oluşturulan ağırlık merkezleri, robot gezinmesi sürecinde yer işareti olarak ifade edilir.

Km++TKP yönteminde gezinme işlemi yer işaretleri arasında gerçekleşir. Yer işaretleri arasındaki hareket ROS `turtlebot3_navigation` adlı paket ile navigasyon yığını kullanılarak sağlanır. Öncelikle kapsamanın planlanması aşamasında oluşturulan yer işaretleri robotun gezinmesi için robota bu aşamada gönderilir. Robotun iç mekânı kapsaması, yer işaretlerini gezinmesi ile gerçekleşir.

3.6. Gezinme

Navigasyon yığını, kavramsal düzeyde ele alındığında odometri ve algılayıcı düğümlerinden bilgi alır ve robotu harekete geçirmek için motorlara hız komutu gönderir. Şekil 3.19'da robotun iki yer işareti arasındaki hareketinin gerçeklemesi için kullanılan navigasyon yığınının diyagramı verilmiştir.



Şekil 3.19 ROS navigasyon yığını şeması

Şekil 3.19'da verilen diyagramda yer alan beyaz bileşenler zorunlu bileşenlerdir, gri bileşenler isteğe bağlı bileşenlerdir ve her robot platformu için mavi bileşenlerin oluşturulması gerekir. Navigasyon yığını için robotun tf kullanarak koordinat noktaları arasındaki ilişkiler hakkında bilgi yayınlanır. Buna ek olarak ortamdaki engellerden kaçınmak için algılayıcılardan gelen bilgiler kullanılır. Bunun için ROS üzerinden `sensor_msgs/LaserScan` veya `sensor_msgs/PointCloud` mesajları yayınlanır.

Odometri bilgileri ise `tf` ve `nav_msgs/Odometry` mesajı kullanılarak yayınlanır. Robotun hareket etmesini sağlayan motorlara, `cmd_vel` başlığındaki `geometry_msgs/Twist` mesajı kullanılarak hız komutları gönderilir. Son olarak, navigasyon yığını için harita bilgisi zorunlu değildir ancak çalışma kapsamında iç mekânın haritası kullanılarak navigasyon yığını çalıştırılır. Navigasyon yığınına harita bilgisi üzerindeki yer işaretlerinin koordinatları sıra ile gönderilerek robotun ilgili koordinatlara erişmesi sağlanır. Bu esnada robot Dinamik Pencere Yaklaşımı (DWA - Dynamic Window Approach) aracılığıyla engellerden de kaçınabilir. DWA, engellerden kaçınma planlaması ve engellerden kaçınma için popüler bir yöntemdir (Pyo, Cho, Jung ve Lim, 2017). Navigasyon yığını kullanımı için ön koşul olarak; robotta ROS kullanılması, robotun `tf` dönüşüm ağacına sahip olması ve algılayıcı verilerini yayınlaması gerekir. Bu gereksinimleri araştırmada kullanılan Gazebo robot simülasyonu ve TurtleBot3 Burger robot karşılamaktadır.

4. DENEYLER VE BULGULAR

Çalışma kapsamında önerilen Km++TKP yönteminin kapsama başarımını test etmek için iki deney ortamı hazırlanmıştır. Bu ortamlardan ilki Gazebo'da bir simülasyon ortamıdır. TurtleBot3 House adlı standart iç mekân kullanılmıştır. Simülasyon ortamındaki deneyler, 3.07 GHz Intel Core i3 işlemci ve 8 GB RAM'a sahip bilgisayarda gerçekleştirilmiştir. İkincisi araştırma sırasında hazırlanan gerçek iç mekândır. Gerçek iç mekân iki farklı şekilde kullanılmıştır. Birincisinde ortam boştur, sadece duvarlardan oluşur; diğerinde ortamda duvarlarla birlikte engeller vardır. Gerçek dünyadaki deneyler, dört çekirdekli 1.2 GHz Broadcom BCM2837 64bit işlemci ve 1 GB RAM'a sahip Raspberry Pi 3 Model B donanımlı robotta gerçekleştirilmiştir. Deneyler; GTTKP yöntemi ve Km++TKP yönteminin TurtleBot3 Burger robot ile simülasyon ortamında gerçekleşen deneyler kendi içinde, gerçek dünyada gerçekleştirilen deneyler kendi içinde başarımlarının karşılaştırılması şeklindedir. Buna göre, aşağıdaki deneyler planlanmıştır.

1. TurtleBot3 House simülasyon iç mekânında GTTKP yöntemi başarım testi.
2. TurtleBot3 House simülasyon iç mekânında Km++TKP yöntemi başarım testi.
3. Gerçek dünya boş iç mekânda GTTKP yöntemi başarım testi.
4. Gerçek dünya boş iç mekânda Km++TKP yöntemi başarım testi.
5. Gerçek dünya engel içeren iç mekânda GTTKP yöntemi başarım testi.
6. Gerçek dünya engel içeren iç mekânda Km++TKP yöntemi başarım testi.

Deneylerde başarımının ölçülmesinde iç mekânın kapsama oranı ve kapsama planı hazırlama süresi parametreleri kullanılmıştır. İç mekânın kapsama oranı verisini deney 1, deney 3 ve deney 5'te doğrudan grid ölçüleri alınarak; deney 2, deney 4 ve deney 6'da piksel ölçüleri oluşturmaktadır. Kapsama oranı, gezinilebilir alandaki kapsanan piksel sayısının yine gezinilebilir alandaki tüm piksel sayısına bölünmesiyle elde edilir (Eşitlik 4).

Kapsama oranı, bütün deneylerde hem GTTKP yöntemi hem de Km++TKP yöntemi için ayrı ayrı ölçülmüştür.

Kapsama planı hazırlama başarım süresi, bütün deneylerde hem GTTKP yöntemi hem de Km++TKP yöntemi için ayrı ayrı ölçülmüştür.

