KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

YÜKSEK LİSANS TEZİ

KAPALI ALANLARDA KONUM BELİRLEME İÇİN ROS TABANLI YÜKSEK HASSASİYETLİ VERİ TOPLAMA SİSTEMİ TASARIMI VE UYGULAMASI

MUHAMMET ALİ YÜCE

KOCAELİ 2020

KOCAELİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ

ANABİLİM DALI

YÜKSEK LİSANS TEZİ

KAPALI ALANLARDA KONUM BELİRLEME İÇİN ROS TABANLI YÜKSEK HASSASİYETLİ VERİ TOPLAMA SİSTEMİ TASARIMI VE UYGULAMASI

MUHAMMET ALİ YÜCE

Dr.Öğr.Üyesi Alpaslan Burak İNNER
Danışman, Kocaeli Üniv. .....

Doç.Dr. Kerem KÜÇÜK
Jüri Üyesi, Kocaeli Üniv. .....

Doç.Dr. Cüneyt BAYILMIŞ
Jüri Üyesi, Sakarya Üniv. .....

Tezin Savunulduğu Tarih: 23.01.2020

ÖNSÖZ VE TEŞEKKÜR

Bu tez çalışması, iç mekan konum tespiti için veri toplama adımı olan çevrimdışı adımda robot ile yüksek miktarlarda stabil veri toplama kabiliyeti kazanıdırılmış ve toplanılan verilerin işlenmesi sağlanmıştır.

Tez çalışmamın her aşamasında desteğini esirgemeyen, çalışmalarıma yön veren, bana güvenen danışmanım Dr. Öğr. Üyesi A. Burak İnner'e sonsuz teşekkürlerimi sunarım.

Bu günlere gelmemde büyük pay sahibi olan aileme ve dostlarıma teşekkürlerimi sunarım.

Bu çalışma Kocaeli Üniversitesi Bilimsel Araştırma Projeleri Koordinasyon Birimi tarafından desteklenmiştir. Proje Numarası: 2017/38

Ocak – 2020 Muhammet Ali YÜCE

İÇİNDEKİLER

ÖNSÖZ VE TEŞEKKÜR i
İÇİNDEKİLER ii
ŞEKİLLER DİZİNİ iii
TABLOLAR DİZİNİ v
SİMGELER VE KISALTMALAR DİZİNİ vi
ÖZET vii
ABSTRACT viii
GİRİŞ 1
1. GENEL BİLGİLER 2
1.1. Problemin Tanımı 2
1.2. Tezin Katkıları 2
1.3. Benzer Çalışmalar 3
1.4. Tezin Organizasyonu 8
1.5. İç Mekanlarda Haritalama ve Konum Belirleme 9
2. RSSI TEMELLİ KONUM BELİRLEME 12
2.1. Veri Toplama Teknikleri 13
2.1.1. Manuel veri toplama 14
2.1.2. ROS temelli otomatik veri toplama 15
2.2. RSSI Temelli Konum Belirleme Yöntemleri 16
2.2.1. Parmakizi yöntemi 18
2.2.2. Üçgenleme (Tirlateration) yöntemi 24
3. İÇ MEKANLARDA YÜKSEK HASSASİYETLİ KONUM BELİRLEME İÇİN ROS TEMELLİ VERİ TOPLAMA 30
3.1. Kullanılan Donanımlar ve Teknolojiler 30
3.2. Robot İle Verilerin Toplanması 32
3.2.1. Verici konumlarının toplanması 42
3.2.2. Android cihaz ile verilerin toplanması 44
3.3. Verilerin İşlenmesi ve Görselleştirilmesi 47
3.3.1. Verilerin robot ile işlenmesi 49
3.3.2. Verilerin Android uygulaması ile işlenmesi 55
3.3.3. Geleneksel parmakizi yöntemi uygulaması 59
3.4. Android Geliştirme Ortamı 62
3.4.1. Harita görünümü komponenti 63
3.4.2. Komponent üzerinde konum tespiti 65
3.5. Gerçek Dünyada Veri Toplama ve Konum Tespiti 68
4. SONUÇLAR VE ÖNERİLER 75
KAYNAKLAR 77
KİŞİSEL YAYIN VE ESERLER 80
ÖZGEÇMİŞ 81

ŞEKİLLER DİZİNİ

Şekil 2.1. Pİ veri toplama şematik gösterimi 22
Şekil 2.2. Pİ uygun veri bulma şematik gösterimi 23
Şekil 2.3. Parmakizi ağırlık hesaplaması şeması 24
Şekil 2.4. Üçgenleme metodu uygulanacak sistem 25
Şekil 2.5. Üçgenleme metodu ilk değer 26
Şekil 2.6. Üçgenleme metodu ikinci değer 27
Şekil 2.7. Üçgenleme metodu üçüncü değer 28
Şekil 2.8. Sinyal gücü ile Trilateration yöntemi 28
Şekil 3.1. Turtlebot 3 Burger 30
Şekil 3.2. Raspberry Pi 3 ve donanım şeması 31
Şekil 3.3. NRF24LE1 yongası 31
Şekil 3.4. OnePlus 5 akıllı telefon 32
Şekil 3.5. Robot ile haritalaması yapılacak olan simülasyon ortamı 33
Şekil 3.6. Robot ile ortamın haritasının oluşturulması 34
Şekil 3.7. Robot ile ortamın haritasının oluşturma işleminin tamamlanması 35
Şekil 3.8. nav_msgs/Odometry mesajının içeriği 39
Şekil 3.9. beacon_save adlı başlatma dosyasının çalıştırılması 43
Şekil 3.10. Veri toplama uygulaması için simülasyon ortamı ve RViz 46
Şekil 3.11. Veri toplama uygulamasının eş zamanlı ekran görüntüsü 46
Şekil 3.12. Robot üzerine yerleştirilen Android telefon 48
Şekil 3.13. Haritalaması yapılmış olan ortam 51
Şekil 3.14. Ortamın haritalanmış görüntüsü 51
Şekil 3.15. RViz ile robotun tahmini konumunu görüntüleme 52
Şekil 3.16. Robotun yerleştirildiği konum 53
Şekil 3.17. Harita üzerinde robot ve Bluetooth vericileri 54
Şekil 3.18. find_pose_service düğüm çalışma sonucu 54
Şekil 3.19. Sonucun harita üzerinde gösterimi (Kırmızı robot başlangıç konumu, mavi robot gerçek konumu, yeşil robot bulunan konumu temsil etmektedir.) 55
Şekil 3.20. PGM dosyasındaki verinin işlenmiş görseli 56
Şekil 3.21. Harita üzerinde konum verisinin gösterilmesi (Pempe nokta) 57
Şekil 3.22. Trilateration ile tespit edilen konumun gösterilmesi (Yeşil nokta) 58
Şekil 3.23. Uygulama üzerinde vericilerin görselleştirilmesi 59
Şekil 3.24. Geleneksel veri toplama uygulaması 60
Şekil 3.25. Geleneksel Pİ konum tespiti 62
Şekil 3.26. Harita bitmap dosyasının PGM dosyası ile oluşturulması 64
Şekil 3.27. Elde edilen haritanın uygulama üzerinde gösterimi 65
Şekil 3.28. Harita üzerinde origin noktasının gösterilmesi 68
Şekil 3.29. Ofis ortamı genel görünüm 69
Şekil 3.30. Ofis zemini 69
Şekil 3.31. Ofis ortamının robot tarafından oluşturulmuş haritası 70
Şekil 3.32. Çalışmanın uygulandığı ev ortamı ve vericilerinin konumları (Bluetooth simgeleri) 70

Şekil 3.33. Ev ortamının haritası 71
Şekil 3.34. Robot üzerine yerleştirilmiş telefon 72
Şekil 3.35. Android ile konum tespiti (Kırmızı nokta tespit edilen konum, Beyaz nokta gerçek konum) 72
Şekil 3.36. Robot ile toplanan veri (Üst), Oneplus 5 ile toplanan veri (Alt) 73
Şekil 3.37. Android cihaz ile geleneksel Pİ yöntemi 74

TABLOLAR DİZİNİ

Tablo 3.1. Robot ile toplanılan veri şeması 36
Tablo 3.2. Bluetooth konum girdileri 44
Tablo 3.3. Bluetooth verici konumları 44
Tablo 3.4. Veri setinin saklandığı veri yapısı 61
Tablo 3.5. Konum tespiti formülünde değerler 67

SİMGELER VE KISALTMALAR DİZİNİ

Kısaltmalar

AL : Artificial Landmark (Sanal İşaretleyici)
AMCL : Adaptive Monte Carlo (Adaptif Monte Karlo)
ANN : Artificial Neural Network (Yapay Sinir Ağları)
AoA : Angle of Arrival (Geliş Açısı)
AP : Access Point (Erişim Noktası)
AVM : Alışveriş Merkezi
BLE : Bluetooth Low Energy (Bluetooth Düşük Enerji)
CSI : Channel State Information (Kanal Durum Bilgisi)
DBSCAN : Density-Based Spatial Clustering of Applications With Noise (Yoğunluk Bazlı Uzaysal Kümelenmesi)
DNN : Deep Neural Network (Derin Sinir Ağı)
GPS : Global Postioning System (Küresel Konumlandırma Sistemi)
HMM : Hidden Markov Model (Gizli Markov Modeli)
IPS : Indoor Positioning System (İç Mekan Konumlandırma Sistemi)
kNN : k-Nearest Neighbor (k- En Yakın Komşu)
LSTM : Long Short Term Memory (Uzun Kısa Zamanlı Bellek)
LF-DLSTM : Local Feature-Based Deep Long Short-Term Memory (Derin Kısa Zamanlı Uzun Bellek Tabanlı Yerel Özellik)
LDA : Linear Discriminant Analysis (Doğrusal Diskriminant Analizine)
MLE : Maximum Likelihood Estimation (En Yüksek Olasılık Tahmini)
MVVM : Model – View – View Model (Model – Görünüm – Görünüm Modeli)
NL : Natural Landmarks (Doğal İşaretleyici)
Pİ : Parmakizi
PGM : Portable Gray Map (Taşınabilir Gri Harita)
PM : Probablistic Methods (Olasılıksal Metotlar)
PoA : Phase of Arrival (Geliş Fazı)
RFID : Radio Frequency Identification (Radyo Frekansı Tanımlayıcısı)
ROS : Robot Operating System (Robot İşletim Sistemi)
RToF : Return Time of Flight (Uçuşun Geri Dönüş Zamanı)
RSSI : Received Signal Strength Indicator (Alınan Sinyal Gücü Göstergesi)
SDA : Stacked Denoising Autoencoder (Yığın Gürültü Otomatik Kodlayıcı)
SLAM : Simultaneous Localization and Mapping (Eş Zamanlı Haritalama ve Lokalizasyon Problemi)
SVM : Support Vector Machine (Destek Vektör Makinesi)
TDoA : Time Difference of Arrival (Geliş Zaman Farkı)
ToF : Time of Flight (Uçuş Zamanı)

KAPALI ALANLARDA KONUM BELİRLEME İÇİN ROS TABANLI YÜKSEK HASSASİYETLİ VERİ TOPLAMA SİSTEMİ TASARIMI VE UYGULAMASI

ÖZET

İç mekanlarda konum tespiti evler, alışveriş merkezleri, fabrikalar gibi devasa çalışma alanları veya ofisler için bir insanın konumunu tespit edebilmek ve buna göre aksiyonlar alabilmek için çok önemlidir. Konum tespiti için günümüzde sıkça kullandığımız Küresel Konumlandırma Sistemi (GPS) dış ortamlarda çok iyi performans ile çalışırken iç mekanlarda engellerden geçemeyen sinyaller nedeniyle verimli çalışmamaktadır. Bunun sonucunda iç mekanlarda veri seti ile konum tespiti yapan yöntemler gerçekleştirilmiştir fakat bu yöntemler insan gücü ile veri toplanmasını temel aldığı için sancılı süreçlere neden olmuştur.

Bu çalışmada çevrimdışı aşama olarak literatürde yer alan bu veri toplama aşamasının otonomlaştırılması için Robot İşletim Sistemi (ROS) tabanlı otonom veri toplayan bir robot yazılımı gerçekleştirdik ve bu yazılımı simülasyon ortamından sonra gerçek dünya ortamında da çalıştırarak haritalama yaptık ve iç mekanda Bluetooth RSSI verilerini robotun konumu ile birlikte topladık. Ardından literatürde bulunan Parmakizi (Pİ) yönteminde ve Üçgenleme yöntemlerinden olan Trilateration yönteminde bu verileri kullanarak Android mobil telefonda konum tespiti yaptık ve kullanılabilirliğini gösterdik. Pİ yöntemi için klasik yöntem ile geliştirdiğimiz yöntemi karşılaştırdık. Herhangi bir yöntemin gerçekleştirilebileceği bir Android geliştirme ortamını da olarak sunduk. Deneysel çalışmalarımız bu yöntem ile veri toplama aşamasını otonomlaştırabildiğimizi ve topladığımız verileri farklı ortamlarda işleyerek kullanabildiğimizi kanıtlamıştır.

Anahtar Kelimeler: Android, İç Mekan Konum Tespiti, Parmakizi, Robot İşletim Sistemi, Üçgenleme.

DESIGN AND IMPLEMENTATION OF ROS BASED HIGH PRECISION DATA COLLECTION SYSTEM FOR DETERMINATION OF LOCATION IN INDOOR AREAS

ABSTRACT

Location detection is indispensable for houses, shopping malls, gigantic workplaces such as factories or offices to be able to locate a person or object and take actions accordingly to it. The Global Positioning System (GPS), which we use today for location detection works very well for outdoor environments but does not works efficiently due to the signals that cannot pass through obstacles indoors. As a result, many techniques developed to locating the user in interior environments. The data collection phase required in some of these techniques has been long and painful processes for people. Stable and in-depth data collection has a great importance for these methods, which collect data from the environment during the data collection stage and provide location estimation to the user at run time.

In this study, we introduced a Robot Operating System (ROS) based robotic software for autonomizing this data collection step which named offline phase in the literature. We performed this software in a real world environment after the simulation environment, we mapped the indoor environment first and after that we collected Bluetooth RSSI datas and the robot's location. Then, we used the data in Fingerprint method and Trilateration method which is one of the triangulation methods in the literature. Also we have determined the users location on the Android mobile phone and showed its usability and we compared the method we developed with the classical method for the PI method. We have also introduced an Android development environment where any method can be implemented.

Keywords: Android, Indoor Localization, Fingerprint, Robot Operating System, Trilateration.

GİRİŞ

İç Mekan Konum Tespit Sistemleri - Indoor Positioning Systems (IPS) - iç mekan navigasyonu, müşteri/misafir yönlendirmesi, nesne konumlandırması ve tespiti, müşteri analizi [1], en yakın/güvenli çıkış rotası (acil durum çıkışı) [2], AVM otoparkında araç konumu tespiti, görme engelli birey için yönlendirme [3], devasa yapılar içerisindeki insanların hedefledikleri konuma ulaşabilmeleri, devasa bir fabrikalarda çalışan işçilerin sorun çıkarmış olan bir makinenin bulunduğu bölüme en kestirme yoldan gidebilmeleri, akıllı evlerdeki cihazların ev sahibinin konumuna göre durum değişikliklerine uğraması, evde hasta takibi sistemleri veya bir alışveriş merkezinde müşterilere bulundukları konuma göre indirim veya promosyon kuponları sağlamak [4] gibi amaçlar için literatürde gerçekleştirilmiştir. İç mekan konumlandırması başta acil durumlarda hızlı çözüm üretmek olmak üzere kullanıcıların günlük hayattaki çoğu alışkanlığını değiştirmek için üzerine düşülen bir konudur.

Dış mekanlarda Küresel Konumlandırma Sistemi - Global Positioning System (GPS) yardımıyla çok düşük hata payları ile konum kestirimi yapılabilmektedir. Fakat iç mekanlarda GPS sinyalleri binalar, duvarlar, kapılar ve ağaçlara yakın bölgelerde engellere çarparak zayıfladığı için iç mekanlarda ve yeraltı mekanlarda etkili çalışmamaktadır [3]. Bu nedenle iç mekanlarda konum tespiti için literatürde çalışmalar gerçeklenmiştir. Bazı teknoloji girişimlerin temelini oluşturan iç mekan haritalama ve harita üzerinde konum tespiti son kullanıcıların hayatına girmek için çalışmaların yoğunlukla sürmeye devam ettiği bir problemdir. Problem farklı teknikler yardımı ile çözümlenmiştir. Bu yöntemlerin geliştirilmesi ile daha efektif konum tespiti yapabilen sistemler hala literatürde yer edinebilmektedirler.

1. GENEL BİLGİLER

1.1. Problemin Tanımı

IPS için literatürde çoğunlukla kullanılan Parmakizi (Pİ) yöntemi iki faz ile ifade edilmektedir. Bu fazlar çevrimdışı ve çevrimdışı fazlarıdır. Çevrimdışı fazda ortamdaki Pİ uygulamak için gerekli olan veriler toplanır. Bu veriler uygulanmak istenen metoda göre değişiklik gösterirken genellikle vericilerden toplanan Received Signal Strength Indicator (RSSI) yani alınan sinyal gücü göstergesi verileridir. Çevrimiçi faz ise toplanılmış olan bu veriler ile çalışma zamanında konum tespiti yapılan fazdır. Bu fazda da anlık olarak toplanılan RSSI verileri veri setindeki veriler ile karşılaştırılma işlemi yapılmaktadır.

Pİ yönteminin ilk fazı olan çevrimdışı faz yöntemin iyi çalışması için üzerine düşülmesi gereken bir konudur. Veri seti ne kadar büyük ve kapsayıcı olursa yöntemin de doğru orantılı olarak iyi konum tespiti yapması sağlanacaktır. Veri setinin büyüklüğünü bir noktadan toplanmış olan verilerin sayısı ile ölçülürken, kapsayıcılığı ise farklı ortam koşullarında toplanmış veriler ile ölçülür. Yöntemin bu fazı insan gücü ile (alıcı cihaz ele alınarak noktalardan veri toplayarak) gerçekleştirildiği için hem elde etmesi zorlu bir süreç olması, kapsamın genişlemesi için farklı ortam koşullarında tekrarlanmasının zorlayıcı olması ve çok da stabil bir veri seti sağlayamamanın yanında ayrıca da zorlu bir süreç oluşturmuştur. Farklı ortam koşullarında toplanılan verilerin insan kaynaklı hatalar nedeniyle de kalite anlamında çok da verimli veri setlerinin oluşamamasına neden olmuştur.

1.2. Tezin Katkıları

Bu çalışmada Pİ çevrimdışı fazı için; insan kaynaklı oluşabilecek hata payını ortadan kaldırmak, büyük veri setleri oluşturabilmek ve farklı ortam koşullarında aynı verileri toplayabilmek için ROS (Robot Operating System) destekli Turtlebot3 robotu ile veri seti oluşturulmuştur. Aynı zamanda veri setini Pİ yönteminin uygulanacağı cihaz özelinde oluşturabilmek için Android cihazlar için robot ile eş zamanlı veri seti oluşturacak bir mobil uygulama geliştirilmiştir. Bu verileri görselleştirmek ve

yöntemleri test edebilmek için ayrı bir Android uygulaması geliştirilmiştir. Bu uygulamada Pİ ve Üçgenleme yöntemlerinden olan Trilateration yöntemini gerçekleştirilmiştir. IPS yöntemlerinin uygulanabilmesi için bir Android Geliştirme Ortamı sunulmuştur. Sunulan yöntemi literatürdeki yöntemler ile karşılaştırabilmek için Android üzerinde insan eli ile Pİ yönteminin çevrimdışı ve çevrimiçi aşamalarını uygulayabilen bir Android uygulaması geliştirilmiştir. Pİ yöntemini robotun kendi konumunu bulması için kullandığı Adaptive Monte Carlo (AMCL) yöntemiyle entegre çalışabilecek şekilde gerçekleştirilmiştir.

1.3. Benzer Çalışmalar

Literatürde IPS için genellikle Pİ yöntemleri üzerine olmuştur ve bu yöntemin geliştirilmesi için çok fazla çalışma yapılmıştır. Pİ tekniği için olasılıksal metotlar ile çözümler üretilmişken aynı zamanda günümüzde oldukça popüler olan makine öğrenmesi, yapay sinir ağları gibi yöntemlerde bu teknik üzerinde gerçeklenmiştir. Robot alanında ise Pİ yöntemi Simultaneous Localization and Mapping (SLAM) çözümlerinde konum tespiti için kullanılmıştır.

Bruno ve Robertson, SLAM çözümlerinden olan FootSLAM ve PlaceSLAM çözümlerine RSSI ölçümlerinden elde edilen veri seti yardımı ile Pİ yöntemini birlikte çalıştırarak robot için konum tespitine yardımcı olmuşlardır [5].

Xu ve Chou, ROS destekli robotlar için IMU ve Lazer ile konum tespiti yapan AMCL yöntemini Pİ yöntemi ile birleştirerek robot konum tespiti yapmışlardır. Yaptıkları çalışma sonucunda konum tespiti süresini kısalttıklarını belirtmişlerdir [6].

Lee ve diğerleri, robot için konum tespitine yardımcı olması için önce Pİ ile kaba bir konum tespiti yapmasını sağlamışlardır. Bu kaba konum tespiti ile 10 m'ye kadar hata payı ile konum tespiti yapılmasını sağlamışlardır. Ardından Parçacık Filtre'sini kullanarak hata payını santimetrelere indirmişlerdir. Deneysel sonuçlar, önerilen yöntemin normal konum tespiti yöntemine kıyasla üstün performans gösterdiğini göstermektedir demişlerdir [7].

Zhang ve diğerleri, kablosuz konum tespiti için derin öğrenme tabanlı bir metot geliştirdiler. Bu metot için önceden Yığın Gürültü Otomatik Kodlayıcı (SDA) ile

eğitilmiş bir haritada 4 katmanlı bir Derin Sinir Ağı (DNN) oluşturmuşlardır. SDA ile büyük ve gürültü değeri yüksek bir veri setinden güvenilir verilerden öğrenme yapılmasını sağlamışlardır. Zaman tutarlılığını korumak ve başlangıç konumunun düzgünleştirilmesi için Gizli Markov Modeli (HMM) tabanlı bir konum tespit edici sunmuşlardır [9].

Alex ve diğerleri, literatürde IPS için çoğunlukla kullanılan WI-FI, Bluetooth, Ultrasound ve Kızılötesi teknolojileri kullanmak yerine bunların arasına GSM teknolojisini temel alan bir yöntem önermişlerdir. Bu yöntem ile büyük binalarda 4 m'lik kat doğruluğu sağlamakta ve çok katlı binalarda 2 kat içerisinde %98'e varan bir hassasiyete ulaşmakta olduğundan bahsetmişlerdir. Sistem GSM parmak izine dayalı olarak çalışmaktadır. GSM'in kullanılması Wi-Fi'a göre daha yüksek alanda hizmet sunduğu için kullanılması daha fazla alanda gerçeklenmesini sağlamıştır [10].

Farid ve diğerleri, çalışmalarında IPS'e olan ihtiyaçtan bahsetmişlerdir ve bu yöntemlerin talep bazlı pazarlardaki önemine değinmişlerdir. Bu çalışmada iç mekan konum tespiti sistemlerindeki teknolojilerden ve sistemleri performans, hassasiyet, maliyet, kapsam kullanılabilirlik gibi yönleriyle incelemişlerdir [11].

Jin ve diğerleri, GPS sistemlerinin çalışmadığı ortamlarda RFID teknolojisini kullanarak düşük maliyetli bir sistem ortaya koymuşlardır. LANDMARC iç mekan konum bulma sisteminden bahsetmişlerdir. Bu sistem referans etiketi konsepti ile RFID etiketi sayısını azaltarak yüksek doğruluk ile konum bulabilme garantisi verdiklerini söylemişlerdir. Makalelerinde hali hazırda var olan LANDMARC sistemlerini analiz etmişlerdir ve sahip oldukları problemler üzerinde durup iç mekan konumlandırması için geliştirilmiş bir LANDMARC önermişlerdir [12].

Güvenç ve diğerleri, literatürdeki verici ile alıcı arasındaki Geliş Zamanı Farkı (ToA) için görüş açısında olmayan verilerin etkilerini azaltarak iç mekan konumlaması üzerine çalışmışlardır. Farklı ToA tabanlı konumlandırma yöntemlerini bir araya toplayıp karşılaştırmalarını yapmış ve gelecekteki araştırma yönlerine dikkat çekmişlerdir [13].

Zhang ve diğerleri, gerçek zamanlı izleme, etkinlik tanıma, sağlık bakımı, navigasyon, acil durum tespiti ve ilgi alanı hedeflemesi için iç mekan konumlandırmasının

öneminden bahsetmişlerdir ve bu yönde yapılan çalışmaları inceleyerek karşılaştırmalar üzerine çalışmışlardır. Makalede zaman bazlı, açı bazlı ve sinyal bazlı sinyal ölçüm yöntemlerini incelemişlerdir ve bunların pozisyon hesaplamasında nasıl kullanıldığını göstermişlerdir. Pozisyon hesaplama yöntemleri olarak Trilateration, Triangulation, Maximum Likelihood Estimation (MLE) yöntemlerini incelemişlerdir. Aynı zamanda ağ tekniklerini ve sistemlerini de incelemişlerdir [14].

Jung ve diğerleri, yaptıkları çalışmada ışık yayan diyotlar (Light Emitting Diodes - LED)'leri kullanarak kablosuz iç mekan konumlaması yapmışlardır ve simülasyon ortamında gerçeklemişlerdir. Her LED lambaya benzersiz frekans atamaları yapmışlardır ve LED tarafından yayılan ışık üzerinden konumlama yapmışlardır. Faz farkı üzerinden Varış Zamanı Farkı (TDoA) konumlama yöntemi kullanarak yöntemi gerçeklemişlerdir. Hali hazırda sistemde bulunan LED lambalarını kullanabildikleri için sistem maliyeti uygun bir yöntem önermişlerdir. Simülasyon ortamında 5m x 5 m x 3 m ölçülerindeki bir odada 1cm' den daha az bir hata payı ile sistemi gerçeklemişlerdir [15].

Wu ve diğerleri, yöntemi gerçeklemeden alan taraması ve veri toplama aşamasını yapmadan iç mekan konumlandırması yapan cep telefonlarına ve WiFi altyapısına dayanan WILL adlı sistemi önermişlerdir. Bu sistem için geleneksel iç mekan konumlandırma yöntemleri için haritadan öncesinde veri toplamak gibi yüksek iş gücü ve maliyet gerektiren aşamayı ortadan kaldırdıklarını belirtmişlerdir. Sistem, 1600 metrekarelik bir binada gerçeklemişlerdir. WILL'in geleneksel yöntemlerle karşılaştırıldığında rekabetçi bir performans elde ettiğini belirtmişlerdir [16].

Wu ve diğerleri, RSSI tabanlı iç mekan konumlandırma sistemleri için RSSI sinyallerinin gürültüden kolayca etkilenmesinden yola çıkarak bu gürültüyü fiziksel katmanda hesaplayarak istenmeyen etkileri sistemden kaldırmak için çalışma yapmışlardır. Ortogonal frekans bölmeli çoğullama sistemlerinde alt taşıyıcıların frekans çeşitliliğini araştırmış ve alıcıda bir yayılma modeli ve bir parmak izi sistemi oluşturmak için Kanal Durum Bilgisini (CSI) kullanan, FILA adında yeni bir yaklaşım önermişlerdir. Deneysel sonuçlar, mesafe hesaplamasının doğruluğunun ve gecikmesinin CSI kullanılarak önemli ölçüde arttırılabileceğini göstermiş olduğunu belirtmişlerdir [17].