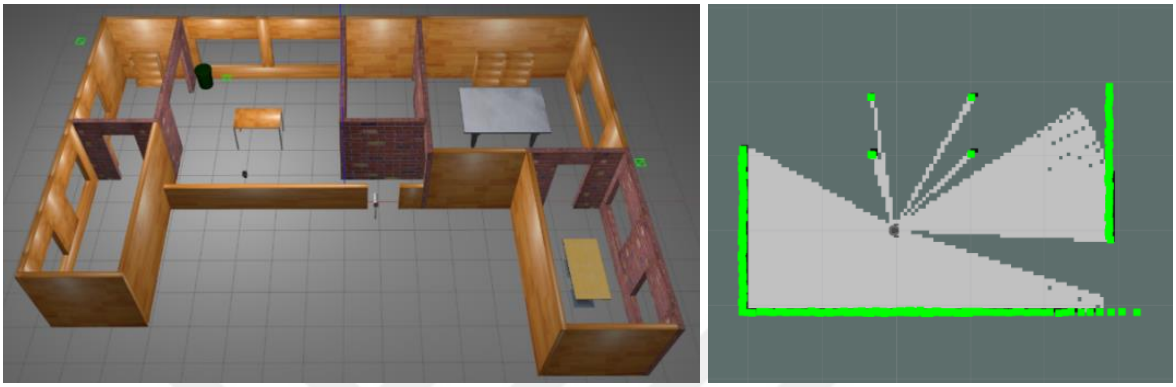
Sayıltılar:

Deneyler sırasında robotun pil seviyesinin ve LIDAR verisinin eşit olduğu varsayılmıştır.

Takip eden başlıklarda, varolan ve önerilen yöntemin deneysel çalışmaları, sonuçları ve karşılaştırmaları verilmiştir.

4.1.Simülasyon Deneyleri

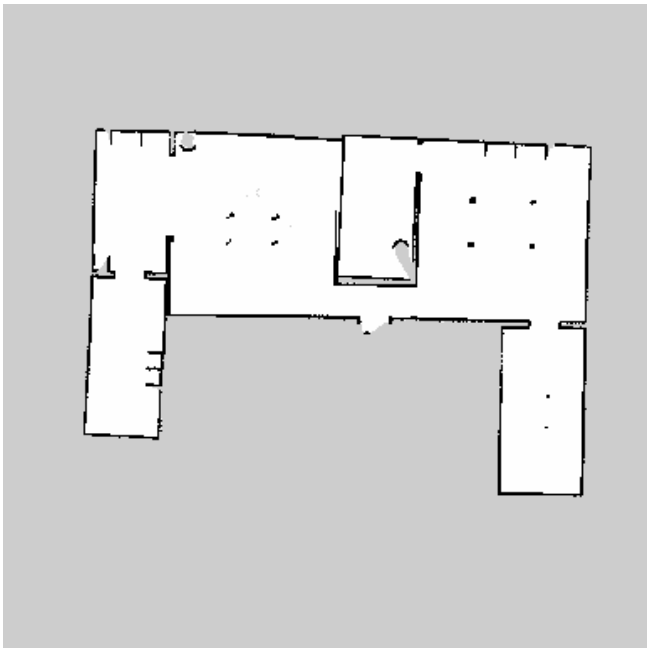
İlk olarak çalışmanın yapıldığı bilgisayarda Gazebo House adlı similasyon ortamı açılmıştır (Şekil 4.1).



(a) (b)

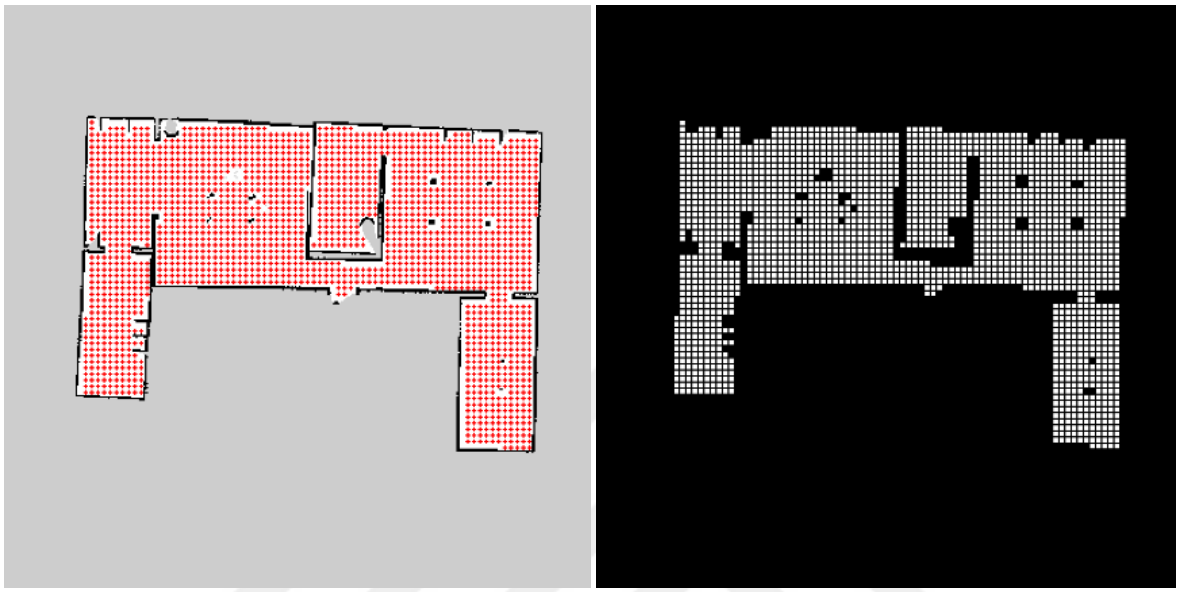
Şekil 4.1 (a) Gazebo House adlı simülasyon ortamı ve (b) robotun başlangıç konumundaki LIDAR algılayıcı verileri

Daha sonra robot haritalama yapmak üzere gezinmeye başlamıştır. Haritalama tamamlandığında ortamın haritası kaydedilmiştir (Şekil 4.2) ve robot tekrar başlangıç konumuna dönmüştür.



Şekil 4.2 Gazebo House adlı simülasyon ortamının oluşturulan haritası

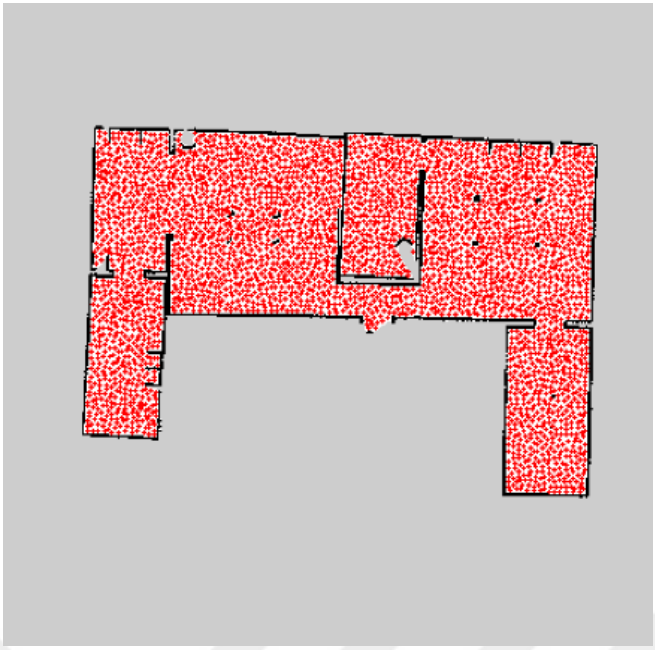
Bu uygulamadan sonra elde edilen harita (Şekil 4.2) GTTKP yöntemi ve araştırma kapsamında önerilen yöntem (Km++TKP) ile ayrı ayrı işlenmiştir. Haritaya GTTKP yöntemi uygulandığında robotun gezineceği 2.113 hücre merkezi hesaplanmıştır. Bu merkezler, Şekil 9(a)'da kırmızı noktalar şeklinde görselleştirilmiştir. Haritanın monokrom gösterimi Şekil 9(b)'de verilmiştir.



(a) (b)

Şekil 4.3 (a) Gazebo House adlı simülasyon ortamında GTTKP yöntemi ile oluşturulan ağırlık merkezleri ve (b) haritanın monokrom gösterimi

Haritaya araştırma kapsamında önerilen Km++TKP yöntemi uygulandığında gezinilebilir alanın ölçüsünün robotun tek seferde kapsadığı alanın ölçüsüne bölünmesiyle *k* değeri 3.708 hesaplanmıştır. Kümeleme işlemi sonunda elde edilen ağırlık merkezleri Şekil 4.4'te görselleştirilmiştir.



Şekil 4.4 Gazebo House adlı simülasyon ortamında önerilen Km++TKP yöntemiyle oluşturulan ağırlık merkezleri

Buna göre; iki yöntemin de kapsama planı hesaplama süresi ve kapsama oranları Çizelge 4.1'de verilmiştir.

Çizelge 4.1 Simülasyon ortamında iki yöntemin kapsama planı hesaplama süresi ve kapsama ölçülerinin karşılaştırılması

	GTTKP	**Km++TKP**
Hücre /Ağırlık Merkezi Sayısı:	2.113	3.708
İşlem Süresi:	1,0016 sn	615,4828 sn
Kapsanan Alan:	84 m^2	90,65 m^2
Kapsanamayan Alan:	9,58 m^2	2,93 m^2

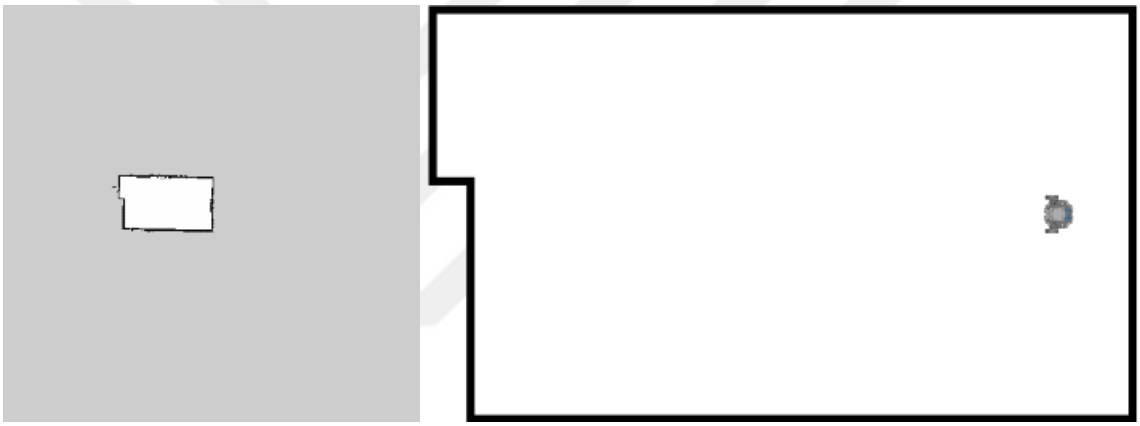
Çizelge 4.1'de yer alan veriler incelendiğinde GTTKP yöntemi iç mekânın %89,76'sını kapsarken, önerilen Km++TKP yönteminin iç mekânın %96,86'sını kapsadığı görülmektedir. Bu hesaplama bilgisayarın işlemcisinde gerçekleştirilmiştir. Hesaplama süresi önerilen yöntem için daha uzundur ancak bu işlem bir kez gerçekleştirilecektir.

4.2. Gerçek Dünya Deneyleri

İlk olarak çalışmanın yapılacağı deney alanı, Kahramanmaraş Sütçü İmam Üniversitesi Robotik Laboratuvarının girişine kurulmuştur (Şekil 4.5). Bu alan 9,7 m^2'dir.

Şekil 4.5 Gerçek dünya boş iç mekân deney alanı

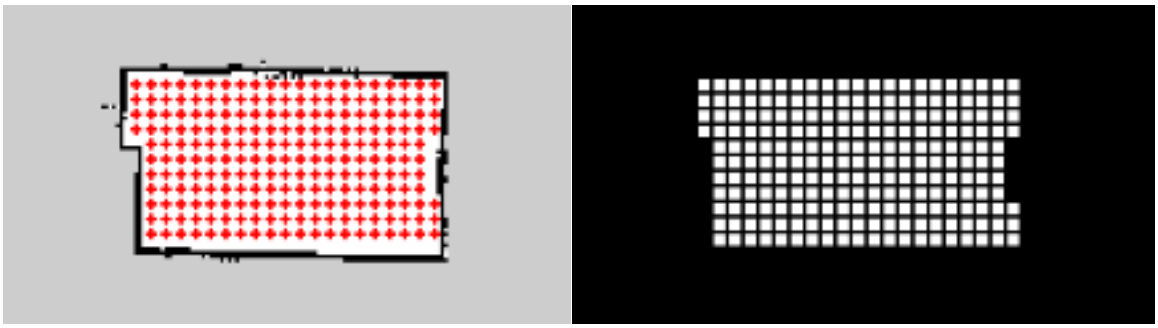
Daha sonra robot bu ortam boşken yani engel içermiyorken haritalama yapmak üzere gezinmeye başlamıştır. Haritalama tamamlandığında ortamın haritası kaydedilmiştir (Şekil 4.6(a)) ve robot tekrar başlangıç konumuna dönmüştür. Çevrenin üstten görünümü ve robotun başlangıç noktası Şekil 4.6(b)'de görselleştirilmiştir.