Zhu ve Feng, RSSI tabanlı iç mekan konumandırması için Üçgenleme temelli bir yöntem önermişlerdir. Bu yöntem sıradan Üçgenleme metodunun ele almadığı engelleri yani WiFi düğümüne sırtını dönen ve telefonu eliyle tutan kullanıcının veri üzerinde oluşturduğu engel gibi engelleri de hesaba katarak oluşturulmuştur. Yönteme AP Faced Trilateral adını vermişlerdir ve orijinal Üçgenleme tekniklerinden Trilateration ile karşılaştırmışlardır [18].

Chen ve diğerleri, WiFi sinyalleri ile iç mekan konumlandırması yapılan yöntemlerde RSSI sinyallerinde oluşan gürültüden çok etkilenmelerinden oluşan hatalardan dolayı FM sinyalleri ile iç mean konumlandırması yapabilen bir yöntem önermişlerdir. WiFi sinyalleri yüksek frekansa sahip oldukları için engellerden çok etkilenirler fakat FM sinyalleri düşük frekanslı olduğundan bu engellerden çok etkilenmez demişlerdir ve bu yöntemle geleneksel WiFi ile yapılan iç mekan konumlandırmalarına göre daha performanslı ve daha yüksek doğruluk oranına sahip bir sistem sunmuşlardır. Ve aynı zamanda WiFi ve FM sinyallerinin birlikte kullanıldığı sistemlerde WiFi'nin tek başına olduğu sistemlere göre %83'e varan oranda doğruluk hassasiyetinde artma sağladığını belirtmişlerdir [19].

Chintalapudi ve diğerleri, iç mekan konumlandırmasını önceden bir veri toplama işlemi olmadan nasıl yapılabileceği üzerine bir çalışma yapmışlardır. Bu çalışmada EZ Konumlandırma algoritmasını kuıllanmışlardır ve küçük bir yapıda 0,7m ile 4m arasında bir doğruluk oranı ile sonuçlar çıktısı vermişken büyük bir binada ise 2m ve 7m arasında bir doğruluk oranında sonuç çıktısı verdiğinden bahsetmişlerdir [20].

Xiong ve diğerleri, zamana dayalı iç mekan konumlandırma için frekans-çevik radyolardan yararlanarak ToneTrack adlı etkili bant genişliğini artırmak için frekans çevik kablosuz ağları kullanarak minimum donanım ve antenler ile alt metre hassasiyetini sağlayan bir iç mekan konumlandırma sistemi önermişlerdir. Deneysel sonuçlar, ToneTrack'ın, 20 MHz bant genişliği AP'leri bitişik kanallardan üç paketi duyduğunda medyan 90 cm hassasiyet elde edebileceğini göstermişlerdir [21].

Martin ve diğerleri, parmak izi yönteminin çevrimiçi ve çevrimdışı süreçlerini kullanarak mobil telefonların da WiFi anteni, hücresel veri anteni, ivmeölçer ve manyetometre olanaklarını araştırmışlardır. Ayrıca, radyo sinyal güçlerinin

istatistiksel olarak işlenmesi için mevcut deterministik teknikleri geride bırakabileceğini gösteren yeni bir yaklaşım uygulamışlardır [22].

Zafari ve diğerleri, iç mekan konum tespitinin son zamanlarda Nesnelerin interneti teknolojisi ile birlikte üzerine düşülen bir konu olduğundan bahsetmişlerdir. Makalelerini literatürde bulunan literatür araştırmasının artık eskimiş olmasından dolayı yazmışlardır. Geliş Açısı (AoA), Uçuş Zamanı (ToF), Dönüş Uçuşu Zamanı (RToF) ve Alınan Sinyal Gücü İşareti (RSSI) temelli yöntemler gibi birbirinden farklı iç mekan konum tespiti yöntemleri için detaylı bir tarama yapmayı amaçlamışlardır. İnsanların ve onların cihazlarının konum tespitinin yapılmasını tartışmışlardır. Halihazırda bulunan taramanın ışığında farklı sistemleri enerji verimliliği, maliyet, alım aralığı, gecikme süresi vb. perspektiflerden değerlendirmişlerdir. Teknolojileri karşılaştırmak yerine sistemleri karşılaştırmışlardır [4].

Kei ve diğerleri, LSTM (Long Short-Term Memory) tabanlı sinir ağı yapısını iç mekan haritalamasına uygulamışlardır ve BLE etiketlerinin sinyal seviyelerini çoklu tarayıcılar ile toplamışlardır. Geleneksel yöntemlerin ortam gürültüsünden çok etkilendiğini, özellikle de çok insanın bulunduğu ortamlarda yüksek hata oranı verdiklerinden bahsetmişlerdir. Önerilen yöntemleri ile gürültü engelleyici katmanlarını ve zamanlar serilerinden özellik çıkarımınu LSTM ile uyguladıklarını savunmuşlardır. Ağ zaman zaman serisini sinyal gücünü giriş olarak almaktadır ve çıkış olarak konum tahminini vermektedir. Sonuç olarak büyük boyutlu bir sahne için, sahnenin yüzde 75'i için 2,44 metrelik bir hata payı ile tahmin sonuçlarına en iyi konfigürasyon ile ulaşmışlardır [23].

Chen ve diğerleri, gürültüden etkilenen parmak izi yöntemleri için Derin Kısa Zamanlı Uzun Bellek Tabanlı Yerel Özellik (LF-DLSTM) adlı WiFi parmak izine dayanan bir yaklaşımı iç mekan haritalaması için önermişlerdir. Yerel özellik çıkarıcı veri üzerindeki gürültüyü azaltmaya çalışarak güçlü yerel özellikler çıkartmaktadır. Gerçek ortamlardaki denemelerinden elde ettikleri verilere görer 1,48 ile 1,75 metre arasında hata elde etmişlerdir. Yaklaşım ile en iyi yöntemler üzerinde %18,98 ile %54,36 oranında iyileştirme sağlamışlardır [24].

Mehryar ve diğerleri, mevcut filtreleme tekniklerine alternatif bir çözüm üzerinde durmuşlardır ve BLE beacon aracılığıyla iç mekan konum tespiti için İnanç Yoğuşma

Filtresinin (BCF) tanıtmakta / tartışmaktadır. İç mekan konum tahmini gibi uygulamalarda görüldüğü gibi, doğrusal olmayan ve Gauss olmayan gürültü profillerinin devam ettiği zorlu ortamlarda dayanıklı bir filtredir diye bahsetmişlerdir [25].

Zhou ve diğerleri, insansız hava araçları teknolojilerinin bu kadar hızlı gelişmesine karşın iç mekan konum tahminine olan ihtiyaçlarında bahsetmişlerdir ve bunun için kablosuz yerel bölge ağı (WLAN) tabanlı konum tahmini önermişlerdir. Çoğu iç mekanda günümüzde WLAN ağı bulunduğu için uygulanması kolay olmaktadır diye bahsetmişlerdir. Stabil olmayan RSSI sinyalleri için Gürültü ile uygulamaların Yoğunluk Bazlı Uzaysal Kümelenmesi (DBSCAN) yöntemini önermişlerdir. DBSCAN ile hareket yolundaki referans noktalarında mobil erişim noktasının (AP) tespitini sağlamışlardır, erişim noktasından gelen RSS verisinin konum tahmininde kullanılan RSSI verisine olan bağımlılığı azaltılması için veritabanından kaldırmışlardır ve çeşitli düzeltme yöntemlerini tatmin edici konum tahmini elde etmek için kullanmışlardır [26].

Luo ve diğerleri, iç mekan konumlandırmasında birden fazla kat barındıran yapılarda konumlandırma yapabilmek için Doğrusal Diskriminant Analizine (LDA) dayalı çok katlı bir tanımlama modeli önermişlerdir. Kat tanımlaması iki fazda yapılmaktadır. Bu fazlar çevrimiçi ve çevrimdışı fazlardır. Çevrimdışı fazında parmak izi veri tabanı WiFi düğümleri üzerinden yapılmıştır ve çevrimiçi fazda ise son kat numarası belirlenir ve eğitimli model çoğunluk oylama mekanizması ile birleştirilir demişlerdir. Kat numarası belirlendikten sonra K En Yakın Komşu (KNN) tabanlı algoritma olan LL_KNN yöntemi ile hedef kattaki konum bilgisini tespit etmişlerdir. Gerçek deney sonuçları ile, sistemlerinin hesaplama karmaşıklığını azaltmak için tüm düğümlerden ziyade yalnızca küçük bir WiFi düğümü parmak izi bilgisi kullanarak kat numarasını tanımlayabildiğini göstermişlerdir [27].

1.4. Tezin Organizasyonu

Tez kapsamında birinci başlık olarak RSSI Temelli IPS Veri Toplama Teknikleri'nden bahsedilmiştir ardından ikinci başlık olarak İç Mekanlar Yüksek Hassasiyetli Konum Belirleme için ROS Temelli Veri Toplama Yöntemi'nden bahsedilmiştir.

Birinci başlık altında elle veri toplama ve robot ile veri toplama tekniklerinden bahsedilmiştir. Ardından Pİ ve Trilateration alt başlıkları ile RSSI Temelli IPS yöntemlerinden bahsedilmiştir.

İkinci başlıkta ise robot ile veri toplamak için kullanılan teknolojiler ve yöntemlerden bahsedilmiştir. Yöntemlerden sonra bu yöntemleri gerçekleştirilmiş teknikler anlatılmıştır. Verileri nasıl görselleştirdiğimizi anlatıp bu veriler üzerinden konum belirleme tekniklerini nasıl gerçekleştirildiği anlatılmıştır.

Sonuçta gerçekleştirmiş olduğumuz eski tip veri toplama Pİ yöntemi ile gerçekleştirmiş olduğumuz Pİ yöntemi ile karşılaştırarak elde edilen veriler paylaşarak tezi sonuçlandırılmıştır.

1.5. İç Mekanlarda Haritalama ve Konum Belirleme

IPS, günümüzde birçok farklı metot ile çözümlenmeye çalışılan bir sistemdir. İç mekanlarda konum tespiti probleminin çözülmesi hemen hemen elektronik cihaz kullanan bütün kullanıcıların günlük hayatlarını etkilemenin yanında fabrika, hastane, büyük işyerleri, depolar ve benzeri büyük iç mekan ortamlarında çalışma ortamlarında çalışanların zamanı efektif kullanmalarını ve zaman zaman oluşabilen kaybolma gibi problemlerin önüne geçmesini sağlayacak bir gelişmedir.

Bahsedilen çalışma alanları dışında afet durumlarında da ilk müdahale için çok kritik kazanımlar sağlayacak bir çalışmadır. Örneğin büyük bir fabrikada çıkmış olan bir yangın durumunda fabrika içerisinde sıkışıp kalmış olan kişi ortam içerisindeki konumunu itfaiye eri ile paylaşarak hızlı bir yardım alabilecek veya bir hastanede çalışan hemşire için hastanenin elektronik donanımı ile entegre çalışan bir sistemde hasta için acil müdahale gereken bir durumda elektronik sistemin hemşire ile hastanın konumunu paylaşması ile hemşire kendi konumundan binlerce hasta arasından ihtiyacı olan hastaya en kısa sürede ulaşıp gerekli müdahaleyi yapabilecektir.

Bunlar ve benzeri yüzlerce durum için iç mekanlarda konum tespiti önümüzdeki teknolojik gelişmelerin aslında çözülmesi için en büyük gereksinim duydukları alanlardan birisi olmuş bulunmaktadır.

Bu ihtiyaçlardan doğan iç mekanlarda konum tespiti için öncelikli ihtiyaç konum tespiti yapılacak ortamın haritasının bulunmasıdır. Harita koşulunun olmasının sebebi konum tespiti yapmak için literatürde bulunan algoritmalar ile sayısal bir konum verisi elde edebiliyor olsak bile bu veriyi kullanıcının kullanabileceği bir şekilde işleyemedikten sonra herhangi bir anlam ifade etmeyecektir kullanıcı için ve kullanılmayacaktır. Bu nedenle yine literatürde bulunan haritalama yöntemleri ile ortamın haritasına sahip olacağız ve bu harita üzerinden algoritmalar vasıtası ile tespit ettiğimiz kullanıcının konumunu akıllı telefonu üzerinden kullanıcıya göstereceğiz ve bu şekilde kullanıcının ihtiyacına yönelik cevap verebileceğiz.

İç mekan haritalama için proje çizimleri gibi yöntemlerle bölgenin haritası çıkarılabilir fakat bu yöntem ile elimizde sadece bir harita geçecektir. Bizim ihtiyacımız olan ise bu ortama ait harita oluşturulurken eş zamanlı olarak haritadan konum bilgileri toplayarak daha sonra konum tespiti için kullanabilmektir. Aynı zamanda kroki gibi çizimlerin sayısal veri olarak aktarılması da ayrı bir zorluk olacaktır.

ROS ortamın haritasının oluşturulabilmesini robot vasıtası ile sağlar. ROS geliştirmeleri simülasyon ortamında hızlıca yapabilme imkânı sağlarken aynı zamanda gerçek dünyada denemek istenildiğinde ise algoritmanın çalışması için gerekli sensörlere sahip ROS destekli herhangi bir robot üzerinde çalışarak bir çok uğraştan geliştiriciyi izole etmekte.

TurtleBot 3 ise ROS ile tam uyumlu olarak çalışan bir robottur. Üzerinde bulunan LIDAR lazer sensörü ve mikro bilgisayarı üzerindeki Bluetooth alıcısı ile bu çalışmayı gerçekleyebilmemiz ihtiyacımız olan bütün kaynakları sunmaktadır.

Haritalama ve Lokalizasyon Problemi (Literatürde SLAM yani Simultaneous Localization and Mapping olarak geçmekte) robot dünyasının süregelen bir problemi olmakla beraber üzerine onlarca makale ve algoritma geliştirilmiş olan bir konudur. Şu anda literatürde bu problem için kullanılabilecek birçok algoritma bulunmaktadır ve ROS platformu bu algoritmaların çoğunu kolayca kurulabilecek şekilde sunmaktadır.

Biz haritalama için literatürdeki gmapping çözümünden faydalandık. Gmapping ortamın haritasını 2 boyutlu olarak çıkarılmasını sağlayan bir algoritmadır. Çıktı

olarak “.pgm” ve “.yaml” formatında iki dosya vermektedir. Bu dosyalar daha sonra Android uygulamasını hayata geçirirken haritayı oluştururken kullanılacaktır.

Haritalama çözümü ile haritayı elde edebiliyoruz fakat bu veri tek başına bir mobil akıllı telefon ile kullanıcının konumunu tespit edebilmek için yeterli bir veri değildir. Kullanıcının konumunu bulmak için önerdiğimiz yöntem ile robotun haritalama işlemini yaparken eş zamanlı olarak ortam içerisinde sabit yerlere yerleştirilmiş olan Bluetooth vericilerinden verileri toplamak. Ve daha sonra literatürde Pİ (Fingerprint) yöntemi olarak geçen yöntem ile kullanıcının konumunu tespit etmek için kullanmaktır.

Konum bulma için ise node.js çatı platformu üzerinden noble kütüphanesi yardımıyla TurtleBot 3 üzerindeki Bluetooth yongasına ulaşan RSSI sinyallerini ve robotun haritalama yaparken anlık bulunduğu konum bilgilerini topladık. Bu veriler de yine Android uygulamasında konum bulma servisi için kullanılacaktır.

IPS için standart bir teknik bulunmamaktadır, bunun sebebi ise birden fazla çözümün olması ve her çözümün farklı teknoloji gerektirmesinden dolayıdır. Bu çözümlerin karışıklık, tahmin, maliyet vb. gibi problemlere farklı ihtiyaçlardan dolayı bu farklı çözümler sunulmuştur [9].

IPS çözümü için sunulan teknikler tahmin hassasiyeti, sistemin maliyeti gibi parametreler göz önünde bulundurularak farklı çözümlere ev sahipliği yapmıştır. Bunlar Alınan Sinyal Gücü Göstergesi - Received Signal Strength Indicator (RSSI), Kanal Durum Bilgisi - Channel State Information (CSI), Parmakizi (Pİ) / Sahne Analizi (Fingerprinting/Scene Analysis), k- En Yakın Komşu - k-Nearest Neighbor (kNN), Geliş Açısı - Angle of Arrival (AoA), Uçuş Zamanı - Time of Flight (ToF), Geliş Zaman Farkı - Time Difference of Arrival (TDoA), Uçuşun Geri Dönüş Zamanı - Return Time of Flight (RToF) ve Geliş Fazı - Phase of Arrival (PoA) teknikleridir [4].

2. RSSI TEMELLİ KONUM BELİRLEME

İç mekanlarda konum belirleme için literatürde temel olarak kullanılan birkaç farklı teknik bulunmaktadır.

- Received Signal Strength Indicator (RSSI)
- Kanal Durum Bilgisi - Channel State Information (CSI)
- Parmakizi (Pİ) / Sahne Analizi (Fingerprinting/Scene Analysis)
- k- En Yakın Komşu - k-Nearest Neighbor (kNN)
- Geliş Açısı - Angle of Arrival (AoA)
- Uçuş Zamanı - Time of Flight (ToF)
- Geliş Zaman Farkı - Time Difference of Arrival (TDoA)
- Uçuşun Geri Dönüş Zamanı - Return Time of Flight (RToF)
- Geliş Fazı - Phase of Arrival (PoA)

teknikleridir. Bu teknikler vericiden toplanan bilgilerden yola çıkarak konum tespiti yaparlar. Bu tespitler tamamen olasılıksal yöntemlerdir yani hiçbir zaman kesin bir konum kestirimi yapmayacaklardır. Tekniklerin temel amacı gerçeğe en yakın tespiti yapabilmektir. RSSI temelli olan teknikler literatürde yoğunluk ile kullanılan tekniklerin başında gelmektedir. Pİ konum tespiti ise genelde temelinde RSSI verilerini kullanarak çalışan bir tekniktir.

İki fazda gerçekleşen Pİ teknikleri temelde RSSI verilerini kullanarak konum tespiti yaparlar. Birinci fazda veri toplayan Pİ tekniği ikinci fazda çalışma zamanında veri toplayarak birinci fazdaki veriler ile karşılaştırma yaparak konum tespiti yaparlar. Bu nedenle iyi gerçekleştirilmiş olan bir veri toplama fazı yani çok ve stabil veri toplayarak oluşturulmuş olan veri seti yöntemin iyi çalışması için çok önemli bir fazdır. Literatürde yapılmış olan çalışmalar genellikle insan gücü ile veri toplaması gerçekleştirmişlerdir ve bu nedenle ne noktalardan fazla sayıda veri toplanması ne de stabil, insan kaynaklı hatalardan arındırılmış veriler elde edilmesi sağlanabilmiştir. Robot ile yapılan çalışmalarda ise robotun kendi konumunu tespit edebilmesi için çalışmalar yapılmıştır.

Robot özelinde toplanan bu veri setleri sadece robotun kullanabileceği veri setleri olarak kısıtlı bir alana sıkışmışlardır.

Bu çalışmada yüksek doğruluk payı ile konum tespiti yapmak için harita üzerindeki belirli noktalardan yüksek miktarda veri toplayacak bir yöntem geliştirilmiştir. Toplanılan verilerin farklı platformlarda kullanılmasını sağlayarak da iç mekanlarda konum tespiti yapılmasını sağlanmıştır.

Bu ana başlık altında öncelikle veri toplama tekniklerinden bahsedilmiştir ve ardından iç mekanlarda RSSI temelli konum tespiti yöntemleri incelenmiştir.

2.1. Veri Toplama Teknikleri

Literatürdeki veri toplama tekniklerini manuel ve otomatik olarak sınıflandırabiliriz. Manuel veri toplama insanların ortamdan bilgisayar veya mobil bir cihaz vasıtası ile toplaması ile gerçekleştirilir. Otomatik veri toplamalarında ise ortamdan bir robot yardımı ile veri toplanması sağlanır.

Manuel veri toplamaları insan gücü ile yapıldığı için belli bir noktadan hiç sarsılmadan veya konum değişmeden yüksek miktarda veri toplanması uygulanması imkânsız bir hal alıyor. İnsanın doğası gereği bulunduğu noktada kendisini ayakta tutabilmek için ağırlık merkezine göre kendisini konumlandırırken yaşadığı sallantılar bile verinin gürültü etkisi ile toplanması için yeterli bir neden. Ayrıca alıcının etrafındaki insan engeli de yine vericilerden gelen sinyalin seviyesini düşürerek veri toplanmasına büyük bir gürültü faktörü eklemektedir.

Otomatik veri toplama da ise robotun hareket ettirilmesi ile robot üzerinde bulunan bir Bluetooth veya WiFi alıcısı ile ortamdan veri toplanması sağlanmıştır. Gerçekleştirilen sistemlerde robot insan eli ile uzaktan kontrol (teleoperation) edilerek ortamda gezdirilir ve veri toplanır.

Biz bu çalışmada veri toplama fazı için robot ile belirli bir örüntüde çok sayıda veriyi toplamak için bir yöntem önerilmiş ve geliştirilmiştir. Cihaz özelinde veri toplamak için de RosJava çatısını kullanarak robot ile senkron olarak veri toplayan bir Android uygulaması geliştirilmiştir. Aynı zamanda karşılaştırma yapabilmek için Android

mobil cihaz ile elle (manuel) veri toplayan ve Pİ konum tespiti yapabilen bir sistem geliştirilmiştir.

2.1.1. Manuel veri toplama

Manuel yani elle veri toplama yöntemi literatürde iç mekanlarda konum belirleme için yoğunlukla veri toplama aşamasının gerçekleştirildiği yöntemdir. Ortama yerleştirilmiş olan verici cihazlarından veri toplamayı ve toplanılan verilerin hangi konumdan toplanıldığının elle belirtildiği yöntemdir.

Veri toplama aşamasında bilgisayar veya üzerinde alıcı olan bir cihaz ile konum tespiti yapılacak olan ortamda gezinilerek ortamdan veri toplanılır. Bu şekilde veri toplamanın oluşturulmaya çalışılan veri seti üzerinde çok fazla gürültü değeri eklemesinden dolayı veri setinin çok sağlıklı oluşturulamayacak olması bir gerçektir.

İnsan eli ile veri toplamanın bir çok zorlayıcı yönü de bulunmaktadır. Bunları düşünecek olursa öncelikli olarak bir noktada veri toplayan bir insan bu noktada çok uzun bir süre dayanamayacaktır muhtemelen, bundan dolayı toplanılan veri miktarı insanın dayanabileceği sınır ile sınırlı kalacaktır. Ayrıca kısa aralıklar ile veri toplamak veri seti için oldukça önemlidir, bu sayede bir çok noktadan veri toplanmış olacağı için yapılan konum kestirimi amacımıza uygun bir şekilde olabildiğince gerçek kullanıcı konumuna yakın olacaktır. Fakat yine elle veri toplamadan dolayı toplanacak nokta sayısı da oldukça sınırlı olacaktır.

Manuel olarak veri toplama aşamasında ortamdan toplanabilecek veriler de sınırlı olmaktadır. Ortamın dijital bir haritasına sahip olmak konum belirleme yöntemlerinde son kullanıcıya sunum yapmak için oldukça önemli bir artıdır. Çünkü manuel veri toplama yönteminde noktalar için koridor, salon, mutfak, ofis, oda vb. gibi basit etiketlerden daha fazlasını kullanıcıya sunamaz ve bu iç mekanlarda konum tespitinin çok küçük bir ihtiyacını karşılamaktan fazlası olmaz.

Bu çalışmada manuel veri toplama yöntemini de Android platformunda gerçekleştirdik. Android cihazlar için geliştirdiğimiz uygulamayı Pİ yönteminin iki fazını da gerçekleştirmesi için kurguladık. Uygulama öncelikle ortamdan veri toplama aşamasını gerçekleştirerek bir veri seti oluşturulmasını sağlamakta, ikinci faz için ise

çalışma zamanında ortamdaki Bluetooth verilerini toplayarak veri setindeki verileri de kullanarak konum tespiti yaparak kullanıcıya ortamda bulunduğu konumu ifade eder.

2.1.2. ROS temelli otomatik veri toplama

İç mekanlarda konum tespiti yapabilmek için öncelikle ortamın bir haritasına ihtiyacımız vardır ki kullanıcının konumunu tespit ettiğimizde anlayabileceği bir şekilde bu veriyi görselleştirebilelim. Kullanıcının anlayabileceği veri ise ortamın kuş bakışı haritası üzerinde konumunun işaretlenmesidir.

Haritalama için Robot İşletim Sistemi (Robot Operating System - ROS) destekli TurtleBot 3 Burger robotu kullanılmıştır. Bu robot üzerinde bulunan meta işletim sistemi olan ROS'un da gücü ile birlikte kolay programlama imkânı ile birlikte üzerinde bulunan 360 LIDAR sensörü ile birlikte ortamın 2 boyutlu haritasına sahip olunabilmesini sağlamıştır.

Haritalama işlemi için ROS üzerinde çözümlenmiş olan Simultaneous Localization and Mapping (SLAM) - Eş zamanlı haritalama ve konumlandırma probleminin bir çözümü olan Gmapping yöntemi kullanılmıştır. Bu problem temelde robotun haritalama yaparken eş zamanlı olarak kendi konumunu bu harita üzerinde bulması için ortaya atılmıştır. Ortamın haritasını oluştururken kendi konumunu kestiremeyen robot gelen yeni verilerin ortamın haritasında neresine ait olduğunun kararını verebilmek için aynı zamanda hangi konumda olduğunu da bilmelidir.

Haritalama işlemi paralelde sürerken aynı zamanda bu harita üzerinden veri toplayarak haritanın Bluetooth parmak izi verilerini de Pİ yöntemini gerçeklerken kullanılanacak aynı zamanda bu verileri de toplayarak yöntemi gerçekleştirebilecek alt yapı hazırlanmıştır.

Verileri toplayabilmek için birden fazla yöntem denenmiş ve bu denemelerin sonunda en iyi yöntemi belirlenmiş ve kullanılmıştır.

Yöntemler önce Gazebo simülasyon ortamında çalıştırıldıktan sonra Turtlebot 3 robot üzerinde çalıştırılarak gerçek dünya ortamında veriler toplayarak da test edilmiştir.

Bluetooth verilerini toplamak için ise Turtlebot 3 üzerinde bulunan Raspberry Pi 3 B+ modeli üzerindeki Bluetooth yongası kullanılmıştır.

2.2. RSSI Temelli Konum Belirleme Yöntemleri

Literatürdeki iç mekan konum tespiti teknikleri; Alınan Sinyal Gücü Göstergesi - Received Signal Strength Indicator (RSSI), Kanal Durum Bilgisi - Channel State Information (CSI), Parmakizi (Pİ) / Sahne Analizi (Fingerprinting/Scene Analysis), k- En Yakın Komşu - k-Nearest Neighbor (kNN), Geliş Açısı - Angle of Arrival (AoA), Uçuş Zamanı - Time of Flight (ToF), Geliş Zaman Farkı - Time Difference of Arrival (TDoA), Uçuşun Geri Dönüş Zamanı - Return Time of Flight (RToF) ve Geliş Fazı - Phase of Arrival (PoA) başlıkları altında sıralanabilir [4].

Bu teknikler, herhangi bir veri setine ihtiyaç duymadan anlık elde edilen veriler ile konum tespiti yapan ve daha önceden toplanmış veriler ışığında konum tespiti yapan teknikler olarak iki başlık altında sıralanabilir. Bu tekniklerde görüntüler, etiketler veya kablosuz sinyaller gibi veriler tekniklerde kullanılan verilerdir. Özellikle kablosuz sinyal yayıcılar tarafından üretilen kablosuz sinyal verileri genellikle bu tekniklerde kullanılmaktadırlar.