(a) (b)

Şekil 4.6 (a) Gerçek dünya boş iç mekânda oluşturulan harita; ve (b) iç mekânın üstten görünümü

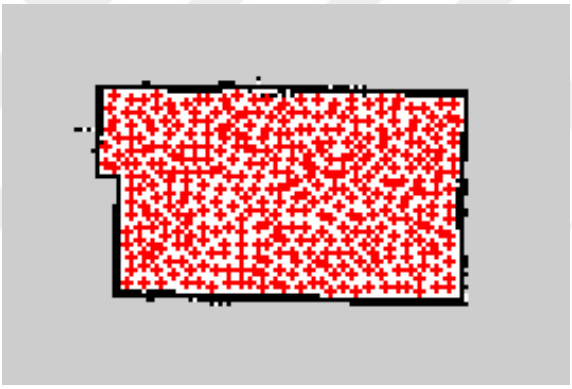
Bu uygulamadan sonra elde edilen harita (Şekil 4.7(a)), GTTKP ve araştırma kapsamında önerilen yöntem (Km++TKP) ile ayrı ayrı işlenmiştir. Haritaya GTTKP yöntemi uygulandığında robotun gezineceği 220 hücre merkezi belirlenmiştir. Bu merkezler, Şekil 4.7(a)'da kırmızı noktalar şeklinde görselleştirilmiştir. Haritanın monokrom gösterimi Şekil 4.7(b)'de verilmiştir.



(a) (b)

Şekil 4.7 (a) Gerçek dünya boş iç mekânda GTTKP yöntemiyle oluşturulan ağırlık merkezleri ve (b) haritanın monokrom gösterimi

Haritaya araştırma kapsamında önerilen Km++TKP yöntemi uygulandığında gezinilebilir alanın ölçüsünün robotun tek seferde kapsadığı alanın ölçüsüne bölünmesiyle *k* değeri 381 hesaplanmıştır. Kümeleme işlemi sonunda elde edilen ağırlık merkezleri Şekil 4.8'de görselleştirilmiştir.



Şekil 4.8 Gerçek dünya boş iç mekânda önerilen Km++TKP yöntemiyle oluşturulan ağırlık merkezleri

Buna göre; iki yöntemin de kapsama planı hesaplama süresi ve kapsama oranları Çizelge 4.2'de verilmiştir.

Çizelge 4.2 Gerçek dünya boş iç mekânda iki yöntemin kapsama planı hesaplama süresi ve kapsama ölçülerinin karşılaştırılması

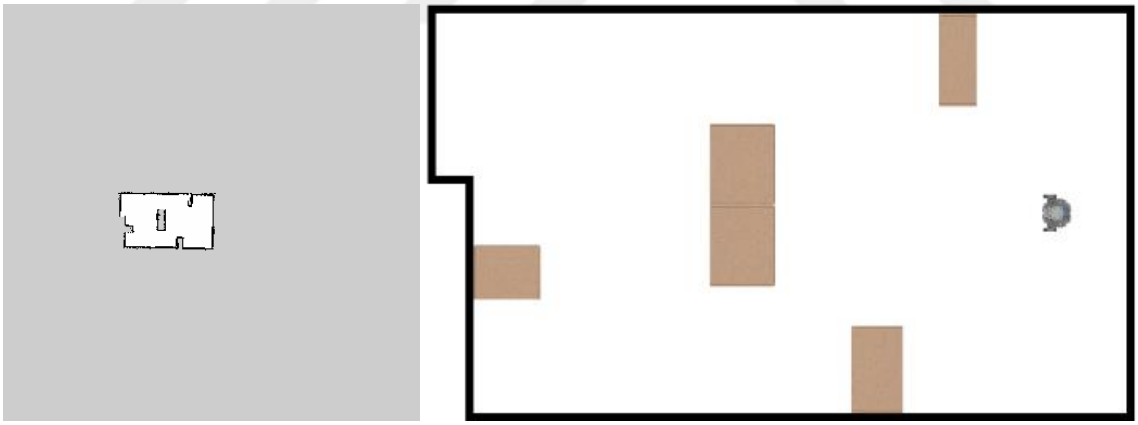
	GTTKP	**Km++TKP**
Hücre /Ağırlık Merkezi Sayısı:	220	381
İşlem Süresi:	6,1838 sn	40,8474 sn
Kapsanan Alan:	8 m^2	9,31 m^2
Kapsanamayan Alan:	1,7 m^2	0,39 m^2

Çizelge 4.2’de yer alan veriler incelendiğinde GTTKP yöntemi boş iç mekânın %82,47’sini kapsarken, önerilen Km++TKP yönteminin iç mekânın %95,97’sini kapsadığı görülmektedir. Bu hesaplama robotun işlemcisinde gerçekleştirilmiştir.

Bu işlemden sonra deney ortamına engeller konulmuştur ve robot tekrar haritalama yapmak üzere gezinmeye başlamıştır (Şekil 4.9). Toplam alan 9,7 m^2 iken gezinilebilir alan 8.98 m^2'dir.

Şekil 4.9 Gerçek dünya engel içeren iç mekân deney alanı

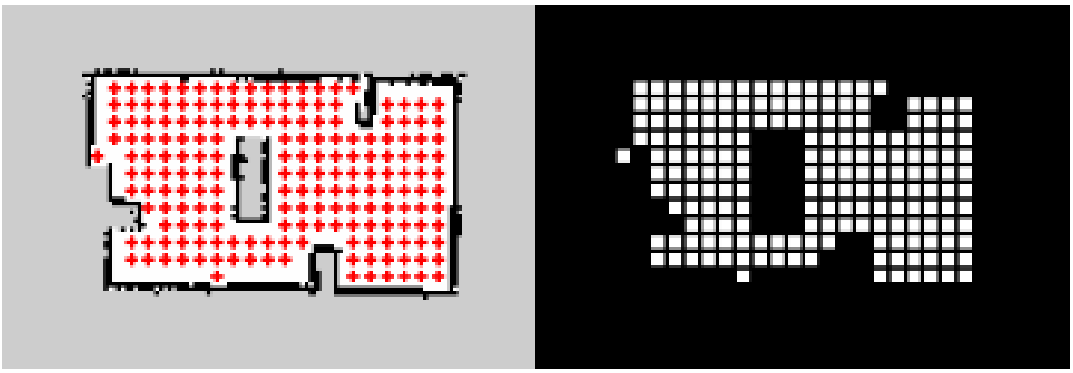
Haritalama tamamlandığında ortamın haritası kaydedilmiştir (Şekil 4.10(a)) ve robot tekrar başlangıç konumuna dönmüştür. Çevrenin üstten görünümü ve robotun başlangıç noktası Şekil 4.10(b)'de görselleştirilmiştir.