Herhangi bir veri setine ihtiyaç duymayan teknikler tamamen matematiksel hesaplamalar ile tahmini konum bilgisi vermektedirler ve bu çıktılar genellikle hata oranı yüksek olan veriler olmaktadırlar. Bunun nedeni ise bu tekniklerin genellikle ortamda bulunan kablosuz sinyal yayıcılarına olan uzaklıkları, açıların veya verilerin gidip gelme gibi sürelerini hesaplayıp ardından buna göre yaptıkları tahminlerdir. Kullanılan bu verilerin ortamın gürültüsünden son derece şiddetli etkilenmesinden dolayı elde edilen veriler farklı senaryolar altında çok farklı sonuçlar vermekte ve hata oranları yüksek çıkmaktadır. RSSI, CSI, kNN, AoA, TDoA, RToF ve PoA teknikleri bu şekilde çalışırken Pİ / Sahne Analizi teknikleri ortamdan daha önceden toplanmış veri setindeki verileri kullanarak konum tahmini yapmaktadırlar.

Önceden toplanmış bir veri setine ihtiyaç duyan tekniklerde; haritadan daha önce toplanmış veriler önceki deneyimler olarak kabul edilir ve bu verilerin değişmeyeceği kabul edilerek konum kestirimi yapılır. Daha önceden konumları bilinen sinyal vericilerinin ortama yerleştirildiği bilinen bir ortamda, toplanmış olan sinyal ve o

sinyalin alınmış olduğu poz bilgileri bir deneyim olarak kabul edilir. Veri seti ile birlikte konum tespiti yapan teknikler, aynı şartlar altında yapılacak olan yeni bir taramada elde edilen sinyal eğer veri setinde var ise poz bilgisinin de ortalama olarak önceki deneyimdeki poz bilgisine yakın bir poz olacağı düşünülerek oluşturulan tekniklerdir.

Literatürde haritadan toplanmış veri seti ile konum tespiti yapan Olasılıksal Metotlar – Probablistic Methods (PM), Yapay Sinir Ağları – Artificial Neural Network (ANN), kNN ve makine öğrenmesi ile tespit yapan Destek Vektör Makinesi - Support Vector Machine (SVM) - yöntemleri mevcuttur. Bu teknikler Pİ / Sahne Analizi teknikleri başlığı altında incelenir [4]. Pİ yöntemi temel olarak haritadan toplanmış olan eşsiz verileri – bu veriler genellikle kablosuz sinyal verileridir fakat görüntü ile tespit yapan sistemlerde bir nesne görüntüsü de olabilir – önceki deneyimlerindeki veriler ile eşleştirerek poz bilgisi konusunda çıkarım yaparak konum tahmini yapmaktadır. Burada sistemler iki farklı yapıyı temel almaktadırlar. İlk yapı sanal işaretleyicileri – Artificial Landmarks (AL) - temel alırken ikinci yapılarda ise doğal işaretleyiciler – Natural Landmarks (NL) – ile poz eşleştirmesi yapılarak konum tahmini elde edilmektedir. Doğal işaretleyiciler ortamda bulunan duvarlar, masa bacakları ve köşeler gibi işaretlerden oluşurken sanal işaretleyiciler radyo frekanslarından oluşan dijital verilerden oluşur. Bu verilere radyo frekansı tanımlayıcıları – Radio Frequency Identification (RFID) – etiketleri ve ortamdaki Bluetooth, WiFi yayınlayıcılarından yayımlanan sinyaller örnek verilebilir. Gerçek işaretleyiciler ile oluşturulan veri setlerinden alışık olunmayan verileri tanımlamak bir robot için zordur [8]. Bu yüzden sistemlerde sanal işaretleyiciler ile işaretlemeler kullanılır.

Literatürde haritadan veri seti toplamak için kullanılan yöntemlerin gerçekleştirilmesi aşamasında veri toplamak için çok fazla yöntem kullanılmıştır. Bu yöntemler genellikle insan gücünü temel almıştır ve insanların veri setini ortam içerisinde ellerindeki dizüstü bilgisayarlar ile gezinerek toplamasıyla oluşturulmuştur ve gözlem yolu ile işaretlemeler yapıldığı için sistemlerdeki hata payı oranları da artmıştır. Aynı zamanda insan gücü ile toplanan verilerin bir sınırı olmuştur ve bir insanın aynı noktada stabil olarak durup dakikalarca veri toplaması gerekmiştir. Veri setinin büyük bir set olması istendiğinde ise bu sefer insan gücünün dayanabileceği sınırların dışında bir ister olmuştur ve bu durum çok büyük veri setleri oluşturmayı imkânsız kılmıştır.

Bu çalışmada önceden toplanmış veri seti ile yapılan iç mekan konum tespiti yöntemleri için veri seti toplama işlemini ROS (Robot Operating System) destekli TurtleBot 3 robotu ile yaparak sistemlerdeki insan kaynaklı hatalar ortadan kaldırılmış, insan gücü ile elde edilemeyecek kadar çok veri toplanmış ve sistemin tahmin kapasitesini arttırarak sistemin daha keskin ve hata oranı düşük bir sistem olmasını sağlanmıştır. Ayrıca çok büyük iç mekan ortamları için robotun pilinin bitmesi durumunda tekrar veri toplamaya başlaması durumunda kaldığı yerden devam edebilecek bir sistem gerçekleştirilmiştir. Aynı zamanda LIDAR sensörüne sahip robot vasıtasıyla ortamın 2 boyutlu haritası elde edilmiştir ve bu harita üzerinde literatürdeki Pİ yöntemi ve RSSI teknikleri altında bulunan Üçgenleme (Trilateration) yöntemi gerçekleştirilmiştir.

Gerçekleştirilen robot yazılımı sayesinde robotun ortamdan istenilen miktarda ilerlemesi sağlanarak noktalardan istenildiği kadar veri toplaması sağlanmıştır. Bu yazılım sayesinde ortamdan veri toplama senaryoları gerçekleştirilebilir ve bu senaryoların hepsine göre sonuç vermesi sağlanabilir. Ortamda insanlar yokken, az sayıda insan varken ve çok sayıda insan varken aynı veri tipinde veriler toplanılabilmesi imkanını sağlamıştır.

Veri toplamak için Android mobil cihaz uygulaması da gerçekleştirilmiştir. Bu uygulama ile de robot üzerindeki bir mobil cihazdan cihaza özel verilerin de toplanması sağlanmıştır.

Sonuçların test edilebilmesi için de son kullanıcılar için iç mekanlarda konum tespiti yapabilen bir Android uygulaması da gerçekleştirilmiştir. Bu mobil uygulama ise Pİ ve Üçgenleme yöntemlerini gerçekleştirerek sonuçlarını mobil uygulama üzerindeki robot tarafından oluşturulan harita üzerinde kullanıcıya sunmuştur.

2.2.1. Parmakizi yöntemi

İç mekan konumlandırması için literatürde birçok metot bulunmaktadır ve bu metotların büyük çoğunluğu Parkmakizi yöntemini temel almaktadır. Literatürdeki metotlar genelde bu yöntem için kullanılan metotların geliştirilmesi ve daha etkili kullanılabilmesi adına olmuştur.

Ölçüm tabanlı kablosuz iç mekan tespit sistemleri sinyalin Time of Arrival (TOA) veya Time Difference of Arrival (TDOA)'e bağlıdır. Fakat kablosuz sinyal alıcılarının geneli gidiş - dönüş zamanını veya açısını ölçemez. Dışarıdan bir cihaza ihtiyaç duyarlar. Bu nedenle Pamakizi yaklaşımlar dışarıdan herhangi özel bir cihaza ihtiyaç duymadan çalışabildikleri için gerçeklenmesi daha mümkündür [11].

Fingerprint yöntemi yani Parmakizi veya İmza yöntemi diye ifade edebileceğimiz yöntem temel olarak önceden verileri toplanmış olan bir harita üzerinde konum tespiti yapar. Bu yöntem literatürdeki çoğu IPS sisteminin temelinde bulunmaktadır. Pİ yöntemi temelde daha önceden ortamandan toplanmış verilerin işlenerek bu veriler üzerinden konum tespiti yapmayı sağlamaktadır.

Pİ yöntemi temelinde basitçe daha önceden deneyimlenmiş verileri sistemin değişmediğini düşünerek herhangi bir zaman diliminde tekrar deneyimlendiğinde aynı sonuçları vereceğini düşünmek olarak anlatılabilir.

Pİ yöntemi iki faz barındırır. Çevrimdışı fazında bütün erişim noktalarından (Access Point - AP) toplanan RSSI bilinen noktalardan referans noktalarıyla birlikte toplanır ve bir veritabanı oluşturulur. Ardından pozisyonlama aşamasında ise yeni toplanan RSSI sinyalleri ile veritabanındaki RSSI sinyallerinden eşleşen verileri tespit ederek en uygun pozisyonu bulur [28].

Bu yönteme göre bu proje için bir örnek vermek gerekirse bir kullanıcı evinin bir konumunda mobil cihazından WiFi ağlarını taradığında gördüğü sinyal değeri bugün -76 dBm iken yarın tekrar aynı sistemde aynı şekilde tarama yaptığında da alacağı sinyal değeri aşağı yukarı aynı olacaktır. Bu referans alınan WiFi erişim noktalarının sayısı ne kadar arttırılırsa yöntemin referans alacağı veri miktarı çoğalacağı için yöntemin doğruluk hassasiyeti yükselecektir.

Bu çalışmadaki amaca yönelik haritada imza olarak toplanılacak veriler harita üzerinde belirli noktalarda sabit olarak konumlandırılmış olan Bluetooth vericileridir. Bluetooth tarafından yayılan sinyalin güç seviyesini ifade etmek için kullanılan desibel cinsinden ifade edilen RSSI verisi ile imza veriler toplanılacaktır. Harita üzerinde konumlandırılmış ne kadar çok verici olursa toplanılan veri örneği daha fazla olacaktır ve bununla doğru orantılı olarak konum hassasiyeti de daha yüksek olacaktır. Aynı

zamanda konum hassasiyetini etkileyecek bir diğer etken ise harita üzerinden örnek toplanılan konumların sıklık miktarıdır. Harita üzerinde ne kadar çok noktadan veri toplarsak konum tahmini aşamasında bu noktadan toplanmış daha çok veriye sahip olunacağı için yöntemin sonuç olarak daha hassas konumlar tespit etmesini sağlamış olunur.

Bu yöntemin çalışması sonucundan ham olarak elimizde bir nokta verisi bulunacaktır. Bu veriyi ham olarak kullanıcıya göstermemiz herhangi bir anlam ifade etmeyecektir. Bu nokta verisi örneklerin toplandığı anda robotun bulunduğu konumlardan elde edilen konum verisidir. Bu veriyi kullanıcıya anlayabileceği bir veri türüne dönüştürerek gösterilmesi gerekmektedir. Bu aşamada konum ve RSSI verilerini toplarken aynı zamanda robotun 2D LIDAR sensörü ile oluşturmasını sağlanılan 2 boyutlu bir harita bulunmaktadır. Bu harita verisi de ham bir veri olarak elde bulunmakta, daha sonra bu verinin nasıl bir haritaya dönüştürüldüğünden bahsedilecektir. Bu harita verisi bir haritaya dönüştürüldükten sonra elimizdeki konum verisi artık harita üzerinde gösterilecek işlenmiş bir konum verisi olacaktır.

PİA yöntemini hayata geçirmek için öncelikle ortamın bir haritasına ihtiyaç duyulacağı için öncelikli hedef bir harita elde etmektir. Yöntemin gerçeklemesi için tek şart tabi ki haritaya sahip olmak değil, aynı zamanda bu haritadan toplanmış bir veri ağacına sahip olmak gerekir. Bu veri ağacında ise ihtiyacımız olan veriler verinin toplandığı anda verinin çıkartılan harita üzerinde hangi noktada toplandığı ve veriyi gönderen verilerden alınan sinyal şiddetidir. Bu iki veriye de sahip olunmuştur ve artık bu verilerden nasıl bir konum verisine sahip olduğuna bakılabilir.

Pİ yöntemlerinin uygulanabileceği onlarca sistem bulunmaktadır. Bu çalışmayı yaparken Bluetooth Low Energy (BLE) cihazları ile sistem inşa edilmiştir. BLE tercih edilmesinin nedeni bu cihazların düşük enerji tüketimi nedeni ile uzun süre ayakta kalabilmesi, maliyetinin düşük olması ve sistemimizi kapsayacak güçte sinyal gücüne sahip olmasıdır.

Pİ yöntemini uygulayabileceğimiz sistemler sadece bununla sınırlı değil tabiki. Bir zaman diliminde kullanıcının konumuna özel eşsiz bir veri üretebilecek herhangi bir cihaz ile de bu yöntem gerçekleştirilebilir. Örneğin sistemi Bluetooth cihazları ile oluşturmak yerine WiFi AP'leri ile de oluşturarak aynı şekilde verileri toplanabileceği

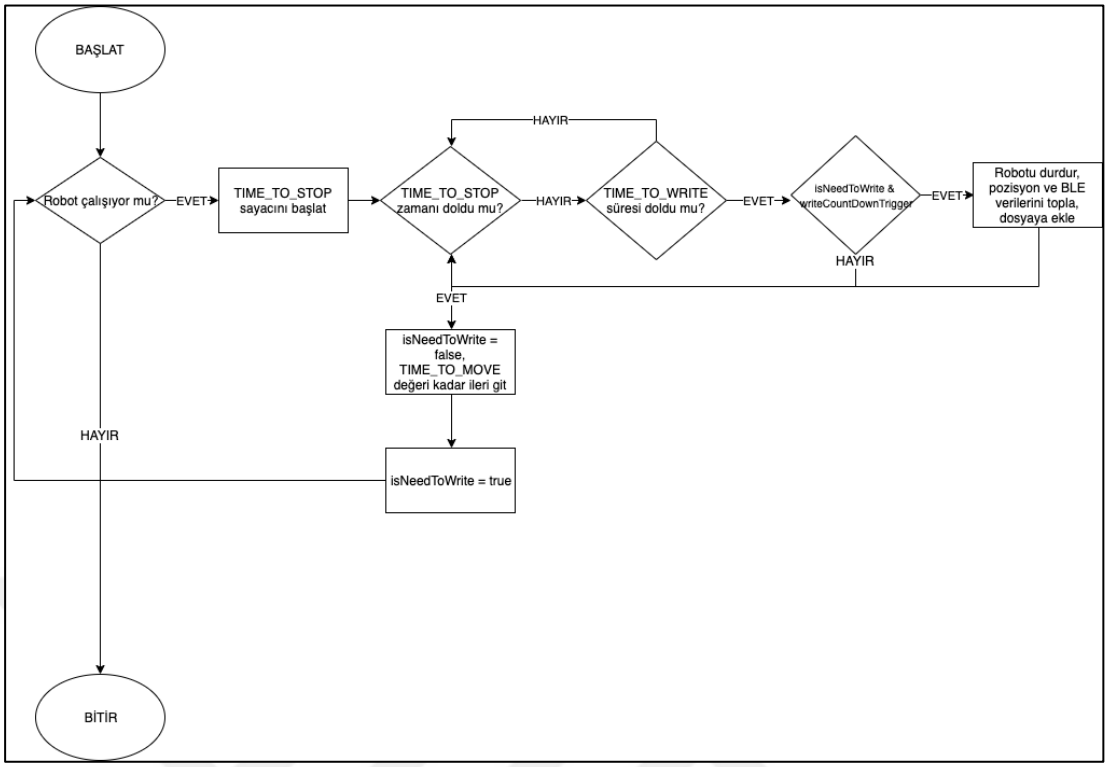
bir sistem inşa edilebilirdi. Fakat düşük enerji tüketimi, maliyeti, küçük alanlar kaplaması, kablosuz bir şekilde batarya ile birlikte oluşturulabilen ve uzun bir ömrü olmasından dolayı BLE cihazlarını bu çalışmada kullanmak için tercih edilmiştir.

Pİ yöntemleri sistemdeki daha önceki deneyimlerden yola çıkarak şu anki sistemde verileri kullanarak tahmin çıktısı veren bir yöntemdir. Bundan dolayı referans olarak kullanılan deneyimde örneklenmemiş bir konum verisi bulunması dahilinde sistem burada iyi bir tahmin çıktısı veremeyecektir ve bu referans noktasına en yakın olan tahmin çıktısını verecektir. Bu yüzden yöntemin ilk aşamasında sistemden ne kadar çok referans verisi toplanırsa sistem o kadar iyi çalışacaktır.

Pİ yönteminde sistemin koşullarının değişmesi de yöntemin çalışmasını kötü etkileyecek bir faktördür. Sistemin referans verileri toplanırken sistem statik ise ve yöntemin ikinci aşaması olan tahmin aşamasında aynı şekilde statik değil de dinamik bir sisteme dönüştüyse sistem, tahmin çıktılarının doğruluğunda düşme yaşaması beklenir.

Pİ yöntemi bahsettiğimiz gibi iki aşamadan oluşur. Bu aşamalardan ilki veri toplamanın yapıldığı aşama, ikinci aşama ise toplanan veriler üzerinden anlık verileri kullanarak tahmin çıkarımı yapılan aşamadır. Bu iki aşamanın gerçekleştirilmesi sonucunda Pİ yöntemi gerçeklştirilebilir hale gelmektedir. Yöntemin gerçekleştirilmesi için temel aşamalar bunlardır.

İlk aşamanın iyi bir yöntem ile gerçekleştirilmesi yöntemin iyi çalışması için çok önemli bir aşamadır. Bahsedilen iyi bir yöntem ise engellerin oluştuğu gürültü faktörlerini ortadan kaldırmak, stabil noktalardan veri toplamak, ortam koşullarının her nokta için aynı olması parametreleri ile açıklanabilir. İlk aşama verilerin toplanılmasını ve ikinci aşamada kullanılmasını sağlar. Buna göre ilk aşama Şekil 2.1'deki gibi gerçekleştirilmiştir. Veri toplama aşaması olan ilk aşamada haritadan veri toplamak için bir şema oluşturulmuştur. Bu şemaya göre belirlenen zaman aralıkları ile haritadan veri toplanması amaçlanmıştır. Parametre süre ile kısıtlı olmak zorunda değildir, süre parametresini uzaklık ile değiştirerek de belirlenen mesafe aralıklarında veri toplanması da aynı şema ile gerçekleştirilebilir. Değişken farklılıkları şemada yerine yerleştirilerek yeni bir şema oluşturulabilir. Bu şemaya uygun kod değişiklikleri ile de ilk aşama tamamlanır.



Şekil 2.1. Pİ veri toplama şematik gösterimi

Şekil 2.1'deki değişkenler:

TIME_TO_MOVE: Robotun veri toplarken kaç adım ileri gideceğinin zaman (milisaniye) türünden ifadesi.

TIME_TO_STOP: Robotun bir noktadan kaç milisaniye boyunca veri toplayacağı.

TIME_TO_WRITE: Robotun kaç milisaniyede bir kayıt ekleyeceğini belirten değişken.

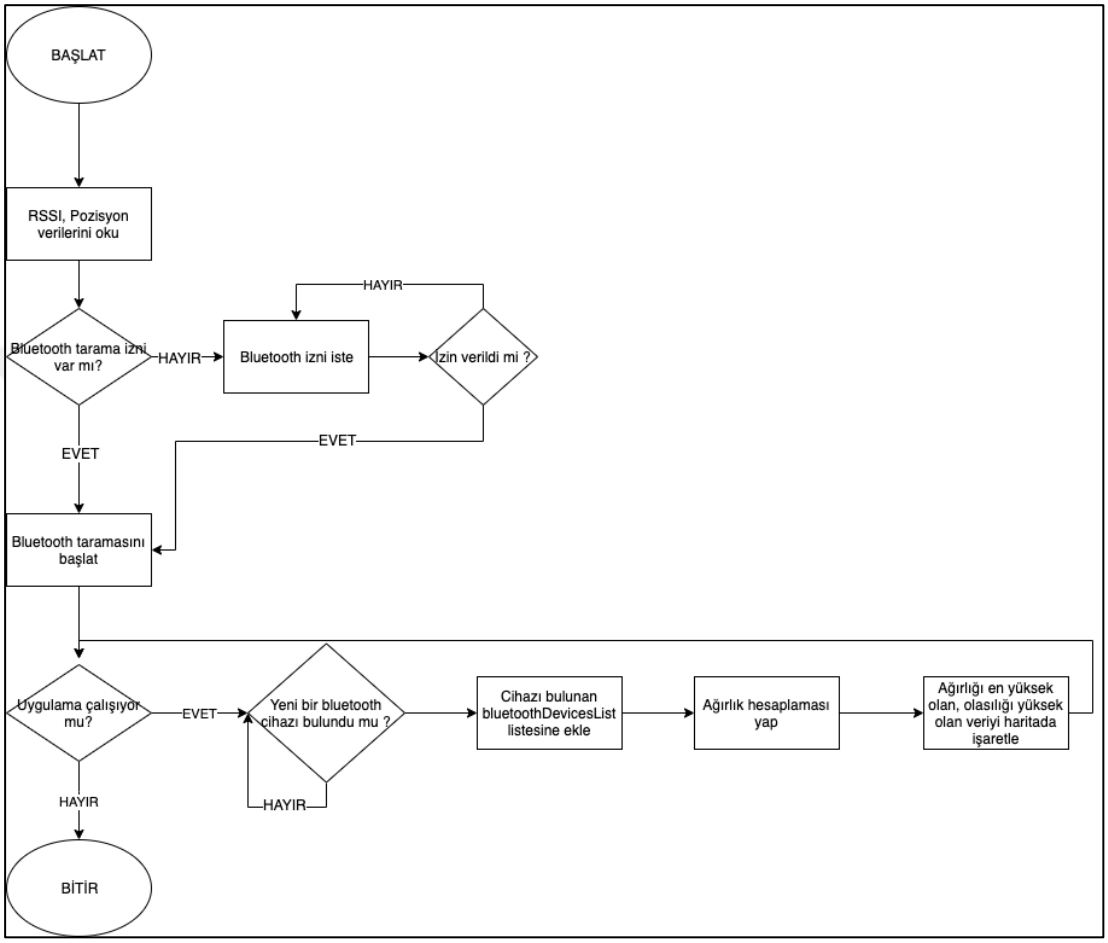
isNeedToWrite: Robotun hareket ettiği durumlarda verilerde bozukluk olmaması için veri kaydetme işleminin durdurulması için kullanılır.

writeCountDownTrigger: Robotun verilen zaman aralıklarında kayıt yapmasını sağlayan değişken.

Bu birinci aşamanın tamamlanması sonucunda robotun çıkardığı harita bilgileri ile birlikte sistemden toplanmış BLE RSSI ve pozisyon bilgilerine sahip olunur. Bu veriler yöntemin ikinci aşamasında kullanıcının konumunu tespit etmek için kullanılacak verilerdir. RSSI değerlerini kullanıcının aldığı RSSI değerleri ile

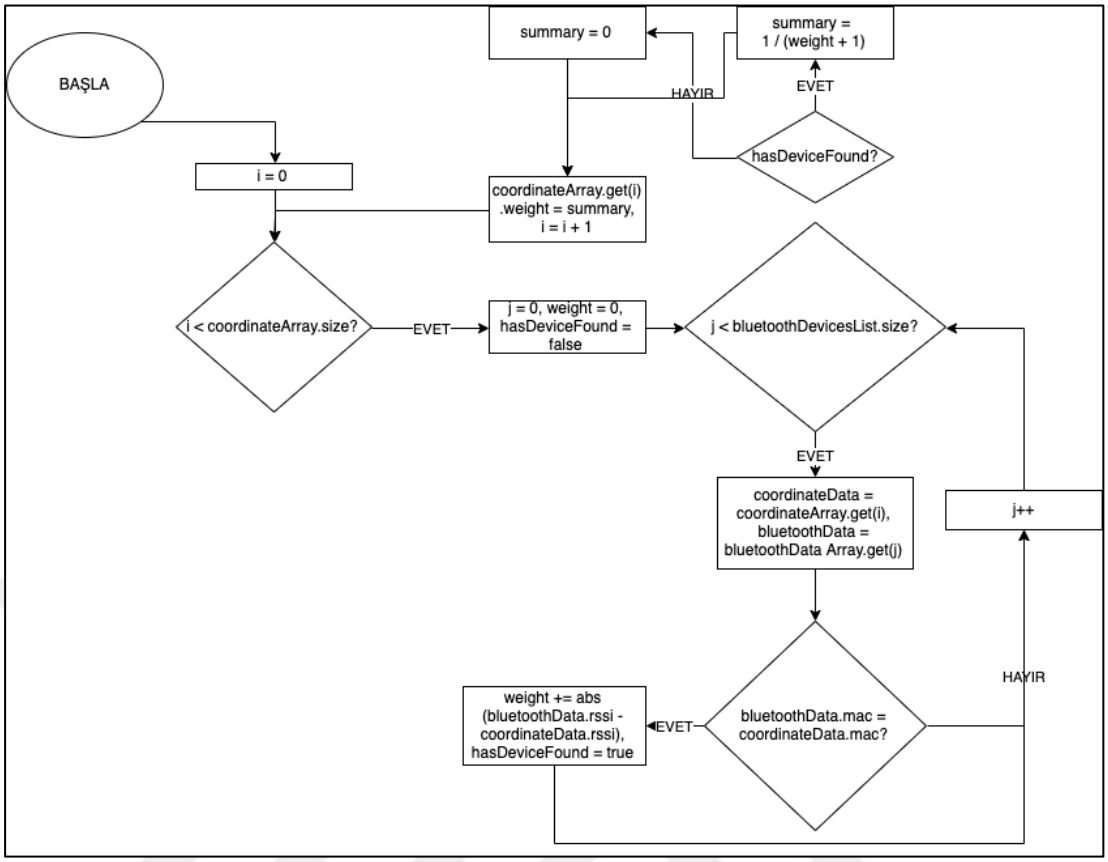
karşılaştırmak için kullanırken pozisyon verilerini de tespit edilen tahmine ilişkin pozisyonu gösterirken kullanılır.

Yöntemin ikinci aşaması Şekil 2.2 ile açıklanmıştır.



Şekil 2.2. Pİ uygun veri bulma şematik gösterimi

Şekil 2.2'de Pİ veri bulma gösterilmiştir. Yöntemin ikinci aşamasını Android işletim sistemine sahip bir cihazda çalıştırdığımız için öncelikle gerekli izinleri alarak uygulama çalışmaya başlıyor. Uygulama sürekli yeni taranan cihazları dinlemekte ve bir cihaz yakaladığında cihazın bilgilerini RSSI değeri ile birlikte bulunan cihazların arasına ekler, bu cihazlar listesi veritabanındaki veriler ile karşılaştırılarak bir ağırlık hesaplaması yapılır. Şekil 2.3'de ağırlık hesaplaması gösterilmiştir.



Şekil 2.3. Parmakizi ağırlık hesaplaması şeması

Şekil 2.3'de Pİ için ağırlık hesaplaması gösterilmiştir. Olabilitesi en yüksek olan veri haritada kullanıcının konumu olarak işaretlenir. İşlem bluetooth cihazları tarandıkça bulundukça tekrarlanır ve sürekli olarak kullanıcının konumu verilere göre güncellenir. Bu sayede kullanıcı haritada gezdikçe yeni konum tahminleri ile konumunu görebilecektir.

Bu işlem kullanıcı ekranda durdukça sürekli veriler geldikçe devam edecektir ve her veri değişikliği kullanıcıya yansıtılarak kullanıcıya harita üzerinde yeni tahminleri gösterilecektir.