(a) (b)

Şekil 4.10 (a) Gerçek dünya engel içeren iç mekânda oluşturulan harita ve (b) iç mekânın üstten görünümü

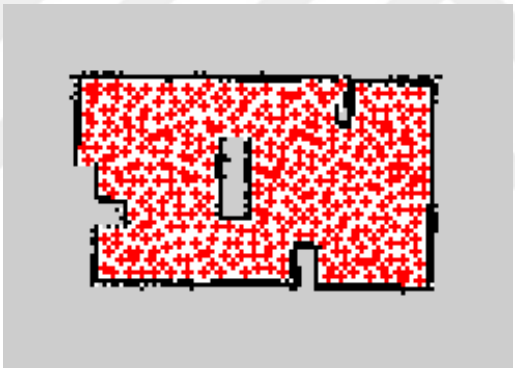
Yapılan uygulama sonucu elde edilen harita, GTTKP yöntemi ve araştırma kapsamında önerilen yöntem (Km++TKP) ile ayrı ayrı işlenmiştir. Haritaya GTTKP yöntemi uygulandığında robotun gezineceği 186 hücre merkezi hesaplanmıştır. Bu merkezler, Şekil 4.11(a)'da kırmızı noktalar şeklinde görselleştirilmiştir. Haritanın monokrom gösterimi Şekil 4.11(b)'de verilmiştir.



(a) (b)

Şekil 4.11 (a) Gerçek dünya engel içeren iç mekânda GTTKP yöntemiyle oluşturulan ağırlık merkezleri ve (b) haritanın monokrom gösterimi

Haritaya araştırma kapsamında önerilen Km++TKP yöntemi uygulandığında gezinilebilir alanın ölçüsünün robotun tek seferde kapsadığı alanın ölçüsüne bölünmesiyle *k* değeri 347 hesaplanmıştır. Kümeleme işlemi sonunda elde edilen ağırlık merkezleri Şekil 4.12'de görselleştirilmiştir.



Şekil 4.12 Gerçek dünya engel içeren iç mekânda önerilen Km++TKP yöntemiyle oluşturulan ağırlık merkezleri

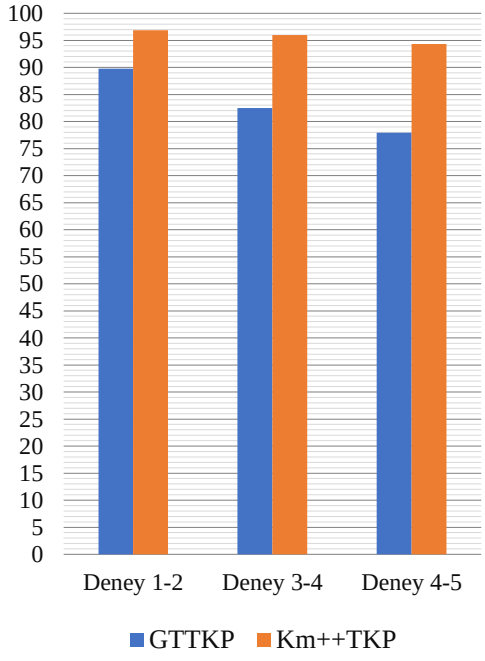
Buna göre; iki yöntemin de kapsama planı hesaplama süresi ve kapsama oranları aşağıdaki Çizelge 4.3'te verilmiştir.

Çizelge 4.3 Gerçek dünya engel içeren iç mekânda iki yöntemin kapsama planı hesaplama süresi ve kapsama ölçülerinin karşılaştırılması

	GTTKP	**Km++TKP**
Hücre /Ağırlık Merkezi Sayısı:	186	347
İşlem Süresi:	4,8004 sn	41,6573 sn
Kapsanan Alan:	7 m^2	8,47 m^2
Kapsanamayan Alan:	1,98 m^2	0,51 m^2

Çizelge 4.3’te yer alan veriler incelendiğinde GTTKP yöntemi engeller bulunan iç mekânın %77,95’ini kapsarken, önerilen Km++TKP yönteminin iç mekânın %94,32’sini kapsadığı görülmektedir. Bu hesaplama robotun işlemcisinde gerçekleştirilmiştir.

Çalışma kapsamında yapılan tüm deneylerden elde edilen bulgular Şekil 4.13'te grafiksel olarak gösterilmiştir.



Şekil 4.13 GTTKP ve Km++TKP yöntemlerinin yapılan deneylerdeki iç mekânları kapsama oranları

Önerilen Km++TKP yönteminin simülasyon ortamında iç mekânın %96,86'sını, gerçek dünya boş iç mekânın %95,97'sini ve gerçek dünya engel içeren iç mekânın ise %94,32'sini kapsadığı görülmüştür. GTTKP yönteminin ise simülasyon ortamında iç mekânın %89,76'sını, gerçek dünya boş iç mekânın %82,47'sini ve gerçek dünya engel içeren iç mekânın ise %77,95'ini kapsadığı gözlemlenmiştir. Buna göre, Km++TKP yönteminin kapsama oranı, tüm deneylerde GTTKP yönteminden daha yüksektir.

5. SONUÇ

Bu çalışmada ele alınan problem, GTTKP yönteminde kısmen dolu olan hücrenin tamamen dolu olarak işlenmesidir (Almadhoun, Taha, Seneviratne ve Zweiri, 2019). Almadhoun ve arkadaşlarının (2019) ifade ettiği bu problem çalışma kapsamında yapılan simülasyon ve gerçek dünya deneyleriyle doğrulanmıştır. Bunun üzerine GTTKP yönteminde karşılaşılan kapsama probleminin ortamın özellikleri dikkate alınarak oluşturulacak kümeleme yönteminin kullanılabilmesi araştırma sorusu olarak belirlenmiştir. Bu doğrultuda Arthur ve Vassilvitskii tarafından 2007 yılında alanyazına kazandırılan ve yaygın olarak kullanılan bir kümeleme algoritması ve bölümleme tekniği olan K-means++ algoritmasının kullanılabilmesi amaçlanmıştır. Bu amaca yönelik olarak, bir mobil robotun gezineceği iç mekânın haritasındaki gezinilebilir alanın K-means++ algoritması kullanılarak kümelendiği ve ağırlık merkezlerinin yer işareti olarak kullanılabileceği bir çevrim dışı TKP yöntemi olan Km++TKP yöntemi önerilmiştir.