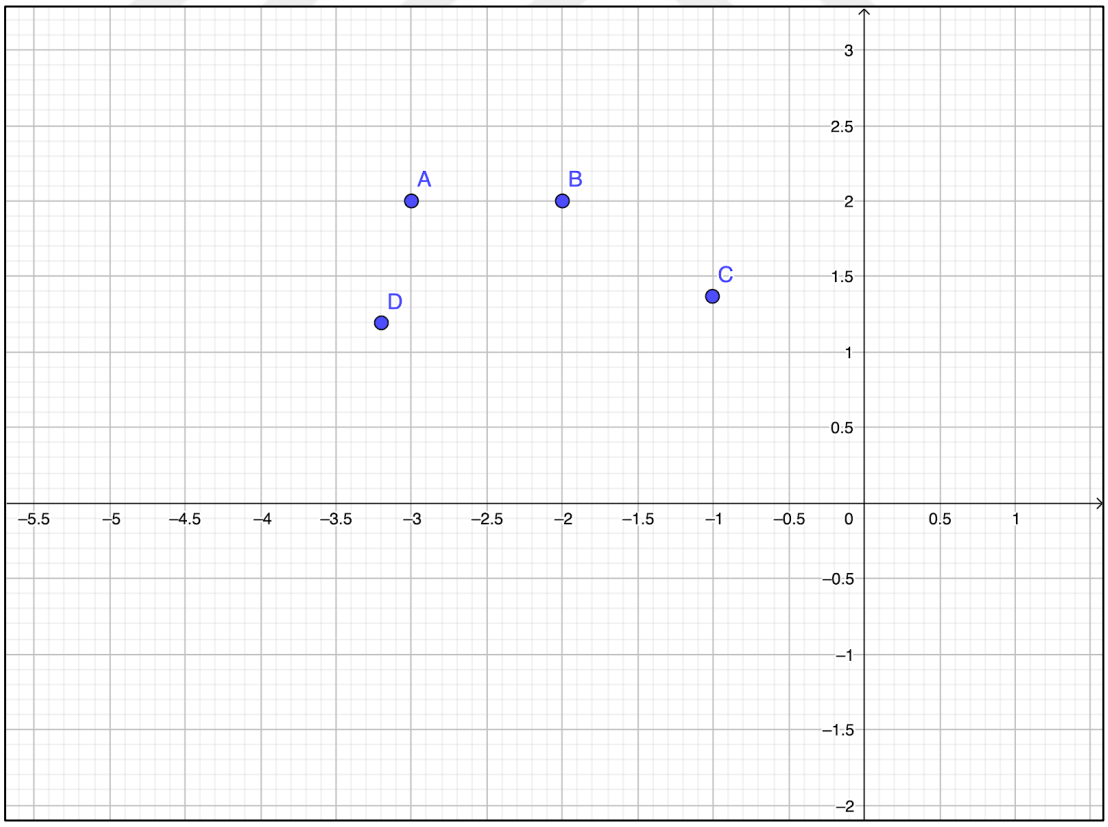
2.2.2. Üçgenleme (Tirlateration) yöntemi

Üçgenleme metodu yani literatürde bilinen adıyla Trilateration yöntemi RSS verisi ile konum tespit etme yöntemlerinden birisidir. Maliyetsiz bir yöntem olması ile tercih edilen yöntemlerden birisidir.

Üçgenleme metodu iç mekan konum tespit etmek için olasılıksal bir yöntemdir, hiç bir zaman kesin bir tahmin yapacağı düşünülmez; amaç kesine en yakın sonucu alabilmektir. Metodun performanslı çalışabilmesi için sistemin veri üzerindeki gürültüsünden olabildiğince etki göstermemesi sağlanması gerekmektedir veya gürültü faktörünü doğru hesaplayarak gürültünün etkisini olabildiğince azaltmak gerekmektedir.

Üçgenleme metodu için temelde gereken üç adet verici ve bir alıcıya ihtiyaç vardır. Yöntemin temel metodu vericilerden alınan sinyal seviyesi ile vericilerin alıcıya olan uzaklığını hesaplamak ve bu uzaklık verisi ile metodun kullanılarak olası alıcı konumunu tespit etmek üzerine kuruludur. Kullanıcının konumunu Trilateration metodu ile tespit edebilmek için vericilerin konumunu kullanıcının cihazına tanımlanmış olması gerekmektedir. Buna göre ölçümler yapılarak kullanıcının konumu Trilateration metodu ile tespit edilebilir.

Üçgenleme yöntemi sistem üzerinde Şekil 2.4'de gösterilmiştir.

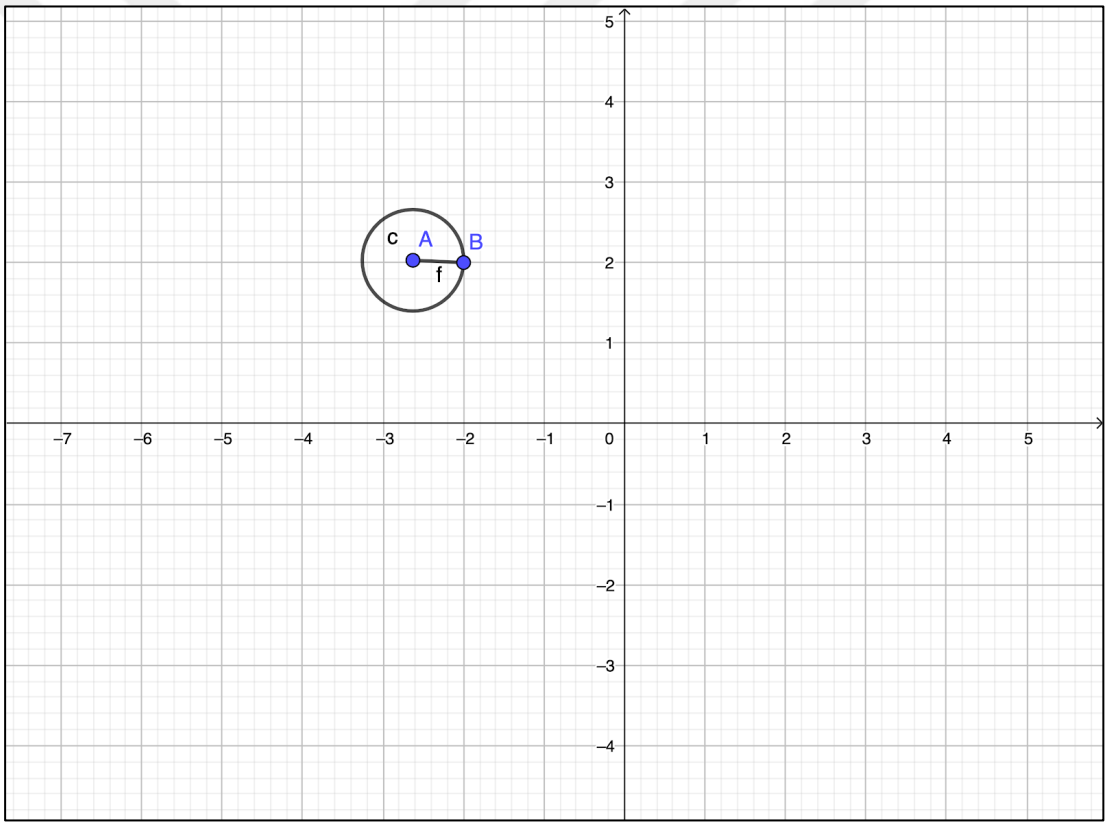


Şekil 2.4. Üçgenleme metodu uygulanacak sistem

Şekil 2.4'deki B noktasında bir kullanıcının mobil telefonunu alıcı olarak ve A, C, D noktaları verici olarak ele alınmıştır.

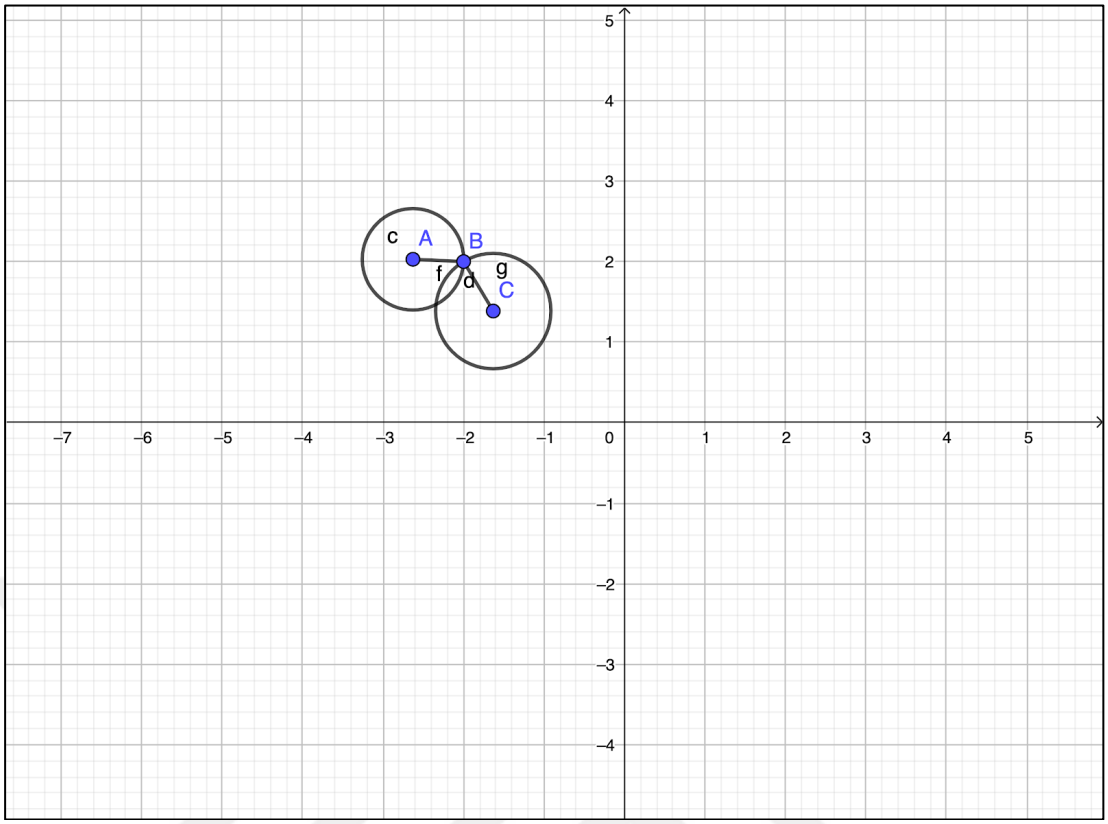
B noktasında bulunan kullancının referanslarına göre nerede olduğunu ifade eden verileri sistemden toplaması gerekmektedir.

Alıcı bulunduğu ortamdaki birinci alıcının sinyal seviyesini tespit eder, bu sinyal seviyesi (RSSI değeri) txPower (alınan sinyalin ne kadar güçlü olduğunu ifade eden değer) adı verilen bir değer ile $10^{(rssi/txPower)}$ formülünde yerine koyularak vericinin alıcıya uzaklığı söyler. Şekil 2.5'de ilk değerin bulunduğu konum ve alıcıya olan uzaklığı gösterilmiştir.



Şekil 2.5. Üçgenleme metodu ilk değer

Şekil 2.5'de alıcı, A vericisine göre (f) uzaklığında bir konumda olduğunu öğrenir. Fakat alıcı A çemberinin yayında bulunan herhangi bir noktada yer alıyor olabilir. Olasılıkları azaltmak için sıradaki vericiye göre olan uzaklığını ölçer. Şekil 2.6'da ikinci nokta için uzaklıklar gösterilmiştir.

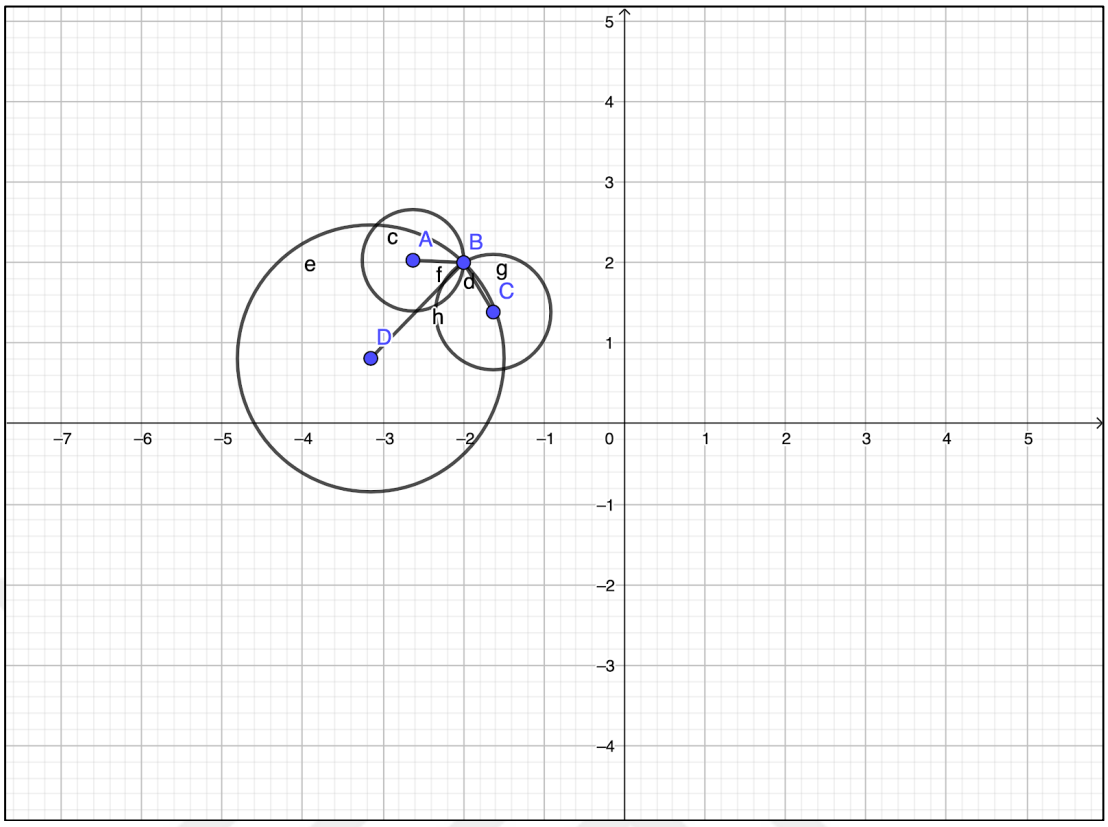


Şekil 2.6. Üçgenleme metodu ikinci değer

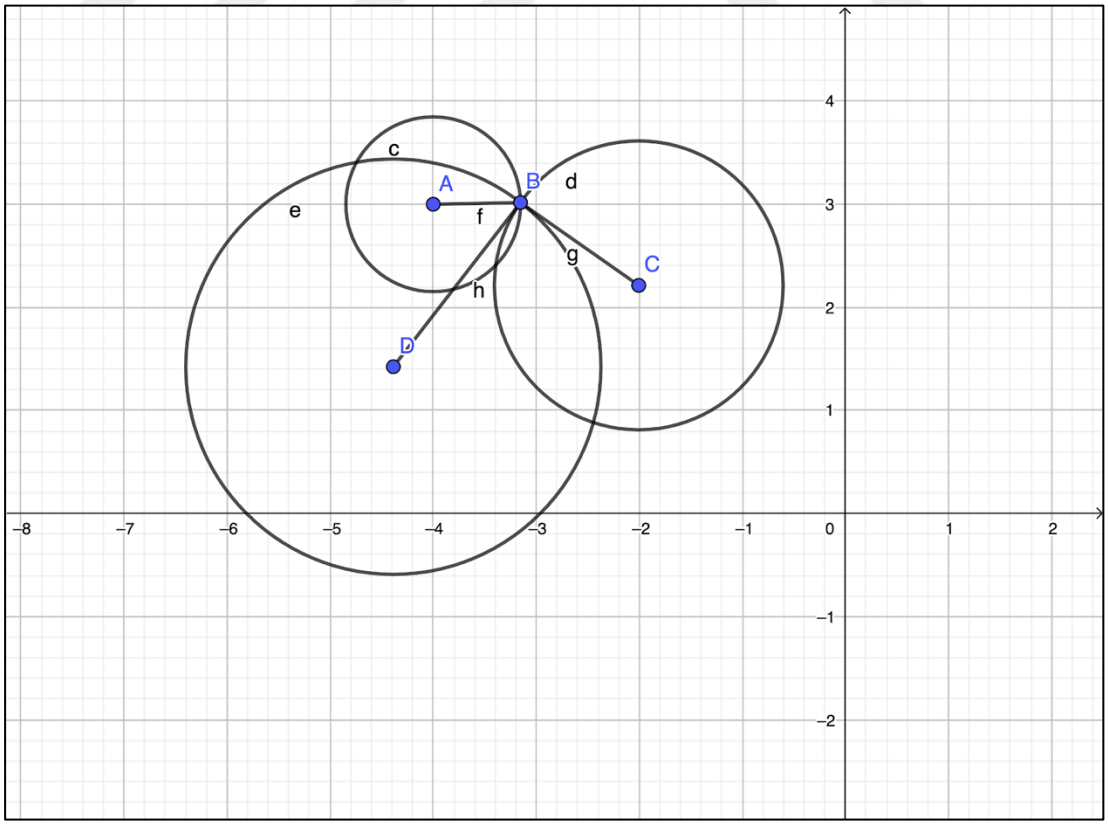
Şekil 2.6'da ikinci aşamada B noktasındaki alıcı C noktasına göre (d) uzaklığında bir konumda olduğunu öğrenir. A ve C vericilerine göre uzaklıklarını bilen B noktasındaki alıcı bu çemberlerin yaylarının kesiştiği iki noktadan birisinde olduğunu biliyor fakat hala tam olarak hangi noktada olduğunu bilmiyor. Bunun için üçüncü ve son verici olan C vericisine olan uzaklığını yine aynı formül ile ölçülür.

Şekil 2.7'de B alıcısına göre D vericisinin (h) uzaklığında olduğunu öğrenen alıcı artık bu üç çemberi kullanarak konumunu tespit edebilir. A vericisine göre f, C vericisine göre g ve D vericisine göre h uzaklığında olduğunu öğrenen alıcı bu üç çemberin kesiştiği tek nokta olan B noktasında olduğunu öğrenecektir.

Trilateration yöntemi, üç adet vericiden gelen sinyallere göre vericilerin alıcıya olan uzaklığı ile konum verisinin elde edilmesini sağlar. Şekil 2.7'de üçüncü nokta için değerler gösterilmişken Şekil 2.8'de bu sistemin şeması gösterilmiştir.



Şekil 2.7. Üçgenleme metodu üçüncü değer



Şekil 2.8. Sinyal gücü ile Trilateration yöntemi

Şekil 2.8'de bulunan B noktası alıcının bulunduğu konum; A, C, D noktaları ise vericilerin bulundukları konumlar ve f A'nın B'ye, g C'nin B'ye, h D'nin B'ye olan uzaklığı olmak üzere olmak üzere;

$A\ (X_a, Y_a)$

$B\ (X_b, Y_b)$

$C\ (X_c, Y_c)$

$D\ (X_d, Y_d)$ (2.1)

$f = \sqrt{((X_a - X_b)^2 + (Y_a - Y_b)^2)}$

$g = \sqrt{((X_c - X_b)^2 + (Y_c - Y_b)^2)}$

$h = \sqrt{((X_d - X_b)^2 + (Y_d - Y_b)^2)}$ (2.2)

$d = (nf / d) \times N$ d; f,g,h olmak üzere (2.3)

$X_{ij} = X_i - X_j$ (2.4)

$Y_{ij} = Y_i - Y_j$ (2.5)

$U_i = X_i^2 + Y_j^2 - da_i^2$ (2.6)

$X_b = (U_1Y_{32}+U_2Y_{13}+U_3Y_{21}) / (2(X_1Y_{32}+X_2Y_{13}+X_3Y_{21}))$ (2.7)

$Y_b = (U_1X_{32}+U_2X_{13}+U_3X_{21}) / (2(Y_1X_{32}+Y_2X_{13}+Y_3X_{21}))$ (2.8)

Yukarıdaki denklem ile B noktasında bulunan alıcının konum verisini A, C, D noktalarındaki vericiler vasıtasıyla Trilateration tekniği kullanılarak tahmini olarak sağlamaktadır. Bu teknik Trilateration tekniği olarak kullanmaktadır [30].

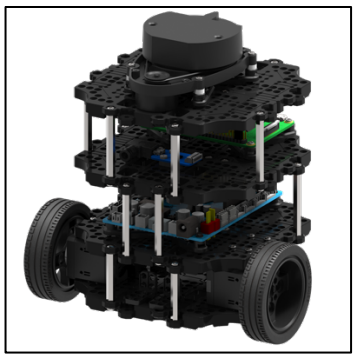
Denklem 2.1'de noktalar tanımlanmak üzere; f, g ve h uzaklıkları 2.2'deki denklem ile hesaplanmaktadır. Denklem 2.3'de ise nf gürültü faktörü (noise factor) ve N 'dir. Denklem 2.7 ve 2.8 ile X_b ve Y_b noktaları olmak üzere; 2.4, 2.5 ve 2.6 denklemleri 2.7 ve 2.8'de yerlerine yazılarak X_b ve Y_b noktaları bulunur.

3. İÇ MEKANLARDA YÜKSEK HASSASİYETLİ KONUM BELİRLEME İÇİN ROS TEMELLİ VERİ TOPLAMA

İç mekanlarda konum belirleme için iki aşamadan oluşan bir yöntem kurgulanmıştır. Yöntemin ilk aşaması veri toplama aşaması ve ikinci aşaması ise verileri işleyip konum tespiti yapma aşamasıdır. Yöntem ile ortamdan farklı farklı veriler toplayarak ortam hakkında eski yöntemlere göre daha fazla veri toplayarak zengin bir veri seti sunulmuştur. Bu veri setini oluştururken robot kullanılmıştır. Robot ile ortamın 2 boyutlu haritasını, ortamdaki vericilerin verilerini, vericilerin konumlarını ve bu veriler ile birlikte her veriye özel robotun 2 boyutlu harita üzerindeki konumunu kapsayan bir veri seti oluşturulmuştur.

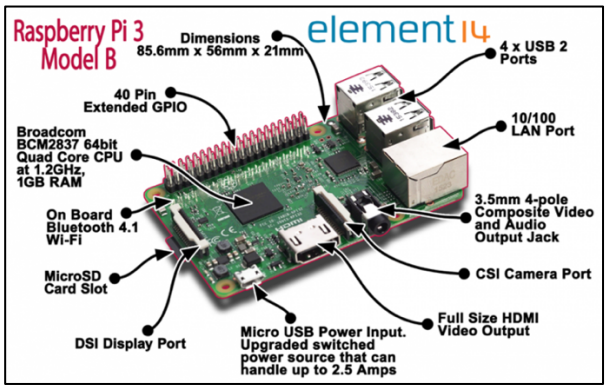
3.1. Kullanılan Donanımlar ve Teknolojiler

Çalışma kapsamında bir çok donanım ve yazılım kullanılmıştır. Çalışmayı gerçekleştirmek için Linux üzerinde çalışan ve robot için meta işletim sistemi görevini üstlenen ROS ile tam uyumlu çalışan Şekil 3.1'deki Turtlebot 3 Burger robotu kullanılmıştır.



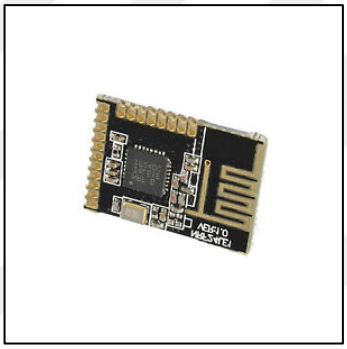
Şekil 3.1. Turtlebot 3 Burger

Bluetooth vericilerinden yayılan RSSI sinyallerini toplamak için ise Şekil 3.1'deki Turtlebot 3 Burger modelinin üzerindeki Rasperry Pi 3 üzerindeki Şekil 3.2'de gösterilen Bluetooth yongası kullanılmıştır.



Şekil 3.2. Raspberry Pi 3 ve donanım şeması

Ortamdan veri toplamak için yerleştirişlmiş Bluetooth vericileri için ise Şekil 3.3'deki NRF24LE1 entegresini kullanılmıştır.



Şekil 3.3. NRF24LE1 yongası

NRF24LE1 yongası 2,4 frekansında 0, -6, -12, -18dBm (TX output power) gönderici gücü ile 1,9 volt -3,6 volt (Tavsiye edilen 3,3 volt) güç kaynağı ile -40 - +80 ºC arasında çalışan System on Chip (8501 Intel) mikroçipine sahip bir yongadır.

Android cihaz ile verilerin toplanması ve toplanmış olan verilerin işlenmesi için ise Android 9.0 işletim sistemine sahip olan Şekil 3.4'de gösterilen OnePlus 5 model bir akıllı telefon kullanılmıştır.

OnePlus 5 cihazı bünyesinde Bluetooth 5.0 yongasını bulundurmaktadır. Aynı zamanda cihazın boyutlarının ve ağırlığının fazla olmaması robotun üzerine telefonu konumlandırırken kolaylık sağlamıştır. Android işletim sistemine uygun yazılmış olan yazılımlar bu cihaz üzerinde çalıştırılacaktır. Cihaz performansının üst seviyelerde olmasından dolayı da yazılımları üst seviye performans sağlayarak yazılımların görevlerini yapmasında büyük kolaylık sağlayacaktır.



Şekil 3.4. OnePlus 5 akıllı telefon

Geliştirilmiş olan yazılımlar için C++, Python, NodeJS, Java, Kotlin dillerini kullandık. C++, Python ve NodeJS dilleri robot için geliştirdiğimiz hareket etme, veri toplama, veri seti oluşturma, teleoperation (robot kontrolü) ve işaretleme yapma yazılımlarını geliştirmede kullanılmıştır. Java ve Kotlin ise Android için geliştirdiğimiz veri toplama, görselleştirme, konum tespiti, eski yöntemle konum tespiti yazılımlarını geliştirilmesinde kullanılmıştır.

Robot ile veri toplanırken noble çatısı bize Bluetooth verilerini toplamayı sağlanırken Android tarafında ise robottan gelen verileri işlenirken RosJava adlı çatı kullanılmıştır.

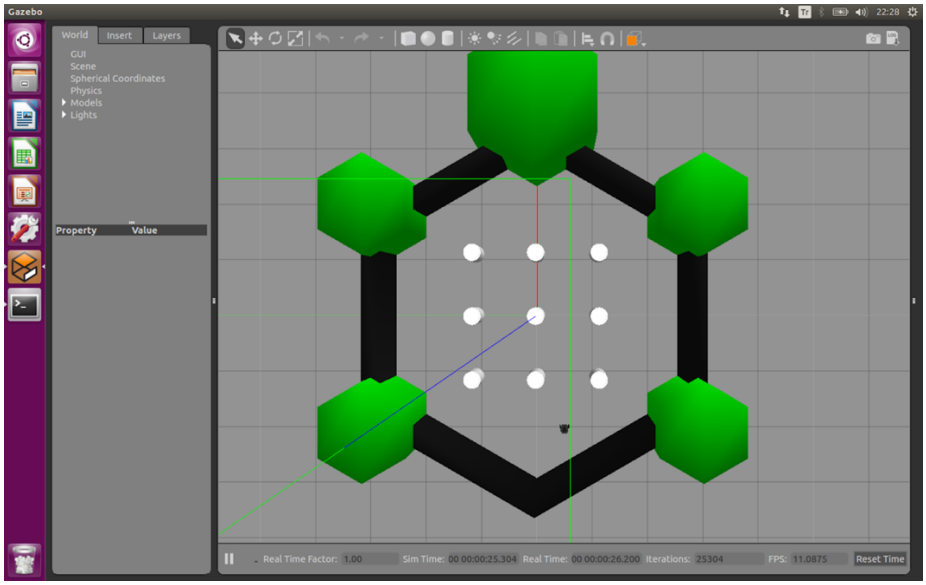
3.2. Robot İle Verilerin Toplanması

Robot ile verilerin toplanması aşamasında iki farklı veri toplanmıştır. İlk veri robotun LIDAR sensöründen gelen verileri işleyerek ortamın haritası olarak sunduğu harita verisi iken, ikinci veri ise parmak izi yöntemi için haritadan topladığımız pozisyon ve Bluetooth sinyal verileridir.

Robotun haritalama yapma aşamasında literatürde yer alan SLAM problemi çözümlerinden olan gmapping yöntemi kullanılmıştır. SLAM problemi robotun ortamın haritasını çıkartırken eş zamanlı olarak konumunu tespit edebilmesine dayanan bir problemdir. Bu iki işlemin birlikte yapılmasının temel nedeni sensörden toplanan verinin oluşturulma aşamasında olan haritanın neresine yerleştirileceğidir. Problemi daha iyi anlayabilmek için şu örneği verebiliriz; bir ofis ortamını düşünelim, bu ortamın haritasını oluşturmak istediğimizde elimize sürekli parça parça harita şekilleri gelmektedir, fakat bunların bir parçanın devamı olduğunu bulabilmek

imkansıza yakın olacaktır çünkü harita üzerindeki şekilleri bilmiyoruz. Ama burada bir jokerimiz var gelen parçanın puzzle şekilleri gibi anahtar noktalarını oluşturacak bir joker bu, tek yapmamız gereken sadece parçayı doğru anahtar parçaya yerleştirmek olacaktır. Bu noktada puzzle şekilleri gibi anahtar - kilit modelini sunan çözüm ise robotun başlangıç konumuna göre odometri verisini de sunmasıdır. Odometri verisi ise robotun hangi yönde ne kadar yol aldığını sağlayacak verilerdir. Yani robotun harita üzerinde konumunu kestirebilmesinden dolayı yeni gelen harita verisinin haritanın hangi noktasından hangi yöne doğru bakarak toplandığını sunmaktadır, bu şekilde iterasyonların devam etmesi sonucunda harita oluşmaktadır ve SLAM problemi çözülmektedir.

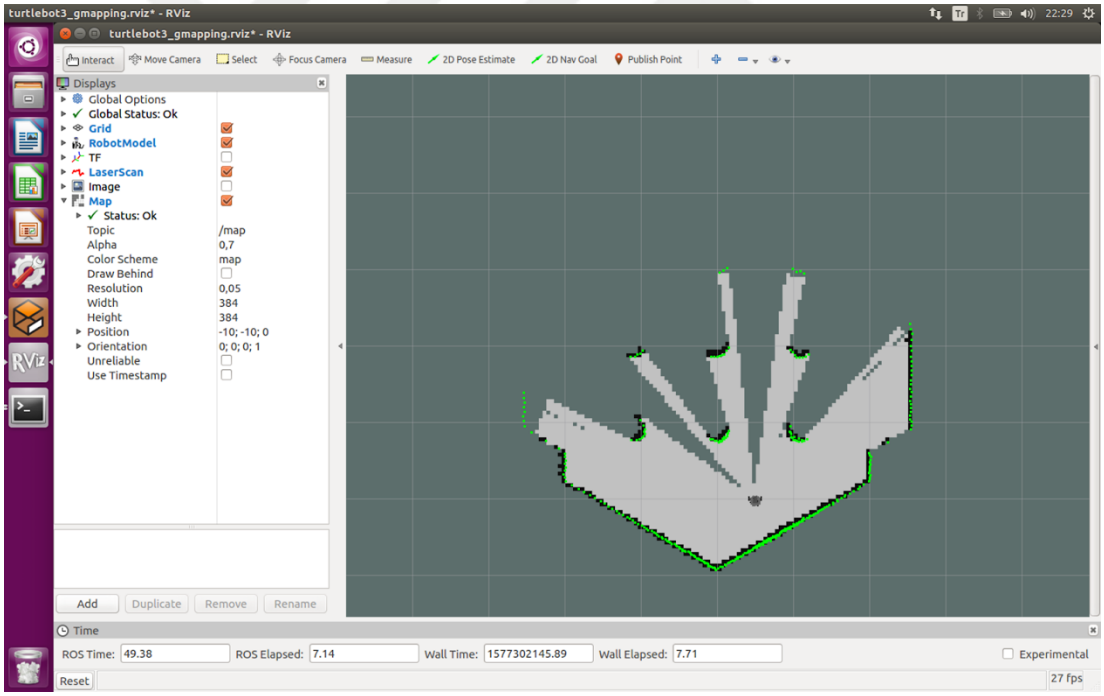
SLAM probleminin çözümlerinden biri olan gmapping ile robot ile harita verilerinin toplanması ve harita verisinin elde edilmesi sağlanmıştır. Haritalama sonucunda haritanın verilerinin bulunduğu bir adet .pgm (Portable Gray Map Image) dosyası ve harita özelliklerinin bulunduğu bir adet .yaml dosyası ortaya çıkmaktadır. PGM dosyasında haritanın her pixel bilgisi tek tek bulunurken YAML dosyasında ise bu haritanın adı, çözünürlüğü, orjin noktası (haritanın en sol alt köşesine denk gelen nokta) gibi bilgileri bulunmaktadır. Şekil 3.5'de bu verileri toplamak üzere kuallanılmış olan simülasyon ortamı gösterilmiştir.



Şekil 3.5. Robot ile haritalaması yapılacak olan simülasyon ortamı

Verilerin toplanması için her seferinde robotu kurmakla oluşacak zaman kaybını engellemek için ve algoritmayı daha hızlı geliştirebilmek için Şekil 3.5'deki Gazebo

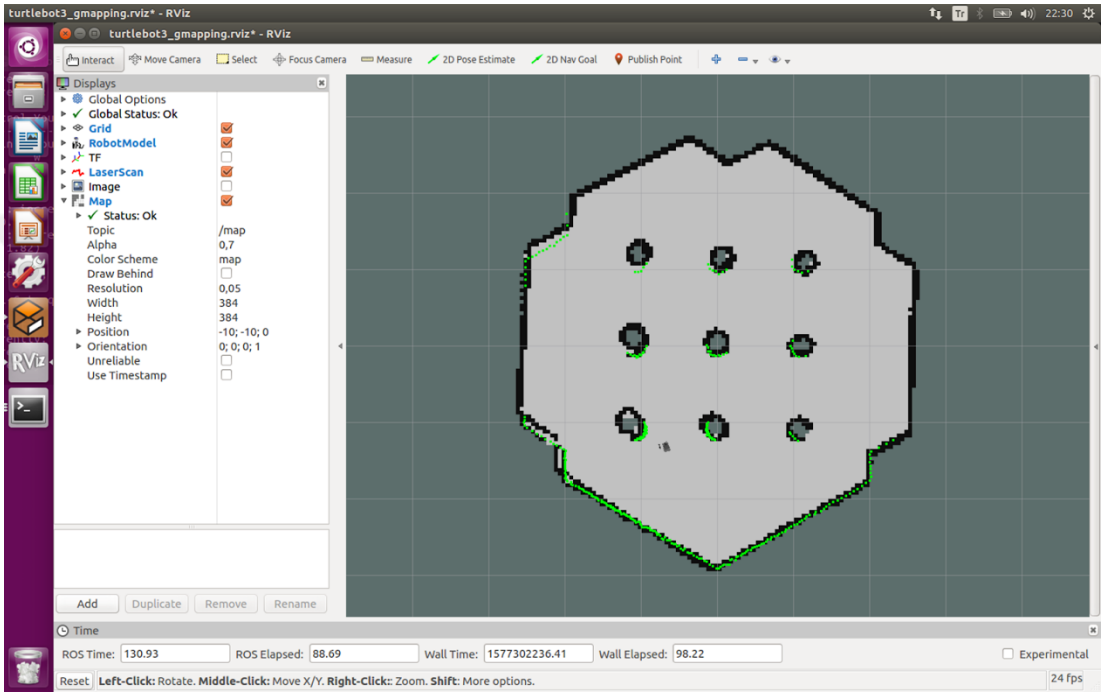
simülasyon ortamından yararlanılmıştır. Gazebo projesi 2002 yılında California'da Dr. Andrew Howard ve öğrencisi Nate Koenig tarafından başlatılmıştır. Linux ortamında geliştirilmiştir. 2009 yılında ise Jhon Hsu tarafından ROS ve PR2 Gazeboya entegre edilmiştir. Bu entegre sürecinden sonra Gazebo gittikçe tercih edilen bir simülatör olma özelliğini kazanmıştır. Apache 2.0 lisansıyla lisanslanmıştır. Hem kapalı hem açık mekanlar için geliştirilmiş açık kaynak kodlu, ücretsiz bir robot deney ortamıdır. Birden fazla robotu 3D ortamda simüle edebilme özelliğine sahiptir. ROS entegrasyonun olması ve ROS içerisinde gömülü olması sebebiyle Gazebo yazılım geliştiricileri için sıklıkla tercih edilen bir simülasyondur. Kullandığımız simülasyon ortamı ise Turtlebot 3 için hazırlanmış ve Turtlebot 3 kurulum aşamasında elde etmiş olunduğu simülasyon ortamıdır. Bu yüzden burada çalışacak olan algoritmanın aynı şekilde robotta da çalışması sağlanmıştır. Şekil 3.6'da RViz ile görselleştirilmiş olan haritalama işlemi gösterilmiştir.



Şekil 3.6. Robot ile ortamın haritasının oluşturulması

Simülasyon ortamını başlattıktan sonra robotun haritalama işlemini başlatması için roslanch komutu ile turtlebot3_slam paketi altında bulunan yine Turtlebot 3 ile birlikte gelen robot üzerinde çalışabilir durumda olan turtlebot3_slam başlatma dosyasını çalıştırma komutu verilerek haritalama işlemi Şekil 3.6'daki gibi başlatılır. Haritalama işleminde adım adım haritayı görüntülenebilmesi için RViz yazılımından destek

alınmıştır. RViz yazılımı ROS için 3 boyutlu görselleştirme sağlayan bir yazılımdır. Temel çalışma prensibi ROS topic (ROS konuları)'ndan gelen verileri işleyerek kullanıcıya sunmasıdır. RViz yardımı ile oluşturulmuş olan haritanın hangi aşamada olduğunu görülebilmekte ve haritanın tamamlandığı kullanıcı tespit edildiğinde haritanın kaydedilmesi gerektiği kararı RViz üzerinden görülerek verilmektedir. Şekil 3.7'de işlemin tamamlanması gösterilmiştir.



Şekil 3.7. Robot ile ortamın haritasının oluşturma işleminin tamamlanması

Harita üzerinde robotu gezdirerek topladığımız verilerden oluşturulmuş haritayı RViz vasıtasıyla Şekil 3.7'deki gibi gördük ve haritalamanın tamamlandığını gördüğümüzde yine bir ROS paketi olan map_server paketi altındaki map_saver isimli çalıştırılabilir dosyayı çalıştırma komutunu rosrun komutu ile birlikte çalıştırarak toplanmış olan haritayı belirtilen yere kaydedilmesi başarılmıştır.

Komutun örneği: rosrun map_server map_saver -f Desktop/simulation_test

Haritalama ve konum tespiti gmapping ile eş zamanlı olarak yapılırken aynı zamanda parmak izi yönteminin gerçeklenebilmesi için bu harita üzerinden toplanmış Bluetooth parkmakizlerine ihtiyacımız vardı. Bu verileri toplayabilmek için node.js için yazılmış olan bir kütüphane olan Bluetooth verisinin toplanmasını sağlayan noble adlı kütüphane kullanılmıştır. Bu kütüphane Turtlebot 3 Burger seti üzerindeki Raspberry

Pi 3 B+ modelinin sahip olduğu Bluetooth yongasına gelen sinyallerin dinlenilebilmesini sağlamıştır. Robot haritanın bir noktasında durduğunda bu verileri dinlemeye başlanmıştır, robot harita üzerindeki konum verisini odometri ile verirken noble vasıtasıyla toplanılan Bluetooth verilerini birleştirerek parmak izi yöntemi için gerekli olan veriler toplanmıştır. Bu verileri haritalama işlemi birlikte yaparak haritalamanın bitmesi sonucunda harita verilerinin yanında bir de bu parmak izi verilerin bulunduğu bir CSV formatlı veri elde edilmiştir. Tablo 3.1'deki verilerin her biri bir noktadaki parmak izleridir, bu parmak izleri kullanılarak Android geliştirme ortamında parmak izi yöntemi geliştirilmiştir.

Tablo 3.1. Robot ile toplanılan veri şeması

Step	**X**	**Y**	**Z**	**Name**	**Mac**	**RSSI**	**Clock**	**Day**	**Month**	**Year**
0	0;-1,886740587	0,893062199	0,010000000	undefined	2d:0d:ee:51:e4:2f	-88	22:31:55	9	2	2020
0	0;-1,8867405	0,893062199	0,010000000	BT05	5c:31:3e:5b:d7:80	-63	22:31:55	9	2	2020
0	0;-1,8867405	0,893062199	0,010000000	undefined	5e:4e:4b:c6:51:87	-71	22:31:55	9	2	2020
0	0;-1,8867405	0,893062199	0,010000000	JDY-08	88:4a:ea:7e:83:3c	-76	22:31:55	9	2	2020
0	0;-1,8867405	0,893062199	0,010000000	BT05	5c:31:3e:5b:d7:80	-60	22:31:55	9	2	2020

Step sütunu örneğin toplanıldığı adımı; x,y ve z sütünları konumun eksenler üzerinde bulunduğu noktayı; mac sütunu Bluetooh vericisinin adresini; rssi sütunu vericinin sinyal gücünü; clock sütunu verinin toplandığı zamanı; day, month ve year sütunları ise verinin toplandığı tarih bilgisini göstermektedir.

Robottan toplanan parmak izi verilerinin yanı sıra yöntemin gerçekleneceği cihazda daha hassas konum tespiti yapılabilmesi için Android cihaz üzerinde verilerin toplanabileceği bir uygulama geliştirilmiştir. Uygulama aynı robot ile aynı veri tipinde aynı verileri robotun "/odom" adlı topic (konu)'ini dinleyerek cihaz içerisinde bir dosyaya kaydeder. Bu veriler de parmak izi yöntemi gerçeklenirken karşılaştırma yapmak için kullanılmıştır.

Robot ile verilerin toplanmasını sağlamak için ROS ile birden fazla yöntem kullanarak en uygun yöntem araştırılmıştır. Bu yöntemlerde ROS'un topic (konu), node (düğüm) ve service (servis) yapıları kullanılmıştır.

Konular düğümlerin birbirleriyle olan mesaj alışverişleri için kullanılırlar. Konular kimin o konuya üye olduklarını bilmezler ve bununla ilgilenmezler. İlgili servisler ihtiyacı olan konulara üye olurlar ve ihtiyaçları olan bilgileri elde ederler. Konular tek yönlü iletişim sağlarlar [31].

Servisler bir dizi işlemi yapan ve bu işlem sonucunda bir sonuç döndüren mekanizmalardır. Yayınlayıcı / abone (publisher / subscriber) çok yaygın bir iletişim yöntemidir fakat dağıtık bir sistemdeki RPC istek / cevap yapısı için çoktan çoğa tek yönlü taşıma uygun değildir. Servisin çalıştığı düğüm bir servisi programcıya sunar ve yapılan çağrı ile servis çalıştırılarak bir işlem servis tarafında yapılır ve isteği yapan düğüm programcıya beklediği cevabı sunar [31]. Servislerin çalıştığı düğümler farklı bilgisayarlarda çalıştırılarak yoğun işlem gücü gerektiren işlemler yüksek performanslı bir bilgisayarda çalıştırılarak performans artışı elde edilebilir.

ROS düğümleri hesaplama yapan işlemlerdir. Düğümler Topic, RPC Servisleri ve Parametre Sunucusu kullanılarak birbirleri ile iletişim kurarlar [29]. Robotun çalışması esnasında birden fazla düğüm çalışır ve bu düğümleri hepsi farklı işlemler yaparken bazıları birbirleriyle de haberleşirler. Örneğin bir düğüm robotu hareket ettirirken diğer düğüm robot üzerindeki kamera ile çevrede nesne algılaması yapabilir ve aranan bir nesneyi bulduğunda bir konu üzerinden yayın yaparak o konuya abone bulunan hareket düğümüne haber verir ve bu düğüm de robotun durmasını sağlayabilir.

ROS bünyesinde işletim sistemine kazandırılan komutlar vardır. Bu komutlardan bazıları şu şekildedir: roscd (ros paketleri arasında gezinirken kolayca paketleri gezmemizi sağlar), rosls (ros paketlerini listeler), rostopic (ros konuları ile ilgili bilgi verir), rosnode (ros düğümleri ile ilgili bilgi verir), rosmsg (ros mesajları ile ilgili bilgi verir) vb. gibi komutlar bazı kolaylıklar sağlarlar. Aynı zamanda paketlerdeki çalıştırılabilir dosyaları robot üzerinde çalıştıran komutlar da bulunmaktadır, diğer komutları bilmeden de geliştirme yapılabilir fakat bu komutlar robot üzerinde düğümleri çalıştırmak için kullanılacak olan komutlardır. Bu komutlar:

rosrun: paketlerin altındaki çalıştırılabilir düğüm dosyalarını çalıştırır.

roslaunch: paketlerin altındaki başlatma dosyalarını çalıştırır. Bu başlatma dosyaları genellikle birden fazla düğümün birlikte çalıştırıldığı ve ros parametrelerinin atandığı durumlarda kullanılır.

Bu tez kapsamında bu mekanizmaların hepsini kullanılarak birden fazla yöntem arasından bir yöntem seçilmiştir. Bu yöntemlerden ve neden seçilip veya seçilmediğinden bahsedilecektir. Robot ile yapmak istenilen iş:

1. Robotun belirli bir hareket yapması ve durması,
2. Robotun durduğu noktadan RSSI verilerini yakalaması,
3. Yakalanan verilerin kaydedilmesi,
4. Robotun tekrar hareket etmesi.

Bu döngü düğüm durdurulana kadar devam edecek bir döngüdür. Düğümün durması terminali yöneten kişiye bağlıdır, ortamdan istediği miktarda veriyi elde ettiğinde düğümü durdurabilir. Burada düğümü otomatik olarak durdurulmasını sağlayabilecek herhangi bir kondisyon bulunmadığı için bu iş terminale bırakılmıştır.

İlk yöntemde izlenilen yol;

1. Robot belirlenen süre ileri gider ve süre bitince durur.
2. Robot belirlenen süre boyunca ortamdan RSSI verilerini toplar.
3. Veri geldikçe robotun konumu ile birlikte gelen veri dosyaya kaydedilir.
4. Süre bittiğinde robot tekrar 1. adımdan devam eder.

Bu yöntemin tamamı bir servis üzerinde kodlanmıştır, herhangi bir düğüm haberleşmesi veya servis yapısı kullanılmamıştır. Bu yöntemde işlem sıralaması asenkron yapıda olduğu için sağlıklı bir veri seti sunmamıştır, dosyaya yazım yapılırken eş zamanlı olarak robotun hareket etmesi gibi istenmeyen durumların oluşmasından dolayı bu yöntemi kullanılmamıştır.

Servislerin senkron çalışma prensibi ile farklı bir yöntem denenmiştir. Bir düğümün servise gönderdiği isteğin cevabını almadan devam etmemesi mantığı ile ikinci yöntemdeki yol izlenmiştir:

1. Robot belirlenen süre ileri gider ve süre bitince durur.
2. Robot belirlenen süre boyunca ortamdan RSSI verilerini toplar.
3. Veri geldikçe robotun konumu ile birlikte gelen veri dosyaya yazma işlemi yapan bir servis aracılığı ile gelen veriyi dosyaya kaydeder.
4. Süre bittiğinde robot tekrar 1. adımdan devam eder.

Bu yöntemde ise her şey güzel çalışmasına rağmen robotun hareketini sağlayan ve aynı zamanda Bluetooth verilerini toplayan düğüm node.js çatısı ile kodlanıldığı için problem yaşanmıştır. Rosnodejs çatısının sunmuş olduğu konuyu dinlemeyi sağlayan yapı robotun konum verilerinde istenildiği gibi sonuç vermediği gözlenmiştir. Abone olunan "/odom" konusu robotun harita üzerindeki konum verisini yayınlamaktadır ve bu veri kullanılarak konum tespiti yapılmaktadır.

```
std_msgs/Header header
  uint32 seq
  time stamp
  string frame_id
string child_frame_id
geometry_msgs/PoseWithCovariance pose
  geometry_msgs/Pose pose
    geometry_msgs/Point position
      float64 x
      float64 y
      float64 z
    geometry_msgs/Quaternion orientation
      float64 x
      float64 y
      float64 z
      float64 w
  float64[36] covariance
geometry_msgs/TwistWithCovariance twist
  geometry_msgs/Twist twist
    geometry_msgs/Vector3 linear
      float64 x
      float64 y
      float64 z
    geometry_msgs/Vector3 angular
      float64 x
      float64 y
      float64 z
  float64[36] covariance
```

Şekil 3.8. nav_msgs/Odometry mesajının içeriği

"/odom" konusundan yayınlanan ve Şekil 3.8'deki Odometry tipinde yollanan verinin içeriğinde bulunan "child_frame_id" değerinin "base_footprint" olduğu durumlarda gelen veri robotun konum verisidir. Fakat bu veri Rosnodejs kaynaklı bir problemden dolayı abone olduğumuz dinleyici fonksiyona bu veri hiç gelememiştir. Bu yüzden sağlıklı veri toplanılamadığı için bir düğüm daha kodlanılarak robotun konum verisini

dinleyen C++ tabanlı bir düğüm yazılmıştır. Dosyaya yazma işlemi yapan düğüme bir servis daha eklenmiştir ve bu servis de konum verisini dinleyen düğümden gelen veriyi sürekli dinleyerek global değişken de tutarak robotun konumunu dinlemeyi sağlarken Bluetooth verisi dinleyen servise yeni bir veri geldiğinde son konum ile birlikte gelen veriyi dosyaya yazarak cevabını tamamlamaktadır. Üçüncü yöntem:

1. Başlatılan servis düğümü sürekli robotun konumunu dinler ve global değişkene kaydeder.
2. Robot belirlenen süre ileri gider ve süre bitince durur.
3. Robot belirlenen süre boyunca ortamdan RSSI verilerini toplar.
4. Veri geldikçe global değişkende tutulan robotun konumu ile birlikte gelen veri dosyaya yazma işlemi servis aracılığı ile gelen veriyi dosyaya kaydeder.
5. Süre bittiğinde robot tekrar 2. adımdan devam eder.

Buraya kadar izlenilen yöntemlerde robotun belirlenen süre hareket etmesi ve belirlenen bir süre boyunca veri toplaması sağlanmıştır. Fakat bu yöntem ile robotun aldığı yol miktarını sabit bir değer olarak ayarlayamadığını ve toplanan RSSI verilerin ise her noktada farklı miktarda olduğu gözlemlenmiştir. Bunun için robotun hareket miktarını saniye cinsinden değil metre cinsinden ayarlayan bir düğüm ve toplanan RSSI verilerinin ise belirlenen saniye içerisinde değil belirlenen miktarda olmasını sağlayan düğümler yazılmıştır.

"/odom" konusundan yayınlanan mesaj robotun uzaydaki konum ve hız tahminini programcıya sağlamaktadır. Bu konu üzerinden robotun harekete başladığı konum ile ilerlediğinde elde edilen konumu karşılaştırılarak belirlenen miktarda gitmesini sağlayan bir düğüm yazılmıştır. Bu yöntemdeki amaç Bluetooth verilerini toplayan metodu da servis düğümünün içine alarak burada yapmaktır. Dördüncü yöntemde izlenen yol:

1. Başlatılan servis düğümü sürekli robotun konumunu "/odom" konusu üzerinden dinler ve belirlenen ilerleme sağlandıktan sonra son konumunu "/catchOdom" adlı konu üzerinden yayınlar ve durur.
2. "/catchOdom" servisini başlatan düğüm veri geldiğinde tetiklenir ve RSSI taramasını başlatır.

3. RSSI verisini yakalayan düğüm dosyaya yazma servisine istek atar ve gönderdiği veriyi dosyaya yazdırır.
4. Belirlenen miktarda veri toplandığında "/catchOdom" servisi cevap döner ve robot tekrar 1. adımdan devam eder.

Bu yöntemde "/catchOdom" servisinin çalıştığı düğüm RSSI verilerini toplayan noble'ın da çalışabilmesi için node.js ile yazılmıştır ve ROS iletişimi de Rosnodejs ile sağlanmıştır. Bu yöntemde ise rosnodejs yapısında ROS'u uyutmak için bulunan sleep() metodu bulunmamasından dolayı yukarıda izlemek istenilen "/catchOdom" servisine istek atacak ve robot belirlenen miktarda veri toplanana kadar uyutulacak ve veri toplaması bittikten sonra servis cevabını dönecek ve robot kaldığı yerden devam edecek algoritmasının çalışmasını engelledi. Noble çalışma prensibinde yayıncı / abone prensibine dayanmaktadır ve bir veri yakalandığında tanımlanmış olan metoda düşer ve burada işlenir, buradaki işlemin asenkron sağlanmasından dolayı robotun uyutulması ve işlem bitene kadar hareket düğümüne cevap vermemesinin sağlanması gerekmektedir fakat rosnodejs çatısında bulunmayan sleep() metodunun bulunaması bu yöntemin de kullanılmasını engellemiştir. node.js kütüphanelerinden olan "system-sleep" kütüphanesi kullanılmıştır fakat bu da sistemi tamamen uyuttuğu için RSSI verilerinin de dinlenmesi engellemiştir.

Bu yöntemlerin başarısız olmasından dolayı beşinci ve çalışan yöntem kurgulanmıştır ve programlanmıştır.

Uygulanan yöntem:

1. Robot belirlenen mesafe boyunca ileri gider ve durur. Durduğu an "/isRobotMoving" adlı konuya false mesajı ile yayın yapar. Bu da robotun durduğu anlamına gelir. Bu işlem C++ ile yazılmış düğümde gerçekleşir ve düğümün tek işi robotun hareketini sağlamaktır.
2. "/isRobotMoving" adlı konuyu dinleyen node.js ile geliştirdiğimiz düğüm bu değişkenin false olması durumunda (yani robotun durduğu anlamına gelir) noble yardımıyla Bluetooth sinyallerini dinlemeye başlar. Aynı zamanda robotun harita üzerindeki konumu da "/odom" konusundan okunur ve Bluetooth verisi ile birleştirilir.

3. Yakalanan veriler “/write_file_node” adlı düğümden yayınlanan “/catchRSSI” adlı servise yollanır ve veriler dosyaya yazılır.
4. Belirlenen miktartda veri toplandıktan sonra node.js ile yazılmış olan düğüm “/isRobotMoving” adlı düğüme true değerini yollar.
5. Robotu kontrol eden “/positionUpdater” adlı düğüm ise bu değeri dinlemektedir ve değerin true olması durumunda robotu yine belirlenen mesafe boyunca ileri götürür.
6. Düğüm kapatılmadıysa tekrar 1. adımdan başlayarak devam eder.

Bu yöntemde sonuçta hedeflenilen doğrultuda çalışan bir yöntem elde edilmiştir. Bu yöntem sonucunda robot ile belirlenen bir mesafe robotun ileri götürerek ardından belirlenen miktarda veri toplaması ve ardından tekrar devam etmesi sağlanmıştır. Robotun stabil bir şekilde hareket etmesi, hareket ederken veri toplamaması ve durduğunda çevresel faktörlerden etkilenmeden stabil veriler toplaması sağlanmıştır.

3.2.1. Verici konumlarının toplanması

Harita üzerindeki vericilerin görselleştirilmesi ve Trilateration gibi yöntemlerin gerçeklenmesi için vericilerin oluşturulmuş harita üzerindeki konumlarının bilinmesi gerekmektedir.