Önerilen yöntem simülasyon ortamı ve gerçek dünyada robot üzerinde yapılan deneyler ile doğrulanmıştır. Aynı zamanda önerilen yöntemin başarımı ile GTTKP yönteminin başarımı karşılaştırılmıştır. Buna göre; tüm deneylerde önerilen yöntemin iç mekânı kapsama başarımı GTTKP yöntemine göre daha yüksek hesaplanmıştır. Buna karşın GTTKP yönteminin kapsama planlama süresi tüm deneylerde önerilen yönteme göre daha düşük çıkmıştır. Bu duruma harita dosyası üzerindeki gezinilebilir alandaki piksellerin kümelenmesi işlemi neden olmaktadır. Kümeleme performansının arttırılması önerilen yöntemin tam kapsama planlama süresinin düşmesinin önünü açacaktır.

Çalışma kapsamında önerilen Km++TKP yönteminin kapsama başarımını test etmek için iki deney ortamı hazırlanmıştır. Bu ortamlardan ilki Gazebo'da bir simülasyon ortamıdır. İkincisi araştırma sırasında hazırlanan gerçek iç mekândır. Gerçek iç mekân iki farklı şekilde kullanılmıştır. Birincisinde ortam boştur, sadece duvarlardan oluşur; diğerinde ortamda duvarlarla birlikte engeller vardır. Yapılan deneylerden elde edilen bulgular incelendiğinde önerilen yöntemin; simülasyon iç mekânının %96,86'sını, gerçek dünya boş iç mekânının %95,97'sini ve gerçek dünya engel içeren iç mekânının %94,32'sini kapsadığı görülmüştür. GTTKP yönteminin ise simülasyon iç mekânının %89,76'sını, gerçek dünya boş iç mekânının %82,47'sini ve gerçek dünya engel içeren iç mekânının %77,95'ini kapsadığı görülmüştür. Buna göre önerilen yöntemin kapsama başarımı tüm deneylerde GTTKP yöntemine göre daha yüksektir.

Gerçek dünya engel içeren iç mekânda GTTKP yönteminin başarımı, aynı mekânın boş olması durumuna göre %4,52 düşmüştür. Önerilen yöntemde bu farkın %1,65 olduğu görülmüştür. Buradan hareketle çalışmada ele alınan problemin çözümüne yönelik, kapsama uygulamalarında en yaygın kullanılan GTTKP yöntemine göre kapsama başarımı daha yüksek bir yöntem önerilmiştir. Aynı zamanda GTTKP yönteminin gerçek dünya boş iç mekânda ve gerçek dünya engel içeren iç mekândaki performansının değişimi deney sonuçları ile desteklenerek ortaya konmuştur.

Sonraki çalışmalarda, önerilen yöntemin tam kapsama planlama süresinin düşürülmesi ve kapsama planı sonucu hesaplanan ağırlık merkezlerinin gezinmesinin planlanlanarak her iki yöntemin gezinme başarımları karşılaştırılabilir.

KAYNAKLAR

Acar, E.U., Choset, H., Rizzi, A.A., Atkar, P.N., Hull, D. (2002). Morse decompositions for coverage tasks. *The International Journal of Robotics Research*, 21(4), 331-344. doi:10.1177/027836402320556359

Ackerman, E. (2013, Mart 26). *TurtleBot Inventors Tell Us Everything About the Robot.* IEEE Spectrum: https://spectrum.ieee.org/automaton/robotics/diy/interview-turtlebot-inventors-tell-us-everything-about-the-robot

Almadhoun, R., Taha, T., Seneviratne, L., Zweiri, Y. (2019). A survey on multi-robot coverage path planning for model reconstruction and mapping. *SN Applied Sciences*, 1(8), 1-24. doi:10.1007/s42452-019-0872-y

Apuroop, K.G.S., Le, A.V., Elara, M.R., Sheu, B.J. (2021). Reinforcement learning-based complete area coverage path planning for a modified hTrihex robot. *Sensors*, 21(4), 1067. doi:10.3390/s21041067

Arthur, D., Vassilvitskii, S. (2007). *K-Means++: The advantages of careful seeding*. Stanford.

Bailey, T., Durrant-Whyte, H. (2006). Simultaneous localization and mapping (SLAM): Part II. *IEEE Robotics and Automation Magazine*, 13(3), 108-117. doi:10.1109/MRA.2006.1678144

Bosse, M., Nourani-Vatani, N., Roberts, J. (2007). Coverage algorithms for an underactuated car-like vehicle in an uncertain environment. *IEEE International Conference on Robotics and Automation*, 698-703. doi:10.1109/ROBOT.2007.363068

Brooks, R.A. (1986). A robust layered control syste for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1), 14-23. doi:10.1109/JRA.1986.1087032

Butler, Z.J., Rizzi, A.A., Hollis, R.L. (1999). Contact sensor-based coverage of rectilinear environments. *1999 IEEE International Symposium on Intelligent Control Intelligent Systems and Semiotics*, 266-271. doi:10.1109/ISIC.1999.796666

Cao, Z.L., Huang, Y., Hall, E.L. (1988). Region filling operations with random obstacle avoidance for mobile robotics. *Journal of Robotic Systems* 5(2), 87-102. doi:10.1002/rob.4620050202

Castellanos, J.A., Neira, J., Tard´os, J.D. (2004). Limits to the consistency of EKF-based SLAM. *IFAC Proceedings Volumes*, 37(8), 716-721. doi:10.1016/s1474-6670(17)32063-3

Cheng, P., Keller, J., Kumar, V. (2008). Time-optimal UAV trajectory planning for 3D urban structure coverage. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2750-2757. doi:10.1109/IROS.2008.4650988

Choset, H. (2001). Coverage for robotics - A survey of recent results. *Annals of Mathematics and Artificial Intelligence*, 31, 113-126. doi:10.1023/a:1016639210559

Choset, H., Pignon, P. (1998). Coverage path planning: The boustrophedon cellular decomposition. *Field and Service Robotics*, 203-209. doi:10.1007/978-1-4471-1273-0_32

Crowley, J.L. (1985). Navigation for an intelligent mobile robot. *IEEE Journal of Robotics and Automation*, 1(1), 31-41. doi:10.1038/sj.bjp.0704266

Çelebi, M.E., Kingravi, H.A. (2015). *Linear, deterministic, and order-ınvariant ınitialization methods for the k-means clustering algorithm*. *Partitional Clustering Algorithms*, 79-98.

Di Franco, C., Buttazzo, G. (2016). Coverage path planning for UAVs photogrammetry with energy and resolution constraints. *Journal of Intelligent and Robotic Systems*, 83, 445-462. doi:10.1007/s10846-016-0348-x

Dissanayake, G., Durrant-Whyte, H., Bailey, T. (2000). A computationally efficient solution to the simultaneous localisation and map building (SLAM) problem. *IEEE International Conference on Robotics and Automation*, 2, 1009-1014. doi:10.1109/ROBOT.2000.844732

Doucet A., de Freitas N., Gordon N. (2001). *An introduction to sequential monte carlo methods. Sequential Monte Carlo Methods in practice*. *Statistics for Engineering and Information Science.* doi:10.1007/978-1-4757-3437-9_1

Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6), 46-57. doi:10.1109/2.30720

Farsi, M., Ratcliff, K., Johnson, J.P., Allen, C.R., Karam, K.Z., Pawson, R. (1994). Robot control system for window cleaning. *American Control Conference*, 1, 994-995. doi:10.1109/ACC.1994.751894

Gabriely, Y., Rimon, E. (2002). Spiral-STC: an on-line coverage algorithm of grid environments by a mobile robot. *IEEE International Conference on Robotics and Automation*, 1, 954-960. doi:10.1109/ROBOT.2002.1013479

Galceran, E., Carreras, M. (2013). A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, 61(12), 1258-1276. doi:10.1016/j.robot.2013.09.004

Gonzalez, E., Alvarez, O., Diaz, Y., Parra, C., Bustacara, C. (2005). BSA: A complete coverage algorithm, *IEEE International Conference on Robotics and Automation*, 2040-2044. doi:10.1109/ROBOT.2005.1570413

Guastella, D.C., Cantelli, L., Giammello, G., Melita, C.D., Spatino, G., Muscato, G. (2019). Complete coverage path planning for aerial vehicle flocks deployed in outdoor environments. *Computers & Electrical Engineering*, *75*, 189-201. doi:10.1016/j.compeleceng.2019.02.024

Guimarães, R.L., de Oliveira, A.S., Fabro, J.A., Becker, T., Brenner, V.A. (2016). *ROS navigation: Concepts and tutorial*. *Studies in Computational Intelligence*, 625, 121-160. doi.org/10.1007/978-3-319-26054-9_6

Hadji, S.E., Kazi, S., Hing, T.H., Mohamed Ali, M.S. (2015). A review: simultaneous localization and mapping algorithms. *Jurnal Teknologi*, 73(2), 25-29. doi:10.11113/jt.v73.4188

Hert, S., Tiwari, S., Lumelsky, V. (1996). *A terrain-covering algorithm for an AUV*. Underwater Robots. doi:10.1007/978-1-4613-1419-6_2

Hörner, J. (2016). *Map-merging for multi-robot system*. Lisans Tezi.

Khan, A., Noreen, I., Ryu, H., Doh, N.L., Habib, Z. (2017). Online complete coverage path planning using two-way proximity search. *Intelligent Service Robotics*, 10, 229-240. doi:10.1007/s11370-017-0223-z

Latombe, J. (1991). *Robot Motion Planning*. Kluwer Academic Publishers. doi:10.1007/978-1-4615-4022-9

Le, A.V., Nhan, N.H.K., Mohan, R.E. (2020). Evolutionary algorithm-based complete coverage path planning for tetriamond tiling robots. *Sensors*, 20(2). doi:10.3390/s20020445

Lemaire, T., Lacroix, S., Sol`a, J. (2005). A practical 3D bearing-only SLAM algorithm. *IEEE/RSJ International Conference on Intelligent Robots and Systems IROS*, 2449-2454. doi:10.1109/IROS.2005.1545393

Li, C.H., Fang, C., Wang, F.Y., Xia, B., Song, Y. (2019). Complete coverage path planning for an Arnold system based mobile robot to perform specific types of missions. *Frontiers of Information Technology and Electronic Engineering*, 20(11), 1530-1542. doi:10.1631/FITEE.1800616

Li, Y., Shi, C. (2018). Localization and navigation for indoor mobile robot based on ROS. *Chinese Automation Congress (CAC)*, 1135-1139. doi:10.1109/CAC.2018.8623225

Lumelsky, V.J., Mukhopadhyay, S., Sun, K. (1990). Dynamic path planning in sensor-based terrain acquisition. *IEEE Transactions on Robotics and Automation*, 6(4), 462-472. doi:10.1109/70.59357

Luo, C., Yang, S.X. (2002). A real-time cooperative sweeping strategy for multiple cleaning robots. *IEEE Internatinal Symposium on Intelligent Control*, 660-665. doi:10.1109/ISIC.2002.1157841

Luo, C., Yang, S.X., Stacey, D.A., Jofriet, J.C. (2002). A solution to vicinity problem of obstacles in complete coverage path planning. *2002 IEEE International Conference on Robotics and Automation*, 1, 612-617. doi:10.1109/ROBOT.2002.1013426

MacQueen, J.B. (1967). Some methods for classification and analysis of multivariate observations. *5th Berkeley Symposium on Mathematical Statistics and Probability*, 1(14), 281-297.

Mishra, R., Javed, A. (2018). ROS based service robot platform. *4th International Conference on Control, Automation and Robotics*, 55-59. doi:10.1109/ICCAR.2018.8384644

Moravec, H., Elfes, A. (1985). High resolution maps from wide angle SONAR. *IEEE International Conference on Robotics and Automation*, 2, 116-121. doi:10.1109/ROBOT.1985.1087316

Najjaran, H., Kircanski, N. Goldenberg, A.A. (2001). Map building for a terrain scanning robot. *IEEE International Conference on Robotics and Automation*, 3728- 3733. doi:10.1109/ROBOT.2001.933198

Nedjati, A., Izbirak, G., Vizvari, B., Arkat, J. (2016). Complete coverage path planning for a multi-UAV response system in post-earthquake assessment. *Robotics*, 5(4), 26. doi:10.3390/robotics5040026

Oh, J. S., Choi, Y.H., Park, J.B., Zheng, Y. (2004). Complete coverage navigation of cleaning robots using triangular-cell-based map. *IEEE Transactions on Industrial Electronics*, 51(3), 718-726. doi:10.1109/TIE.2004.825197

Oksanen, T., Visala, A. (2009). Coverage path planning algorithms for agricultural field machines. *Journal of Field Robotics*, 26(8), 651-668. doi:10.1002/rob.20300

Ollis, M., Stentz, A. (1997). Vision-based perception for an automated harvester. *IEEE/RSJ International Conference on Intelligent Robot and Systems, Innovative Robotics for Real-World Applications, IROS'97*, 3, 1838-1844. doi:10.1109/IROS.1997.656612

Özbek, Ç. (2010). *Çok gezgin robotlu tam kapsama yol planlaması için sarmal kapsar ağaç tabanlı bir yaklaşım*. Yüksek Lisans Tezi.