Konumları işaretlemek için robotun hareketinin sağlanacağı bir düğüm gerekmektedir. Hareketi kullanıcı tarafından alıp robotu yönetecek olan bu düğüm bir tuşa basıldığında robotun olduğu konumda Bluetooth cihazına ait bilgileri toplayacak ve dosyaya kaydedecektir. Bu hareketi Turtlebot3 paketi ile birlikte gelen turtlebot3_teleoperation paketinin altındaki python ile kodlanmış olan turtlebot3_teleop_key adlı düğümün klonlanması ile sağlanmıştır. Bu düğüm wireless_localization adlı paketin altına taşınmıştır ve bu paket içerisinde python çalışması için gerekli bağımlılıkları projeye eklenmiştir. Klonlanmış olan bu düğümde w, a, s, d ve x tuşları sırasıyla ileri, sol, sağ, dur ve geri aksiyonlarını sağlamaktaydı. Bu düğüme “k” tuşunu da eklenerek ve k tuşuna basıldığında yazılmış olunan kayıt servisinin çalışması sağlanmıştır.

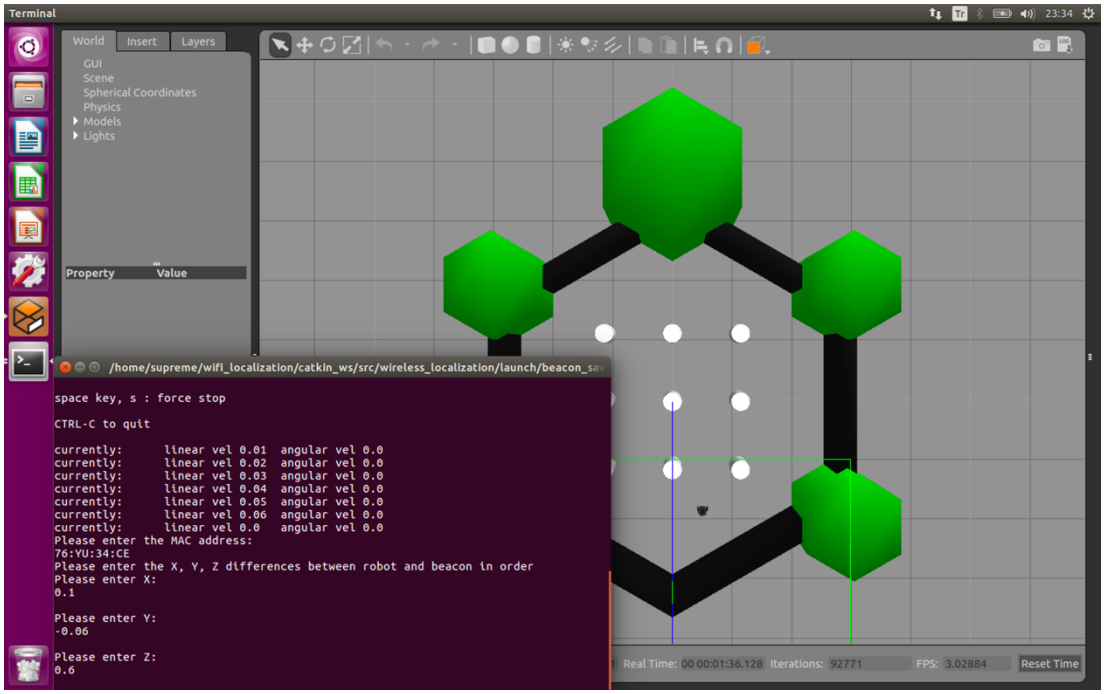
Beacon Catch diye adlandırdığımız servis Bluetooth cihaz bilgilerini kayıt etmek için tetiklenmeyi beklemektedir ve bu tetiklenme paketimizin altındaki

turtlebot3_teleop_key tarafından sağlanmaktadır. Düğüm tetiklendiğinde aşağıdaki bilgileri sırasıyla ister.

- Kayıt edilen cihazın MAC adresi
- Robotun Bluetooth cihazına olan X eksenindeki farkı (1)
- Robotun Bluetooth cihazına olan Y eksenindeki farkı (2)
- Robotun Bluetooth cihazına olan Z eksenindeki farkı (3)

1, 2 ve 3 ile belirtilen adımlarda beklenilen giriş için 1 metrelik fark 0.1 girişi ile tanımlanmaktadır.

Paketin altındaki turtlebot3_teleop_key düğümü ve beacon_catch adlı servisi birlikte çalıştırmak için bir başlatma dosyası yazılmıştır. Wireless Localization paketinin altında bulunan "beacon_save.launch" adlı başlatma dosyası roslaunch komutu ile çalıştırıldığında tek terminalde hareket düğümü ve verici yakalama düğümü eş zamanlı olarak çalışacaktır. Servis senkron çalıştığı için çağrıldığında turtlebot3_teleop_key adlı düğümün işini bitene kadar kesecektir ve bununla birlikte verici bilgilerini girerken basılan w, a, s, d, x ve k tuşları robotun hareketini etkilemeyecektir.



Şekil 3.9. beacon_save adlı başlatma dosyasının çalıştırılması

Simülasyon ortamında Şekil 3.9'da belirtilen konuma göre aşağıdaki girdiler ile verici noktası işaretledik. Tablo 3.2'deki girdiler ile robot ile ortamdaki verici konumları toplanmıştır.

Tablo 3.2. Bluetooth konum girdileri

MAC	X ekseni farkı	Y ekseni farkı	Z ekseni farkı
5c:31:3e:5b:d7:80	0,01	-0,06	0,1
88:4a:ea:7e:83:3c	-0,01	0,01	0,1

Tablo 3.3'de ise Tablo 3.2'deki girdiler karşılığında dosyaya yazılan Bluetooth verici konumları gösterilmiştir.

Tablo 3.3. Bluetooth verici konumları

MAC	X	Y	Z
5c:31:3e:5b:d7:80	0,713228	-0,296619	0,1
88:4a:ea:7e:83:3c	4,24236	0,98733	0,1

Dosyaya yazma işlemini yaparken robotun o andaki konumu ile verilen X, Y ve Z ekseni bilgileri toplanarak vericinin konumu işaretlenir. Yani bilgilerin hepsinin 0.0 verilmesi durumunda robotun konumu vericinin konumu olacaktır.

3.2.2. Android cihaz ile verilerin toplanması

Bluetooth cihazlarında kullanılan donanımlara göre topladıkları RSSI verilerinde farklılıklar oluşmaktadır. Aynı noktadan iki farklı donanıma sahip olan mobil cihaz ile X vericisinden sinyal toplandığında mobil cihazların iki farklı veri toplaması beklenmektedir.

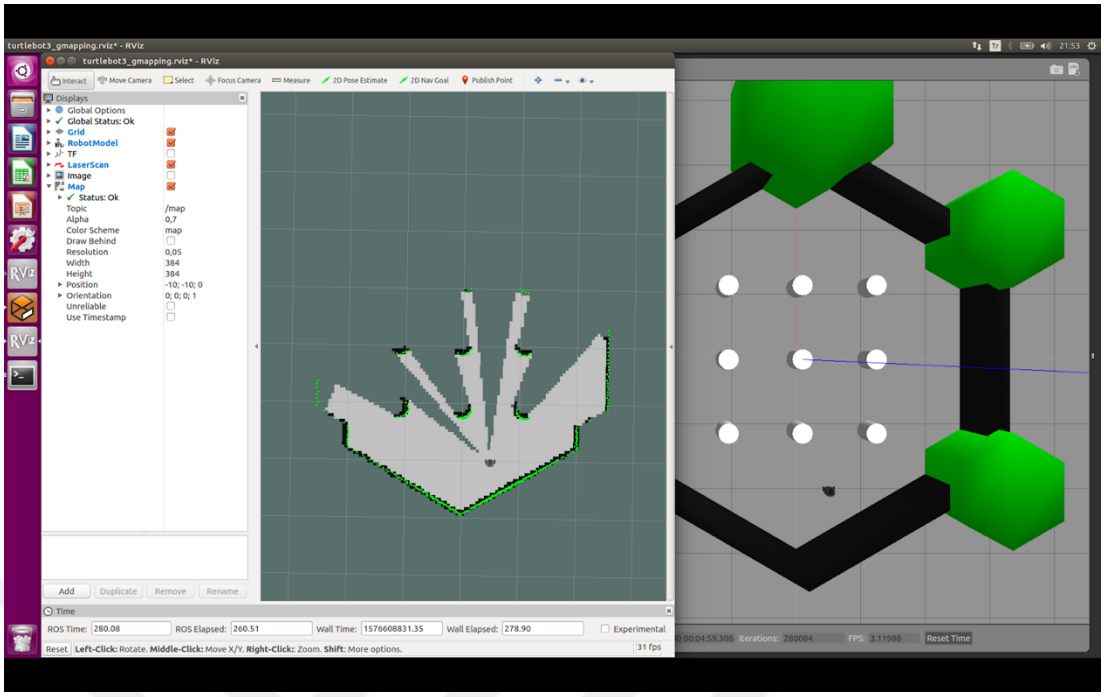
Cihaz farklılığından dolayı oluşan bu veri farklılığının oluşması zaten beklenmekte olan bir durumdur. Bundan dolayı konum tespitinin yapılacağı mobil cihazı da robotun üzerine yerleştirerek bu cihazdan da Bluetooth verilerini toplayarak oluşan iki farklı ile karşılaştırma yaparak, cihaza özgü veri toplayarak oluşan farklılığı gözlemlenecektir.

Android cihazdan veri toplamak için Java programlama dili için geliştirilmiş olan Rosjava kütüphanesi kullanılmıştır. Rosjava Google ve Willow Garage ile birlikte geliştirilen bir kütüphanedir. Rosjava ile ROS tamamen baştan Java ortamında geliştirilmiştir ve Google'un amacı Android ortamında ROS'un tüm fonksiyonları ile birlikte çalışmasıdır [29].

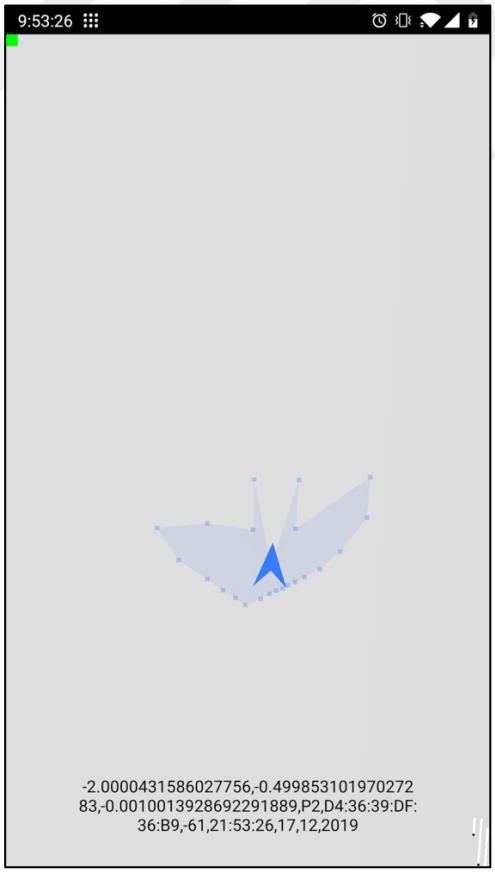
Bu Android uygulamasında iki özellik sunulmuştur. Uygulamada görsellik için kullanıcıya robotun oluşturduğu harita için topladığı LIDAR sensör verilerinden elde edilen "/scan" konusuna yapılan yayın ile elde edilen verileri Rosjava'nın sunduğu LaserScanLayer komponenti yardımıyla görsel veriye çevirerek uygulama arayüzünde sunulmuştur. Aynı zamanda "/map" konusundan yayınlanan arttırımlı şekilde oluşturulan harita verisini de yayınlandıkça yine Rosjava'nın bir komponenti olan OccupancyGridLayer yardımıyla görsel bir veriye dönüştürülerek kullanıcıya gösterilmiştir.

Bu veriler sadece görsel bir arayüz oluşturmak için kullanılmıştır. Uygulamanın asıl amacı ise "/odom" konusundan yayınlanan konum verisini yakalamasıdır. Konum verisinin ViewModel üzerinde tanımladığımız değişken üzerinde tuttuk. Her yenilemede bu veri Robot ile birlikte mobil uygulamada da güncellenmektedir. Bu işlem yapılırken aynı zamanda telefon üzerinde eş zamanlı olarak Bluetooth veri taraması da yapılmaktadır. Bu tarama sonucunda mobil cihaz sürekli ortamdaki Bluetooth vericilerini tarayarak veri tabanının oluşmasını sağlamıştır. Robot üzerinden yayınlanan konum verisi ve toplanan Bluetooth verilerinin birleştirilmesi ile robot ile toplanılan veri yapısının aynı yapısında bir veri yapısı oluşturulmuştur.

Şekil 3.10'daki görüntü görüldüğü şekilde oluşturulan uygulamanın test edilmesi için kullanılan simülasyon ortamına aittir. Burada Gazebo simülasyonunda ortam sağda bulunurken, sol tarafta ise bu simülasyon ortamından toplanan verilerin RViz ortamında görselleştirilmesi görülmektedir. RViz gelişmiş arayüzü ile toplanmakta olan ham verileri işleyerek anlamdırılmalarını sağlaması ile büyük kolaylık sağlamaktadır. RViz, robot ile eş zamanlı olarak çalışarak gerçek zamanlı veriler sunmaktadır. Özellikle haritalama aşamasında robotun haritada gezmesi ile birlikte 2-3 saniyelik yenileme periyoları ile oluşturulan haritayı ve bu harita üzerinde robotun konumunu göstererek kullanıcısına yardımcı olmaktadır.



Şekil 3.10. Veri toplama uygulaması için simülasyon ortamı ve RViz



Şekil 3.11. Veri toplama uygulamasının eş zamanlı ekran görüntüsü

Gazebo ortamındaki simülasyon ortamının Android veri toplama uygulamasındaki ekran görüntüsü ise Şekil 3.11'deki gibidir. Ekranda robotun "/map" ve "/scan" konularından gelen verilerin işlenmiş verisi ekranda gösterilmiştir. Aynı zamanda Bluetooth ile birlikte "/odom" verisinin toplanması ile oluşturulmuş veriyi ise anlık olarak kullanıcıya gösterilmiştir. Bu veri eş zamanlı olarak telefonun hafızasında oluşturduğu dosyaya kaydederek veri setini oluşturmaktadır. Sol üst köşede bulunan tanımlayıcı ise eğer gelen veri "base_footprint" child_frame_id'sinden gelirse kaydedildiğini göstererek yeşil olmaktadır, eğer başka bir frame_id'den gelirse kırmızı olarak yazılmadığını kullanıcıya belirtmektedir.

3.3. Verilerin İşlenmesi ve Görselleştirilmesi

Bundan önce anlatılan bölümlerde işlenmemiş veriler toplanmıştır. Bu verilerin kullanıcı için bir anlam ifade etmesi gerekmektedir. Bu anlamı verilere yükleyebilmek için verileri Android platformunda görselleştirme tercih edilmiştir. Burada Android bir cihaz üzerinde robottan toplanan harita verisini görselleştirerek kullanıcının anlayabileceği, kroki haritayı görselleştirilecektir. Bu görselleştirmenin ardından Pİ yöntemini Android uygulaması üzerinde gerçeklenerek konum tespiti metodu gerçekleştirilmiştir. Aynı zamanda Android uygulaması üzerinde Üçgenleme metodu da gösterilmiştir. Android uygulamasının yanı sıra robot için de bu toplanılan veriler anlamlı verilere dönüştürülerek robotun harita üzerinde konumunu bulmasına yardımcı olunmuştur, Pamakizi yöntemini robot için de gerçekleyip robotun harita üzerinde konum tespitini yapmasına yardımcı olunmuştur.

Toplanılar veriler iki farklı kaynaktan toplanmıştır. Robot üzerinde ortamın haritası toplanırken aynı zamanda ortamdaki Bluetooth verileri de toplanmıştır. Android cihaz üzerinde ise robottan gelen "/odom" verisi ile birlikte ortamdaki Bluetooth sinyalleri toplamıştır. Burada toplamda 2 farklı Bluetooth verisi setine sahip olduk, bir veri Android cihazımıza özel olarak toplanmışken diğer veri robotun yongasına özel toplanmıştır. İki farklı veri toplamamızdaki amaç Pamakizi metodunu Android cihaz üzerinde test ederken cihazın Bluetooth yongasından toplanmış olan veriden elde edilen konum bulma hassasiyet - performansını cihaz özelinde toplanmamış olan veri ile karşılaştırarak bir karşılaştırma yapabilmektir.

Bu bölümde kullanılmış veriler:

map.pgm : Harita verilerini sağlar

map.yaml : Harita verilerine dair bilgileri sağlar

dataset.csv : Ortamdan toplanmış olan Bluetooth verileri ve pozisyon verilerini sağlar

Bu veriler kullanılarak geliştirme ortamında kullanıcının anlayabileceği şekilde veriler işlenerek kullanıcıya sunulmuştur.

Android uygulaması üzerinden veri toplanırken robotun bulunduğu noktadan veri toplamasına ihtiyaç duyulduğu için kullanılan telefonu tarama sırasında Şekil 3.12'deki gibi robotun üzerine yerleştirilmiştir.



Şekil 3.12. Robot üzerine yerleştirilen Android telefon

Şekil 3.12'de görüldüğü şekilde monte edilmesinin sebebi cihazı robotun üstünde veya başka bir yerinde yerleştirebilecek bir bölme olmamasıdır. Robotun üstüne koyulmamasının nedeni ise LIDAR sensörü dönerek çalışmasıdır.

3.3.1. Verilerin robot ile işlenmesi

Robotun haritası bilinen bir ortamda konum tespiti yapmasını sağlayan yöntem Adaptive Monte Carlo (AMCL) yöntemidir. AMCL yöntemi ile girdi olarak bir harita ile başlayan yöntem sürekli lazer sensöründen gelen veriyi dinleyerek taramalar yapar ve topladığı veriyi elindeki haritadaki taramalar ile eşleştirerek konumunu tahmini olarak bulmaya çalışır. Bu şekilde robot harita üzerinde gezindikçe robot etrafından daha çok veri örneği toplar ve harita ile karşılaştırarak konumunu iteratif bir şekilde daha iyi tespit edebilmeye başlar. Bunu ise taramalara göre oluşturduğu harita üzerinde oluşturduğu konumlara göre sağlar. Harita üzerindeki her bir konum verisinin ağırlığı bulunmaktadır ve bu ağırlıklar robotun o noktada olabileceğine dair ağırlık verisidir. Konumun taramalar ile eşleşme oranına göre ağırlığı artar ve robot kendini bu olasılıksal noktalardan en yüksek ağırlıklı olanında gibi farzeder ve veri geldikçe – yani robot haritada gezdikçe – robot bu olasılıkların çoğunu eler ve gerçeğe en yakın olduğunu düşündüğü konumdaymış gibi kendisini konumlandırır.

Bu çalışmada haritadan toplanılan veriler ile robotun kendi konumunu tespit ederken daha hızlı tespit edebilmesini sağlamak için toplanılan veriler ile Parmakizi tekniğini robot üzerinde de uygulanmıştır. Topladığımız verilerden robot için anlam ifade edenler map.pgm ve map.yaml dosyasında tutulan verilerdir. Robot için bu veriler bir haritayı ifade etmektedir ve robot bu haritayı üzerinden konum tespiti yapabilmek, navigasyon hedefi belirmek gibi farklı farklı işler için kullanabilir. Fakat bu işlemleri yapabilmesi için öncelikle harita üzerinden kendi konumunu tespit edebilmelidir. Toplanmış olunan dataset.csv dosyası yalın hali ile robota herhangi birşey ifade etmemektedir.

Parmakizi metoduna yazmış olduğumuz düğüm dataset.csv dosyasını okuyarak robotun anlayabileceği verilere dönüştürmektedir. Burada düğümü yine noble'ı kullanabilmek için node.js ile kodlanmıştır. Düğümün birden fazla görevi bulunmaktadır. Bu düğümün yaptığı iş:

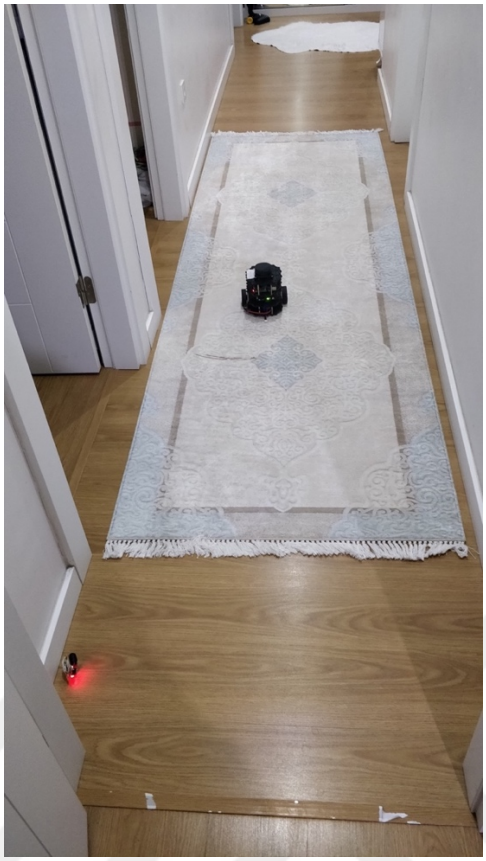
1. dataset.csv dosyasını okuyarak bir dataset dizisinde bu verileri sakla.
2. Bluetooth taramasını başlat.
3. Bluetooth verisi gelince listeye ekle veya listede zaten varsa RSSI değerini güncelle.
4. TIME_LIMIT değeri ile belirtilen sürede bir Parmakizi algoritmasını çalıştır.
5. Ağırlık değeri en yüksek olan veriyi "/initialpose" konusuna gönder.
6. Eğer ağırlık değeri 1 ise pushCount'u 1 arttır.
7. Eğer pushCount, PUSH_LIMIT'e eşit ise düğümü sonlandır.
8. Eğer değil ise 3. adımdan tekrar devam et.

Düğüm kapatılana kadar bu şekilde çalışır ve sürekli robota konum tahmini yaparak gönderir. Buradaki amacımız robotun çevresindeki örnekleri harita dosyası ile karşılaştırarak konum tahmini yaptığı metoda yardımcı olarak daha kısa sürede konum tahmini yapabilmesini sağlamaktır. Robot ile yöntem arasındaki iletişimi ise AMCL yönteminin yayınlamış olduğu "/initialpose" adlı konu üzerinden sağladık. Bu konu üzerinden yayınlanan "geometry_msgs/PoseWithCovarianceStamped" tipindeki mesajı robot konum tahmini olarak işlemektedir ve kendisini o noktadaymış gibi konumlandırmaktadır. Pamakizi metodundan gelen çıktıyı "/initialpose" adlı konuya PoseWithCovarianceStamped mesajı olarak yayınlanır.

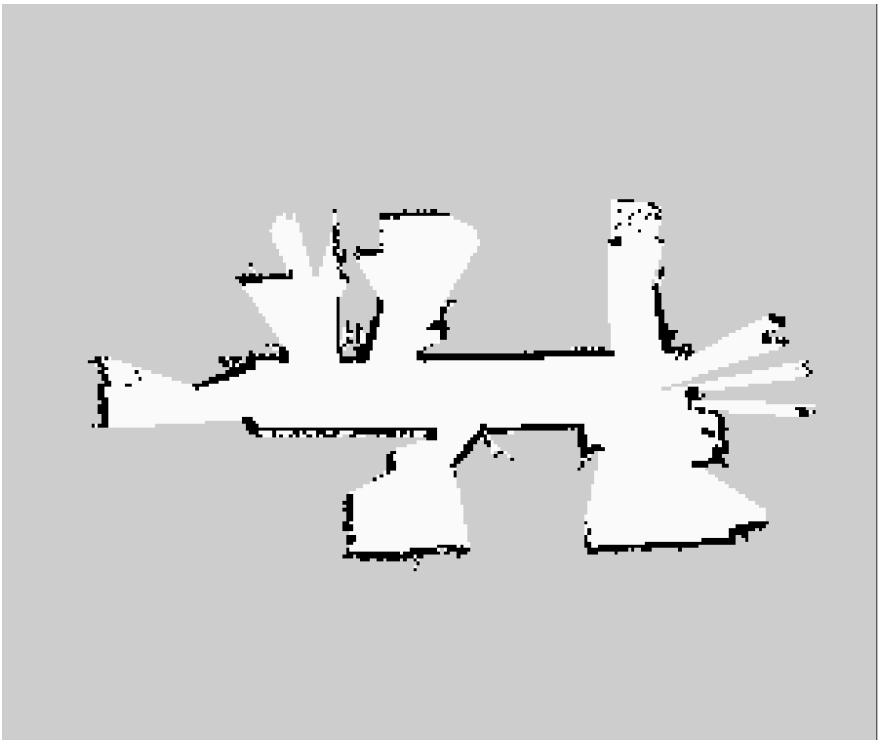
Yöntemin gerçeklenmesi için robot ile ortamın haritasının daha önceden elde edilmiş olması ve Parmakizi yönteminin çevrimdışı aşamasının yapılmış olması gerekmektedir.

Şekil 3.13'de ortam olarak seçilen bu koridorun daha önceden haritalaması robot ile yapılmış ve eş zamanlı olarak ortamdaki sinyal vericilerinden ise robotun üzerindeki Bluetooth alıcısı ile veriler toplanmıştır.

Haritalama sonucu robottan elde edilmiş olan PGM dosyasının işlenmiş hali ise 3.14'de gösterilmiştir. Robot tarafından sağlanılan bu harita verisinin boyutları haritalama yapılmış olan ortamın büyüklüğüne göre boyutlandırılıp uzatılabilirdir. Yani daha büyük haritalar da bu teknik ile elde edilebilir.



Şekil 3.13. Haritalaması yapılmış olan ortam

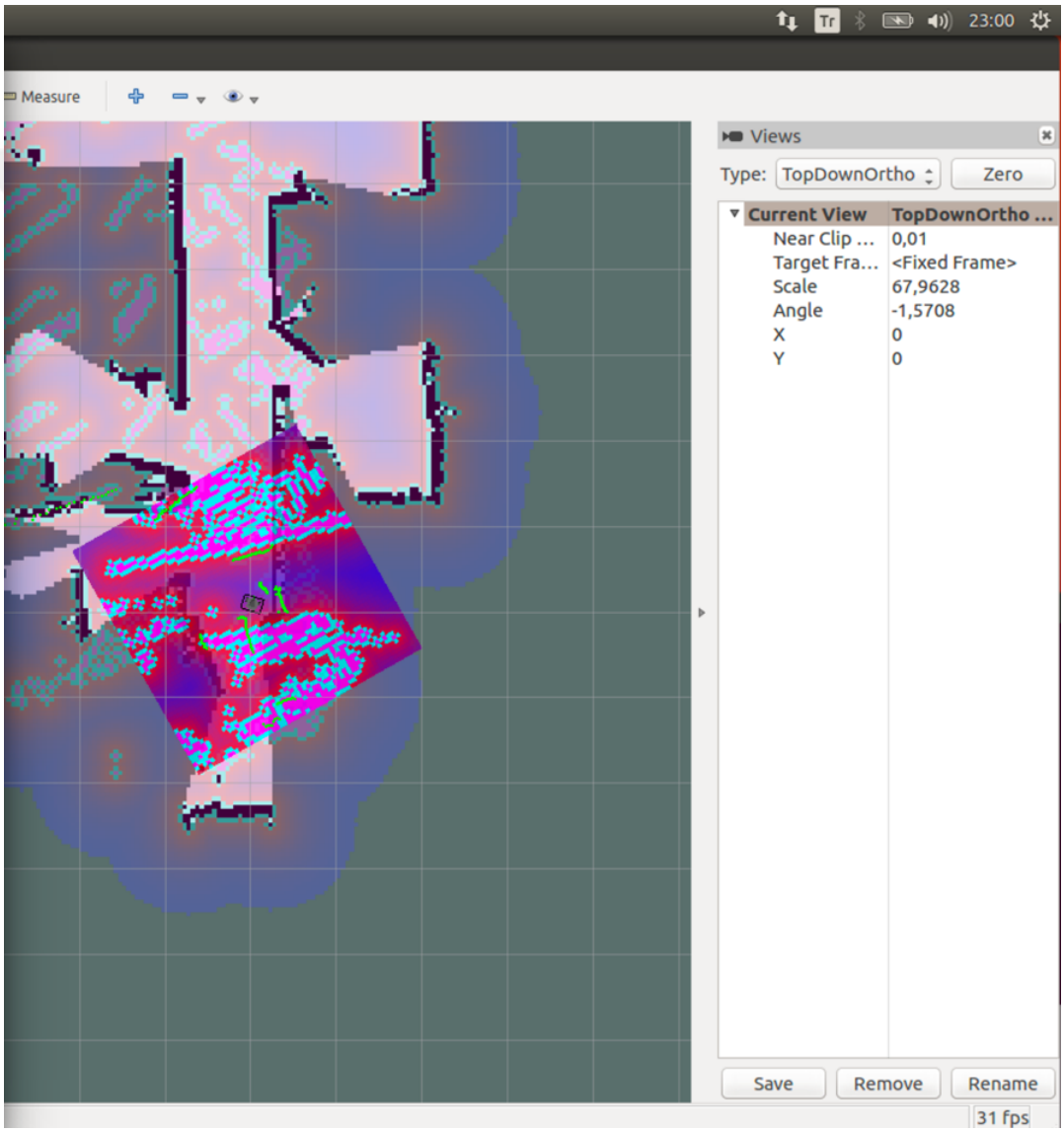


Şekil 3.14. Ortamın haritalanmış görüntüsü

Robot üzerinde Şekil 3.14'deki harita üzerinde konum bulma düğümlerini çalıştırmak için turtlebot3_navigation paketi altındaki turtlebot3_navigation.launch dosyası çalıştırılmıştır. Bunun için terminalde çalıştırılan komut:

roslaunch turtlebot3_navigation turtlebot3_navigation.launch

Ardından RViz ile haritayı ve robotun tahmini konumunu görülecek ekran Şekil 3.15'de gösterilmiştir.



Şekil 3.15. RViz ile robotun tahmini konumunu görüntüleme

Şekil 3.15'deki gibi robot kendisini başlangıç konumu olarak konumlandırmış durumdadır. Renkli olarak görünen kısımlar robotun anlık olarak LIDAR'dan aldığı

lazer tarama verileri iken siyah kısımlar önceki harita üzerindeki engelleri temsil ederken gri alanlar ise harita üzerindeki boş alanları temsil etmektedir.

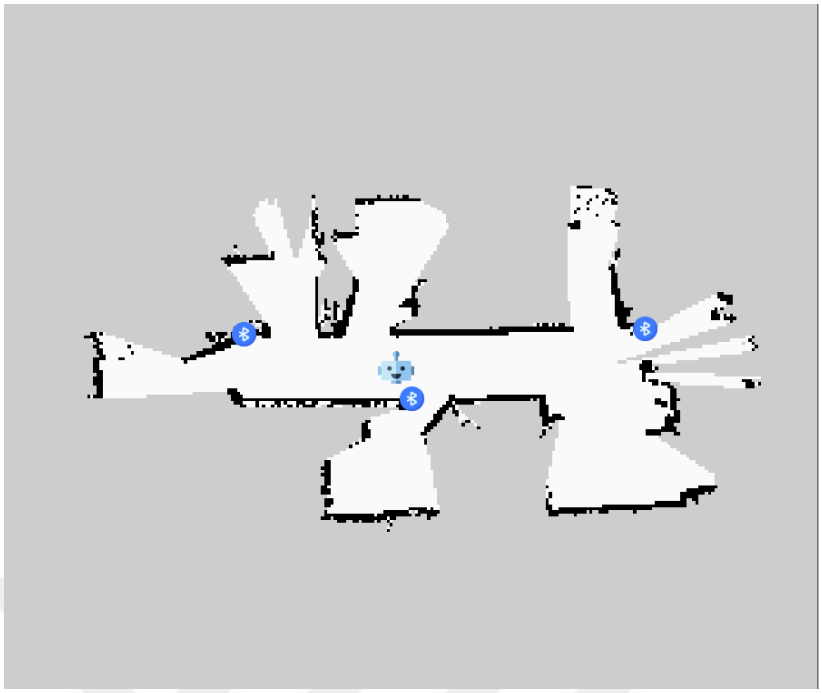
Robot Şekil 3.16'da göründüğü gibi holün ortasına doğru yerleştirilmiştir.



Şekil 3.16. Robotun yerleştirildiği konum

Şekil 3.17'de robotun konumu ve Bluetooth vericileri robot tarafından oluşturulmuş harita üzerinde şekiller ile gösterilmiştir.

Robotun konumunu tespit etmek için ise wireless_localization paketinin içindeki kodlanmış bulunan find_pose_service adlı düğümün çalıştırılması gerekmektedir. Fakat bu düğümü master bilgisayarda değil robot üzerinde çalıştırılması gerekmektedir, bunun nedeni robotun üzerindeki Bluetooth alıcısının kullanılmak istenmesidir.

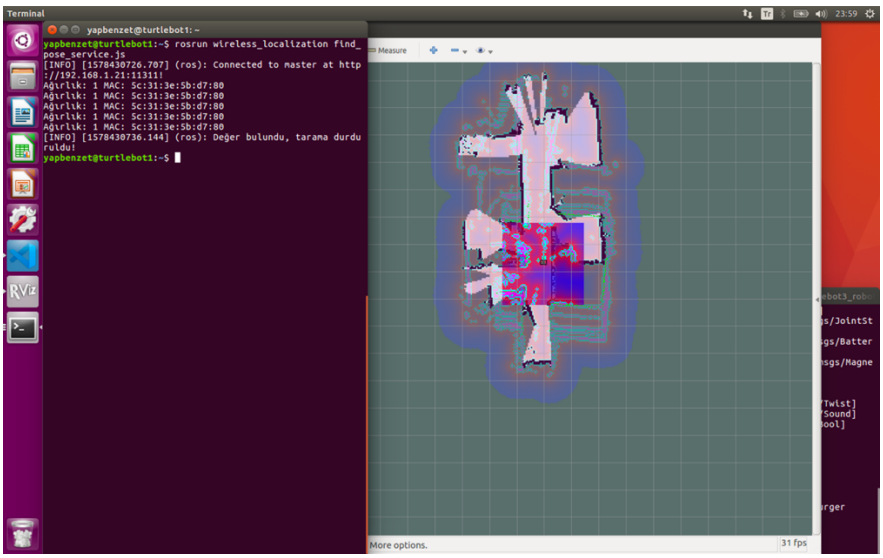


Şekil 3.17. Harita üzerinde robot ve Bluetooth vericileri

rosrun wireless_localization find_pose_service.js

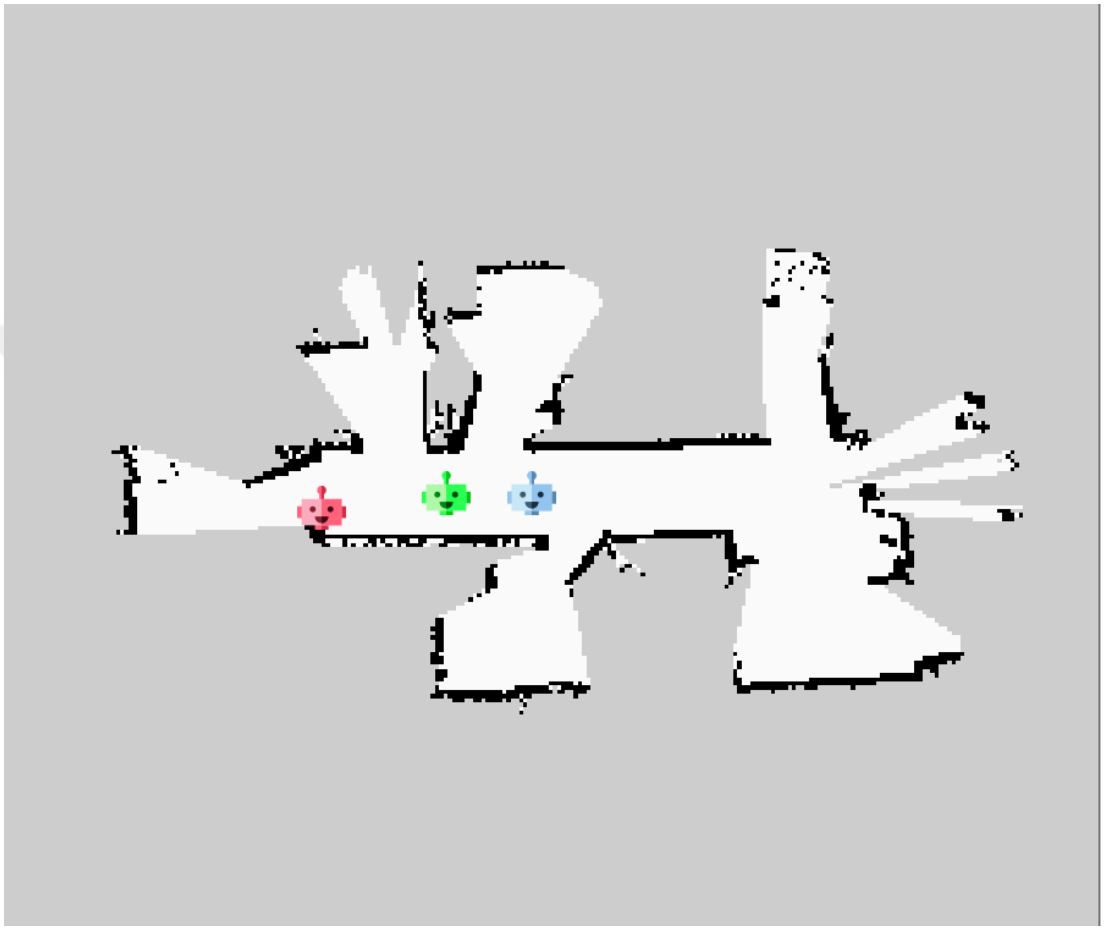
Komutunun yeni bir terminalde çalıştırılması ile yöntem başlatılmıştır.

Düğüm için belirlenmiş olan TIME_LIMIT 1000 milisaniye iken PUSH_COUNT ise 5'dir.



Şekil 3.18. find_pose_service düğüm çalışma sonucu

Şekil 3.18'de görüldüğü gibi düğüm 5 kere 1 ağırlığında veri yayınlayarak kendisini sonlandırmıştır. Düğümün çalışmasının sonucunda robotun başlangıç konumu haritanın ortalarına doğru ilerlemiştir. Şekil 3.19'da temsili olarak robotun konumları gösterilmiştir.



Şekil 3.19. Sonucun harita üzerinde gösterimi (Kırmızı robot başlangıç konumu, mavi robot gerçek konumu, yeşil robot bulunan konumu temsil etmektedir.)

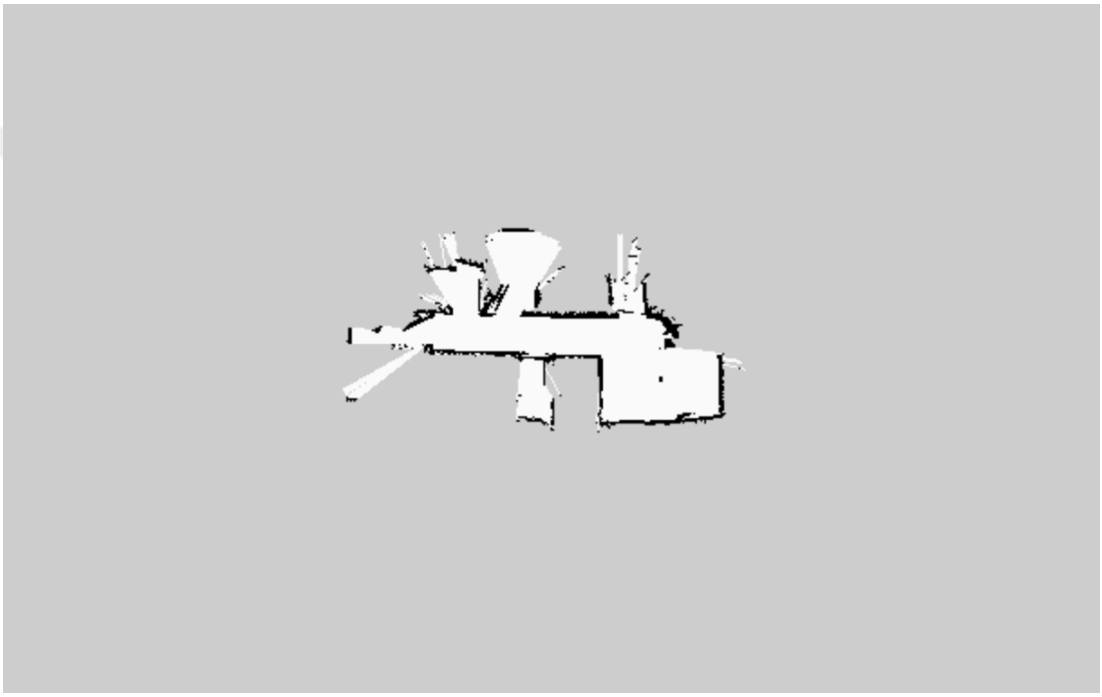
AMCL algoritması başlatıldığında robotun konumunu kırmızı robotun bulunduğu konumda göstermiştir. Mavi robot ise robotun gerçekte olduğu konumu temsil etmekteyken yeşil robot ise Parmakizi sonucunda bulunan konumu temsil etmektedir.

3.3.2. Verilerin Android uygulaması ile işlenmesi

Robot ve Android cihaz ile topladığımız verilerin bir anlam ifade edebilmesi için öncelikle işlenmesi gerekir. Toplanılan RSSI verilerinin yanında robot ile toplanılmış harita verisi de bulunur. Harita verisi PGM (Portable Grayscale Map) formatında

tutulur. Bunun yanında veriyi ifade eden bir de YAML dosyası oluşur. Burada da harita ile ilgili çözünürlük, harita uzayının boyutları gibi bilgiler tutulur.

Android uygulamasında harita verisinin görselleştirilmesi için bir harita komponenti oluşturulmuştur. Bu harita komponenti kendisine verilen PGM dosyasını okuyarak kullanıcının anlayabileceği harita verisine dönüştürür ve bu veri uygulama üzerinde harita olarak ekrana getirilir. Şekil 3.20'deki gibi harita görselleştirildikten sonra tek başına herhangi bir anlam ifade etmez.

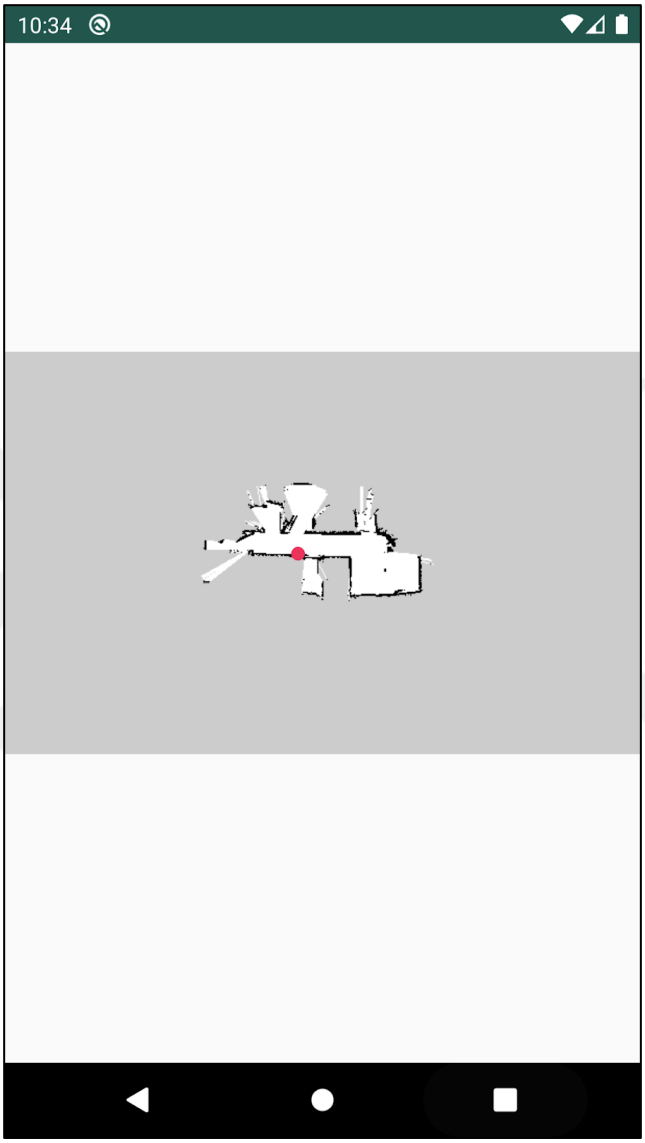


Şekil 3.20. PGM dosyasındaki verinin işlenmiş görseli

Elde işlenmemiş bir veri daha var o da ortamdan toplanmış olan biri robottan diğeri telefondan olmak üzere iki farklı veri seti bulunur. Bu iki veri seti de aynı formatta toplanmıştır. Harita komponentinin üzerinde bir nokta görselleştirebilmek için bir fonksiyon yazılmıştır. Bu fonksiyon harita üzerinde odometri konusundan toplanmış olan nokta verilerini görselleştirmeyi sağlar.

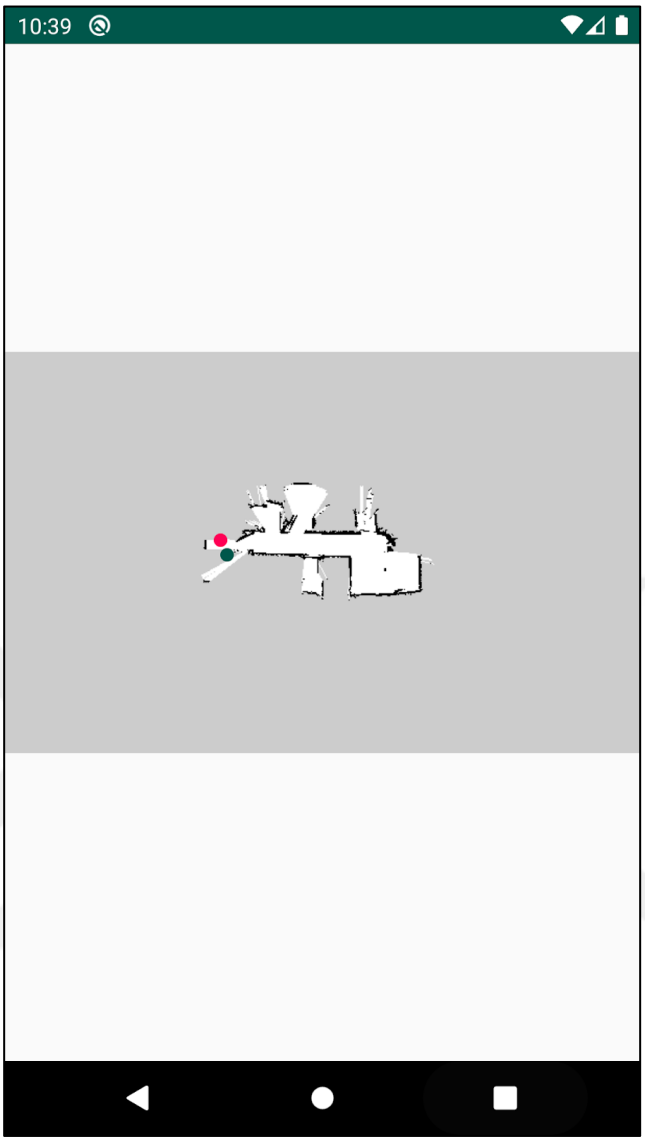
Şekil 3.21'deki komponent bu durumları otomatik olarak ele alacak şekilde geliştirilmiştir.

Robottan ve telefondan toplanan verilerin aynı olması ise kolayca veriler değiştirilerek aradaki farkın görülmesini sağlar. Karşılaştırma yaparken sadece veri setini değiştirilmesi yeterli olur.



Şekil 3.21. Harita üzerinde konum verisinin gösterilmesi (Pempe nokta)

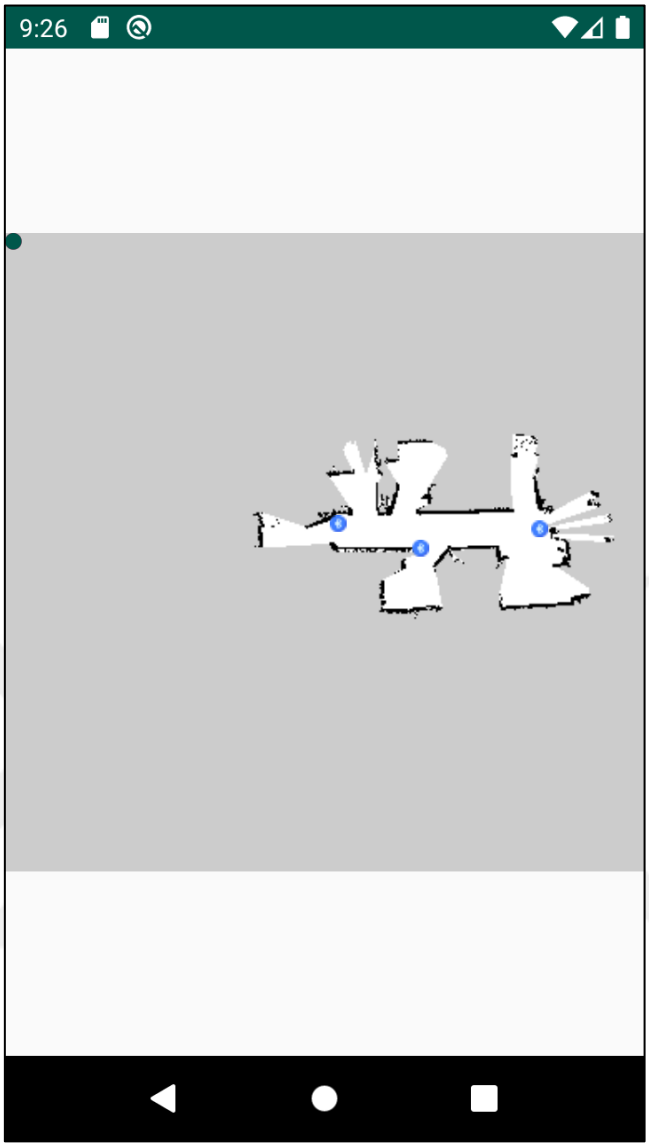
Aynı zamanda uygulamada trilateration metodu da eklenmiştir bu metot ise bashedilmiş metotlardan Üçgenleme Metodudur. Bu metot ise ortamdaki BLE cihazlarının taranması ve konumu bilinen BLE cihazları arasında üçgenleme metodunu çalıştırmaktadır ve çıktısının harita komponenti üzerinde ayrı bir nokta olarak göstermektedir. Şekil 3.22'de Android uygulaması ile verinin görselleştirilmesi ve test edilebilmesi sağlanmıştır.



Şekil 3.22. Trilateration ile tespit edilen konumun gösterilmesi (Yeşil nokta)

Bu platform üzerinde başka tespit algoritmaları da gerçeklenebilecek şekilde geliştirişlmiştir.

Son olarak Android uygulamasında gerçekleştirilmiş görselleştirme ise Bluetooth vericilerinin harita üzerinde işaretlenmesi için toplanılan verilerin görselleştirilmesidir. Bunu ise toplamış olunan verilerin tutulduğu beaconsLocations.csv adlı doyayı uygulamaya sunularak sağlanmıştır. Buradaki verileri x ve y konumlarına göre değerlendirerek harita üzerinde gösterir. Şekil 3.23'de haritadan toplamış olan veriler mobil uygulama üzerinde şekildeki gibi gösterilir.



Şekil 3.23. Uygulama üzerinde vericilerin görselleştirilmesi

Bu veriler aynı zaman trilateration algoritmasının da gerçeklenmesi için kullanılmaktadır.

3.3.3. Geleneksel parmakizi yöntemi uygulaması

Geliştirmiş olduğumuz bu yöntemi karşılaştırmak için literatürde gerçekleştirilmiş olan elle veri toplama ile Pİ yöntemini gerçekleştirmek için ayrı bir Android uygulaması geliştirilmiştir. Bu uygulama Pİ yönteminin iki fazını da gerçekleştirebilen bir uygulamadır. Uygulama veri seti oluşturmak için bir arayüz ve konum tespiti yapmak için oluşturulmuş olan diğer bir arayüzden oluşur.

Veri toplama aşaması için ortamdaki Bluetooth vericilerinden veri toplamaya başlayan uygulama başla tuşuna tıklanması ile veri toplama işlemine başlar. Veri toplama işleminin adım adım yapılması üzerine kurgulanmış bir yöntemdir. Kullanıcı belirlenen bir mesafe kadar yol alarak harita üzerinde gezinir ve bu gezinme esnasında haritadan adımlara özgü veriler toplanır. Şekil 3.24'de harita üzerinde 7 adımlık veri setinde toplam 350 Bluetooth verisi toplanmıştır.



Şekil 3.24. Geleneksel veri toplama uygulaması

Sonraki Adım (Next Step) tuşuna tıklanıldığında cihaz adım sayısını 1 arttırır ve ortamdan veri toplamaya başlar, belirlenmiş değer kadar veri topladıktan sonra beklemeye geçer ve kullanıcının bir sonraki adıma geçip tuşa tıklamasını bekler. Veri Setini Temizle (Clear Dataset) tuşu ise veri setini temizler. Parmakizi (Fingerprint)

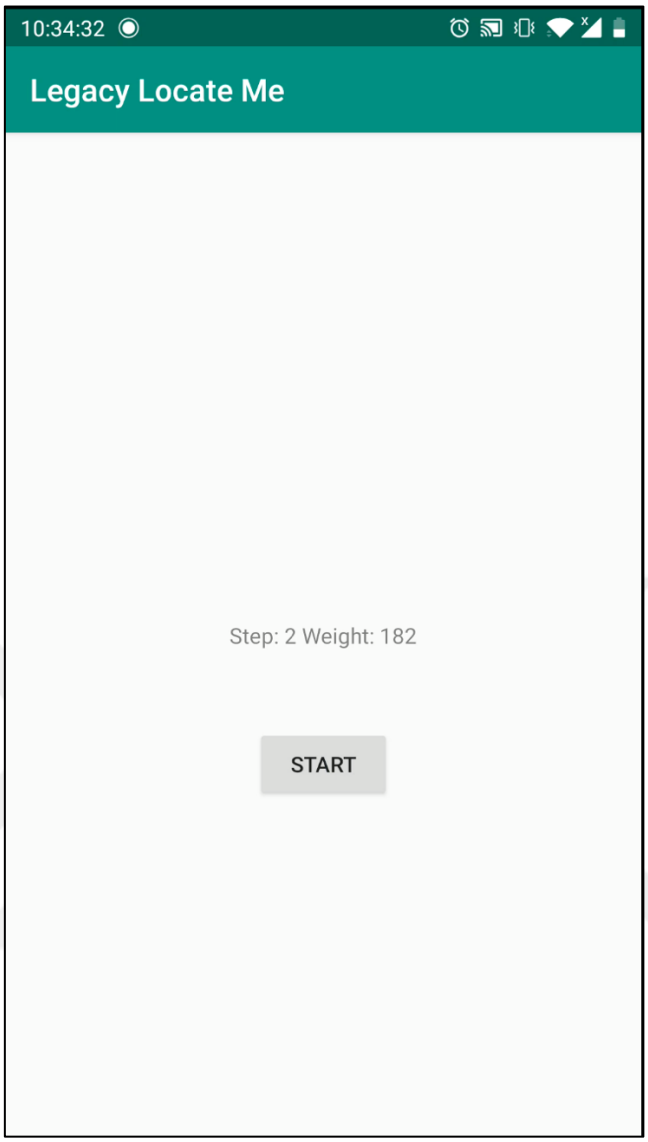
tuşu ise Pİ yöntemini gerçekleştirerek konum tespiti yapan ekrana geçişi sağlar. Tablo 3.4'de veri seti gösterilmiştir.