Patle, B.K., Ganesh, B.L., Pandey, A., Parhi, D.R.K., Jagadeesh A. (2019). A review: On path planning strategies for navigation of mobile robot. *Defence Technology*, 15(4), 582-606. doi:10.1016/j.dt.2019.04.011

Prabakaran, V., Elara, M.R., Pathmakumar, T., Nansai, S. (2018). Floor cleaning robot with reconfigurable mechanism. *Automation in Construction*, 91, 155-165. doi:10.1016/j.autcon.2018.03.015.

Pyo, Y., Cho, H., Jung, R., Lim, T. (2017). *ROS robot programming*. ISBN: 9791196230715

Rekleitis, I., New, A.P., Rankin, E.S., Choset, H. (2008). Efficient boustrophedon multi-robot coverage: an algorithmic approach. *Annals of Mathematics and Artificial Intelligence*, 52(2), 109-142. doi:10.1007/s10472-009-9120-2

ROS Wiki. (2019). Gmapping package. http://wiki.ros.org/gmapping

ROS Wiki. (2014). ROS filesystem level. http://wiki.ros.org/ROS/Concepts

Saeedi, S., Trentini, M., Seto, M., Li, H. (2016). Multiple-robot simultaneous localization and mapping: A review. *Journal of Field Robotics*, 33(1), 3-46. doi:10.1002/rob.21620

Santos, J.M., Portugal, D., Rocha, R.P. (2013). An evaluation of 2d slam techniques available in Robot Operating System. *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 1-6. doi:10.1109/SSRR.2013.6719348

Seriani, S., Cortellessa, A., Belfio, S., Sortino, M., Totis, G., Gallina, P. (2015). Automatic path- planning algorithm for realistic decorative robotic painting. *Automation in Construction*, 56, 67-75. doi:10.1016/j.autcon.2015.04.016.

Sidhu, H.K. (2020). *Performance evaluation of pathfinding algorithms*. Yüksek Lisans Tezi.

Singh, A., Yadav, A., Rana, A. (2013). K-means with three different distance metrics. *International Journal of Computer Applications*, 67(10), 13-17.

Singh, D., Trivedi, E., Sharma, Y., Niranjan, V. (2018). TurtleBot: design and hardware component selection. *International Conference on Computing, Power and Communication Technologies (GUCON)*, 805-809. doi:10.1109/GUCON.2018.8675050

Stachniss, C., Grisetti, G., Burgard, W. (2005). Information gain-based exploration using rao-blackwellized particle filters. *Robotics: Science and Systems*, 2, 65-72. doi:10.15607/rss.2005.i.009

Takaya, K., Asai, T., Kroumov, V., Smarandache, F. (2016). Simulation environment for mobile robots testing using ROS and Gazebo. *20th International Conference on System Theory, Control and Computing (ICSTCC)*, 96-101. doi:10.1109/ICSTCC.2016.7790647

Thrun, S. (1998). Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1), 21-71. doi:10.1016/S0004-3702(97)00078-7

Thrun, S. (2003). *Exploring artificial intelligence in the new millennium. Robotic mapping: A survey.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1-35. ISBN: 1558608117

Üstünel, M. (2018). *K-ortalamalar algoritmasına dayalı kümeleme analizi sistemi ve perakendecilik sektöründe uygulaması*. Yüksek Lisans Tezi.

Vega-Heredia, M., Mohan, R.E., Wen, T.Y., Siti'Aisyah, J., Vengadesh, A., Ghanta, S., Vinu, S. (2019). Design and modelling of a modular window cleaning robot, *Automation in Construction*, 103, 268-278, doi:10.1016/j.autcon.2019.01.025

Ben-Kiki, O., Evans, C., Ingerson, B. (2009). YAML ain't markup language (YAML™) version 1.1. Working Draft 2004-12-28.

Yang, C., Tang, Y., Zhou, L., Ma, X. (2018). Complete coverage path planning based on bioinspired neural network and pedestrian location prediction. *IEEE 8th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, 528-533. doi:10.1109/CYBER.2018.8688311

Zaman, S., Slany, W., Steinbauer, G. (2011). ROS-based mapping, localization and autonomous navigation using a Pioneer 3-DX robot and their relevant issues. *2011 Saudi International Electronics, Communications and Photonics Conference (SIECPC)*, 1-5. doi:10.1109/SIECPC.2011.5876943

Zelinsky, A., Jarvis, R.A., Byrne, J.C., Yuta, S. (1993). Planning paths of complete coverage of an unstructured environment by a mobile robot. *International Conference on Advanced Robotics*, 13, 533-538.

Zhou, Y., Sun, R., Yu, S., Sun, Y., Sun, L. (2018). A complete coverage path planning algorithm for cleaning robots based on the distance transform algorithm and the rolling window approach in dynamic environments. *IEEE 7th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, 1335-1340. doi:10.1109/CYBER.2017.8446258

Zhu, D., Tian, C., Sun, B., Luo, C. (2019). Complete coverage path planning of autonomous underwater vehicle based on GBNN algorithm. *Journal of Intelligent and Robotic Systems*, 94(1), 237-249. doi:10.1007/s10846-018-0787-7

Qian, W., Xia, Z., Xiong, J., Gan, Y., Guo, Y., Weng, S., Deng, H., Hu, Y. Zhang, J. (2014). Manipulation task simulation using ROS and Gazebo. *IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)*, 2594-2598. doi:10.1109/robio.2014.7090732

Wonnacott, D., Karhumaa, M., Walker, J., Önder, N. (2012). Autonomous navigation planning with ROS. *Michigan Technological University*, 2-12.

Wong S.C., MacDonald, B.A. (2003). A topological coverage algorithm for mobile robots. *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1685-1690. doi:10.1109/IROS.2003.1248886

Xu, L. (2011). Graph planning for environmental coverage. Doktora Tezi.

Xu, P.F., Ding, Y.X., Luo, J.C. (2021). Complete coverage path planning of an unmanned surface vehicle based on a complete coverage neural network algorithm. *Journal of Marine Science and Engineering*, 9(11), 1163. doi:10.3390/jmse9111163

Xuexi, Z., Guokun, L., Genping, F., Dongliang, X., Shiliu, L. (2019). SLAM algorithm analysis of mobile robot based on lidar. *Chinese Control Conference (CCC)*, 4739-4745. doi:10.23919/ChiCC.2019.8866200