Tablo 3.4. Veri setinin saklandığı veri yapısı

Step	Name	Mac	RSSI	Clock	Day	Month	Year
1	JDY-08	34:15:13:E5:31:DB	-59	17:55:07	09	02	2020
1	null	A2:36:9A:64:AD	-69	17:55:07	09	02	2020
1	JDY-08	88:4A:EA:7E:83:3C	-80	17:55:08	09	02	2020
1	null	0E:DE:20:4E:DB:65	-91	17:55:10	09	02	2020
1	null	63:A2:36:9A:64:AD	-64	17:55:10	09	02	2020
1	BT05	5C:31:3E:5B:D7:80	-82	17:55:10	09	02	2020

Pİ yönteminin gerçekleştirildiği aşamada ise ortamdan toplanan Bluetooth verilerinin veri setindeki veriler ile karşılaştırılarak hesaplanan ağırlık değerine göre, ağırlık değerinin en düşük olduğu adım yani gerçek konuma en yakın olduğu düşünülen konuma göre tahmin yapılır. Burada çalıştırılan yöntem ile otomatik veri toplama yönteminde çalıştırılan yöntem birebir aynıdır.

Şekil 3.25'de Başla (Start) tuşuna basıldığında yöntem çalıştırılır ve tespit edilen adım kullanıcıya gösterilir. Burada adım olarak gösterilen veri sadece bir etikettir. Bu etiket ofis, mutfak, lavabo veya bekleme alanı gibi etiketler de olabilir. Fakat burada ölçmek istenilen kısa mesafeler ile toplanmış olan veriler üzerinde tespit yaparak gerçek konuma yakın veriler göstermek olduğu için etiketleri adımlar ile etiketlenmiştir. Adım verilerini ise birinci adım başlangıç adımı olarak düşünülmüştür ve her adımda adım sayısı bir arttırılarak bir sonraki adımda veri toplanılması sağlanmıştır. Veri toplamaya başlanılmasının ardından uygulama önceden belirlenmiş sayıdaki veriyi toplayana kadar ortamdan veri toplar. Birden fazla test yapılacağı için burada bu sayı 50 olarak belirlenmiştir. Bu sayıdaki veriyi toplamak kullanıcıyı çok zorlamayacaktır ve yöntemin çalışması için yeterli olacaktır.



Şekil 3.25. Geleneksel Pİ konum tespiti

3.4. Android Geliştirme Ortamı

Android platformu için geliştirilmiş olunan komponentler bu başlık altında anlatılmıştır. Bu geliştirme ortamında hedef bir harita görüntüleyicisi sunabilmek ve bunun üzerinde kullanıcının konumunu gösterebilecek yapılar oluşturmaktır. Harita verisini okumak için oluşturmuş olduğumuz komponent PGM dosyasından okuma yaparak haritayı oluşturur. Komponentin konum verisini işlemesi için ise YAML dosyasını ve konum bilgilerini işlemesi gerekir. Yani bu verileri komponente sağladıktan sonra bu ortam üzerinde herhangi bir konum tespiti yöntemi kolayca gerçeklenmiştir. İstenilen metodun sadece matematiksel ifadesini kodladıktan sonra verinin gösterimi için sadece çıktıyı komponente gönderilmesi yeterli olmaktadır.

Geliştirilmiş olunan platformda Pamakizi ve Bluetooth Trilateration yöntemleri gerçeklenmiştir ve sonuçları toplanmıştır. Ayrıca platformun kaynak kodlarını da sunulmuştur [32]. Bu geliştirme ortamının geliştirilmesi ve farklı algoritmaların gerçeklenmesi için yapılmıştır.

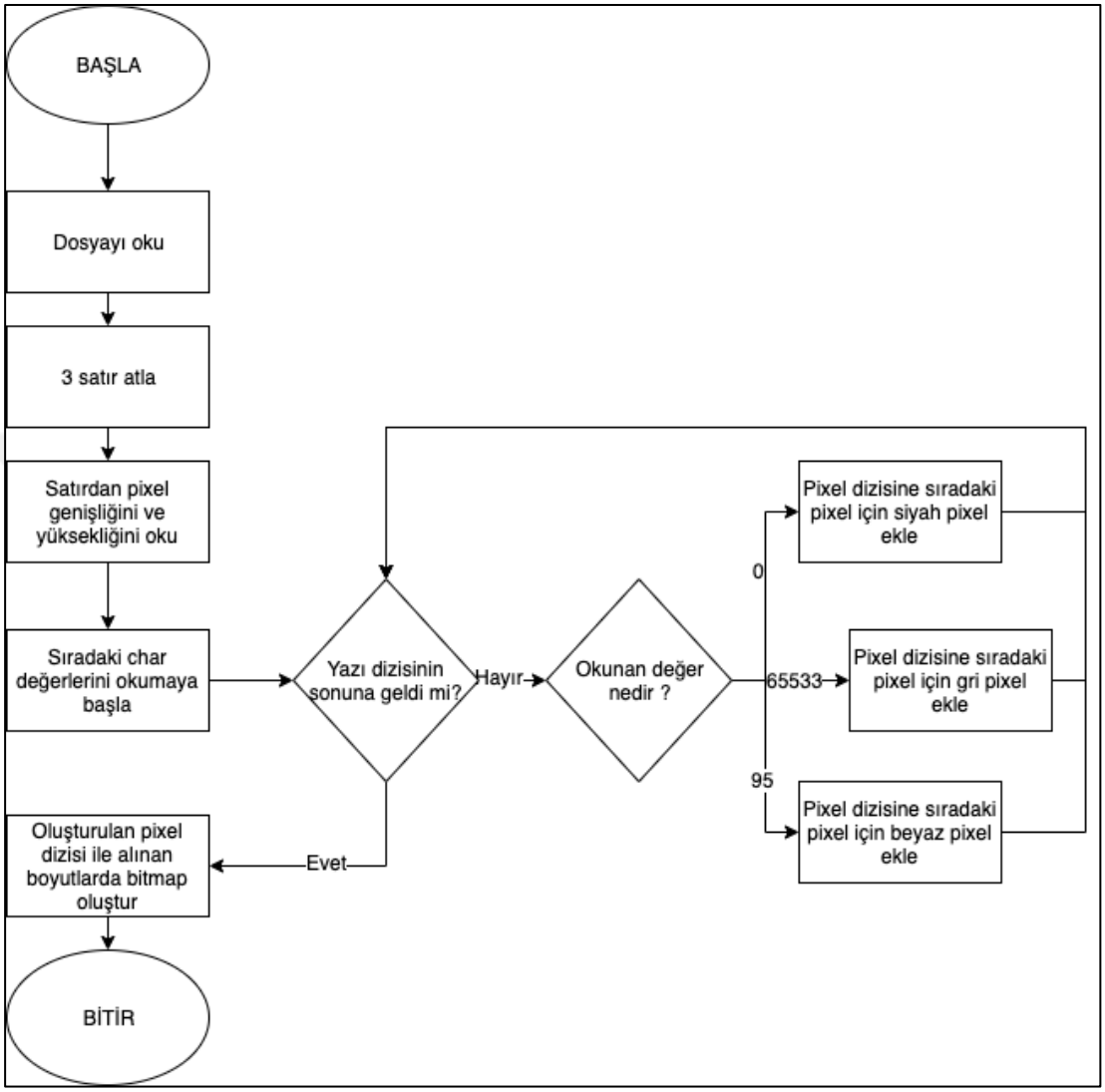
Geliştirme ortamı MVVM (Model - View - ViewModel) prensipleri ile kodlandı; mantıksal işlemler ViewModel üzerinden gerçeklenirken, Trilateration gibi metotlar ise interactor'a bölündü ve her işlem parçacığı bu şekilde parçalanarak kod karmaşasının önüne geçilmiştir. Görüntü ile ilgili işlemler ise View sınıfı olan Activity sınıfı üzerinde gerçeklenmiştir. Android lifecycle komponentlerinden olan LiveData'lar üzerinden ise ViewModel - View geçişleri sağlanmıştır. Observable pattern ile kodlanan View üzerinden ise anlık veri değişimleri sağlanmıştır. Harita komponenti de ayrı bir Component View olarak tasarlnamıştır ve konum gösterim fonksiyonları da bu komponent üzerinden sağlanmıştır.

Geliştirme ortamı Android'in yeni yeni desteklemeye başladığı Kotlin programlama dili kullanılarak kodlanmıştır. Kotlin'in sağlamış olduğu programlama gücü sayesinde birden fazla foknsiyon ile yapılacak işlemler tek satırda kodlanılabilmiştir.

3.4.1. Harita görünümü komponenti

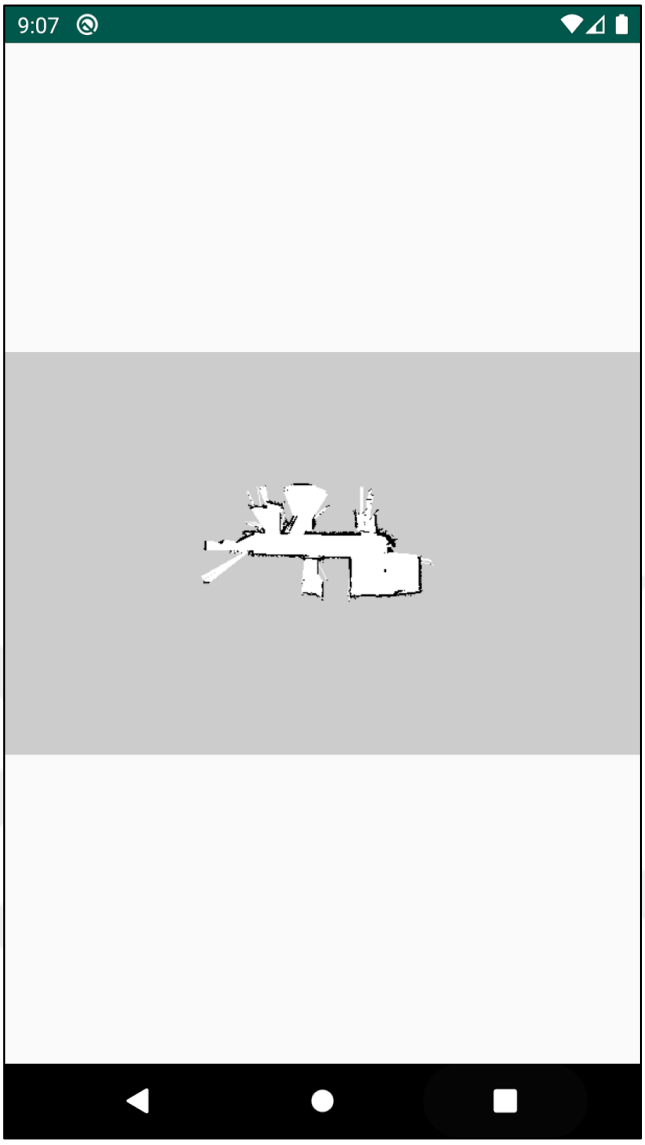
Android geliştirme ortamında üzerinde özellikle durulan komponent harita komponenti olmuştur. Harita komponenti çalışılan konunun en önemli başlıklarından birisidir. Amacı ise elimizdeki ham veriyi kullanıcının anlayabileceği bir veri olarak kullanıcıya sunmasıdır. Harita verisinin işlenilerek kullanıcıya bir kroki şeklinde sunulması ise bu veririnin en verimli işlenebileceği yöntemdir. Komponent iki farklı özelliğe sahiptir; birincisi harita verisini haritaya dönüştürmek iken ikincisi ise bu harita üzerinde işaretlenen bir noktayı gösterebilmesidir. Harita komponentin beslenme kaynağı robot tarafından üretilmiş olmuş olan PGM dosyasıdır. Portable Grayscale Map olarak geçen bu format haritadaki verileri 3 farklı char olarak beyaz, siyah ve gri alanlar olarak tutmaktadır. Burada gri renkte tutulan pixeller taranmamış alanları ifade ederken siyah olan pixeller engelleri, beyaz olan pixeller ise taranmış uzaydaki engel olmayan boş alanları ifade etmektedir. Bu okumayı yapacak olan MapView adlı komponentimiz "windows-1254" charset'i ile okunmuş bir String beklemektedir. Bu Stringi karakter karakter okuyarak bitmap dosyasını oluşturarak

ekranda göstermektedir. Şekil 3.26'da gösterildiği gibi bitmap dosyası elde edilir ve harita komponenti üzerinde gösterilir.



Şekil 3.26. Harita bitmap dosyasının PGM dosyası ile oluşturulması

Şekil 3.27'deki harita üzerinde tıpkı diğer navigasyon yazılımlarında olduğu gibi pinch-to-zoom hareketleri de bulunmaktadır ve bu şekilde kullanıcıya alışkın olduğu deneyimi sunulmuştur. Robotun toplamış olduğu harita verisi ise bu şekilde işlenerek haritaya dönüştürülmüştür. Harita verisi ise konum tespitinde kullanıcının konumunun tespitinde görselleştirme adına çok önemli bir yer tutmaktadır.



Şekil 3.27. Elde edilen haritanın uygulama üzerinde gösterimi

Bu harita üzerinde tespit edilen kullanıcı konumu ise bir başka fonksiyon yardımı ile sağlanmaktadır.

3.4.2. Komponent üzerinde konum tespiti

Harita oluşturmaktan sonra ikinci problem ise bu harita üzerindeki pixellerde elimizdeki veri ile harita üzerindeki bir noktayı işaretlemektir. Bu işlemi gerçekleştirebilmek için robot tarafından haritalama esnasında oluşturulan YAML dosyasından yardım alınmıştır. Örnek bir yaml dosyası aşağıdaki gibi oluşmaktadır.

image: Desktop/home_working_rssi.pgm

resolution: 0,050000

origin: [-10,000000, -10,000000, 0,000000]

negate: 0

occupied_thresh: 0,65

free_thresh: 0,196

Buradaki değerler aşağıdaki açıklamaları ifade etmektedir:

image: PGM dosyasının bulunduğu konum

resolution: Haritanın çözünürlüğü, metre / pixel

origin: Haritanın en sol altındaki pixelin 2D pozisyonu w rotasyon olmak üzere (x,y,w)

negate: Serbest / işgal edilmiş anlamının tersine çevrilmeli mi sorusunu ifade eder

occupied_thresh: Pixellerin tamamen dolu sayılma eşiği

free_thresh: Pixellerin tamamen boş sayılma eşiği

Burada sağlanan resolution (çözünürlük) değeri harita üzerinde konum işaretlemesi yaparken efektif olarak kullanacağımız bir değer. Harita üzerinde kaç pixelin 1 metreye denk geldiğini bilmemiz örneğin gelen bir A(1.545, 2.456) noktasını harita üzerinde ifade ederken çok büyük anlam ifade etmektedir.

Konum tahmini hesaplaması yapılırken 2 değer hesaplanmaktadır bu değerler x ve y değerleridir. x ve y değerleri hesaplanan konum görüntüsü ekranda kullanıcının konumu olarak gösterilir.

$$X = (W/2) + (xWR) - (PW/2) \quad (3.1)$$

$$Y = (H/2) - (yHR) - (PH/2) \quad (3.2)$$

3.1 ve 3.2 denklemlerinde Tablo 3.5'de açıklanmış olan veriler yerine yerleştirilmesiyle A(x,y) noktasının harita üzerinde gösterilmesi için gerekli olan P(X,Y) noktası sonuç olarak bulunur.

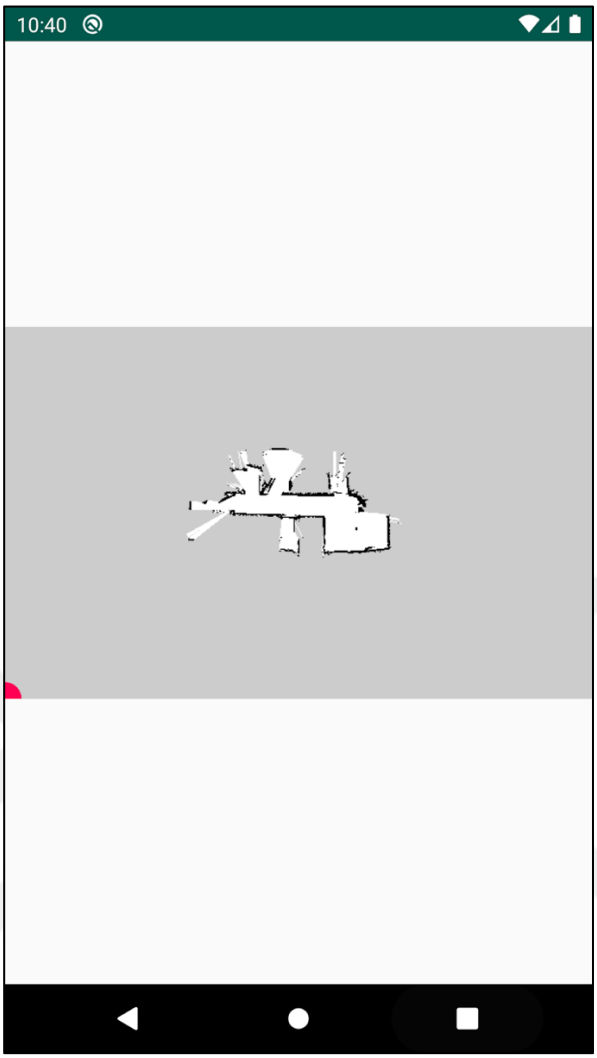
Tablo 3.5. Konum tespiti formülünde değerler

Değişken	Açıklama
W	Haritanın genişliği
H	Haritanın yüksekliği
R	Çözünürlük değeri
X	Hesaplanmak istenen x ekseni değeri
Y	Hesaplanmak istenen y ekseni değeri
x	Bulunmak istenen x konumu
y	Bulunmak istenen y konumu
PW	İşaretleyici görselin genişliği
PH	İşaretleyici görselin yüksekliği

Harita komponentinin en sol en üst köşesi O(0,0) noktası origin olarak kabul edilmektedir. Fakat harita verisi üzerinde ise origin (en sol alt köşe) ise komponente göre (0,H) noktası olarak kabul edildiği için yukarıdaki dönüşüme bu da eklenmiştir.

Burada x ve y ekseni için için işaretleyicinin boyutunu etki etmesinin sebebi ise y ekseninde görüntü yukarıdan aşağıya doğru görselleştirildiği için noktanın kendisini vermemiz durmunda aslında noktayı işaretleyici boyutu kadar aşağıda gösterecek olmasıdır. Aynı durum x ekseni içinde sağa doğru bir kayma olarak gerçekleşecektir. İşaretleyicinin tam orta noktayı göstermesi için x ekseninden ve y ekseninden işaretleyicinin boyunun yarısı eksiltilmektedir.

Sonuç olarak bu harita için B(-10,-10) olan origin yani en sol alt köşe olan değeri harita komponente değer olarak yollandığında Şekil 3.28'deki görüntü elde edilmiştir. Şekil 3.28'de sonuç olarak harita komponenti üzerinde kendisine verilen bir harita üzerindeki konum verisini komponente göre hesaplayarak gösterimini yapması sağlanmıştır.



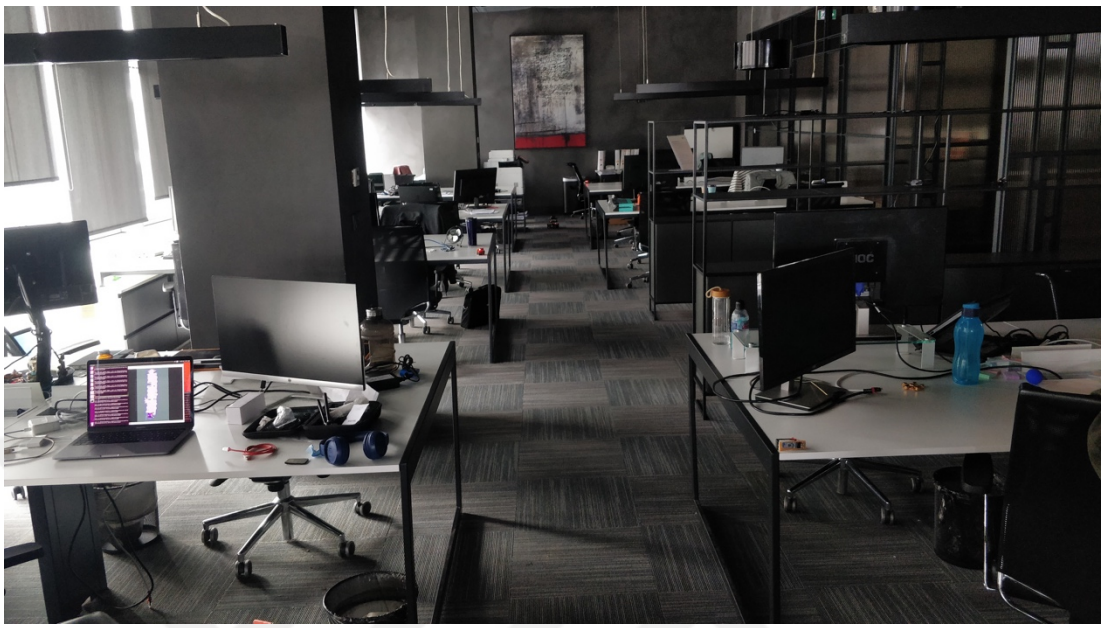
Şekil 3.28. Harita üzerinde origin noktasının gösterilmesi

Yaml dosyasında belirtilmiş olan origin noktasının yerine yerleştirilmesi ile konum gösterme yönteminin doğruluğu sağlanır.

3.5. Gerçek Dünyada Veri Toplama ve Konum Tespiti

Çalışmayı gerçekleştirmek için bir ofis ortamını test çalışma alanı olarak belirlenmiştir. Bu ofiste öncelikle ortamın haritası oluşturulmuştur ve ardından hem robot ile hem de Oneplus 5 model telefon ile veriler toplanmıştır. Toplanılan verileri işlendiğinde ise ofis ortamının zemininden dolayı oluşturulan haritada ve yöntemin gerçeklenmesinde istenilmeyen durumlar ortaya çıkmıştır. Şekil 3.29'daki ofis ortamında haritalama işlemi yapıldığında zeminin robotun hareketini etkilemesinden

ve robotun bu etkiyi hesaba katamamasından dolayı elde edilen harita gerçek dünyadaki haritadan farklı bir harita olmuştur.



Şekil 3.29. Ofis ortamı genel görünüm

Haritadan oluşan farklılık ise robotun düz bir doğrultuda hareket ederek topladığı verilerden zeminin saptırmasından dolayı oluşan yamuk harita olmuştur. Şekil 3.30'daki ortamın zemini gösterilmiştir.



Şekil 3.30. Ofis zemini

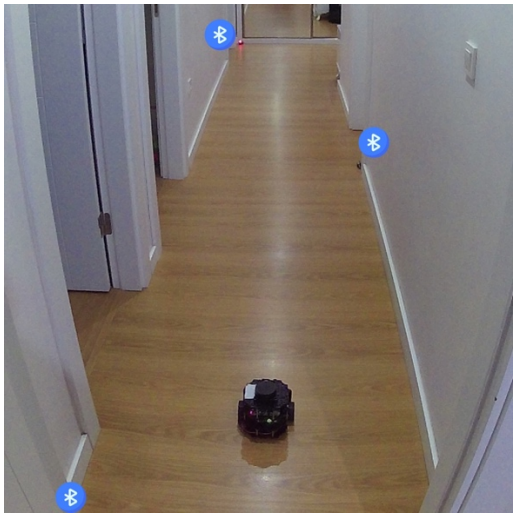
Zeminin robotun ön kısmındaki sarhoş teker üzerinde oluşturduğu istenmeyen hareket sonucunda Şekil 3.31'daki gerçeğini yansıtmayan bir harita elde edilmiştir.



Şekil 3.31. Ofis ortamının robot tarafından oluşturulmuş haritası

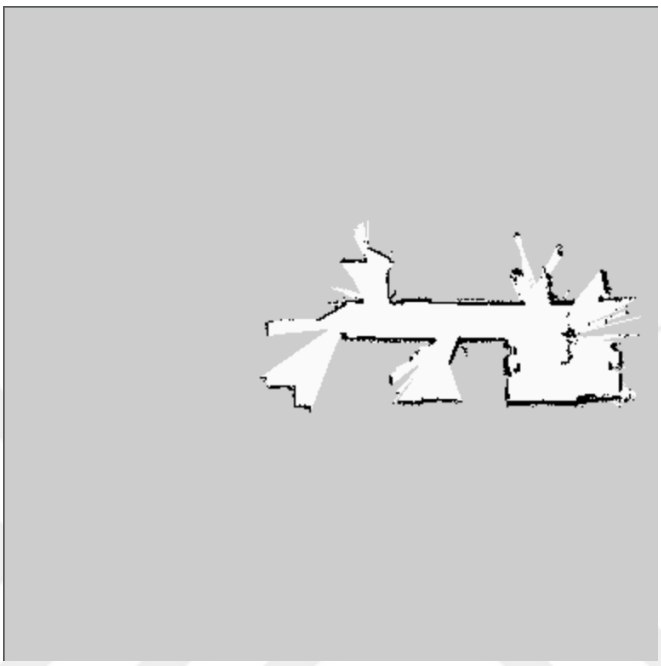
Oluşturulan haritada görülebileceği üzere yamulmalar bulunmaktadır. Sadece haritalamada değil aynı zamanda veri toplama yöntemimizde de bu zemin nedeniyle nedeni ile yanlış konum verileri toplama problemi oluşmuştur.

Alışveriş merkezlerinde de yine tekerleği etkilecek bir zemin yapısı bulunduğu için bu işlemi Şekil 3.32'deki ev ortamında düz zeminde yapmak tercih edilmiştir.



Şekil 3.32. Çalışmanın uygulandığı ev ortamı ve vericilerinin konumları (Bluetooth simgeleri)

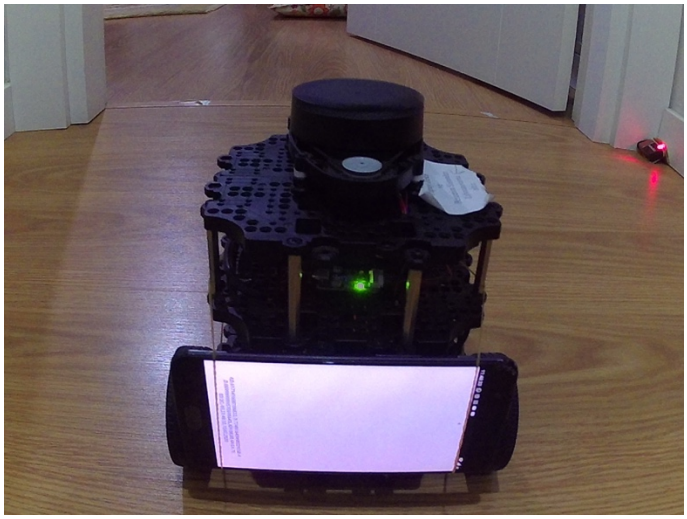
Çalışma ortamı, ortamdaki vericiler ve ortama yerleştirilmiş olan robot Şekil 3.32'de gösterilmiştir. Ortamın öncelikle haritası gmapping yöntemini kullanılarak oluşturulmuştur. Zeminin düz olması Şekil 3.33'deki gösterildiği gibi düzgün bir haritalama yapılabilmesini sağlamıştır.



Şekil 3.33. Ev ortamının haritası

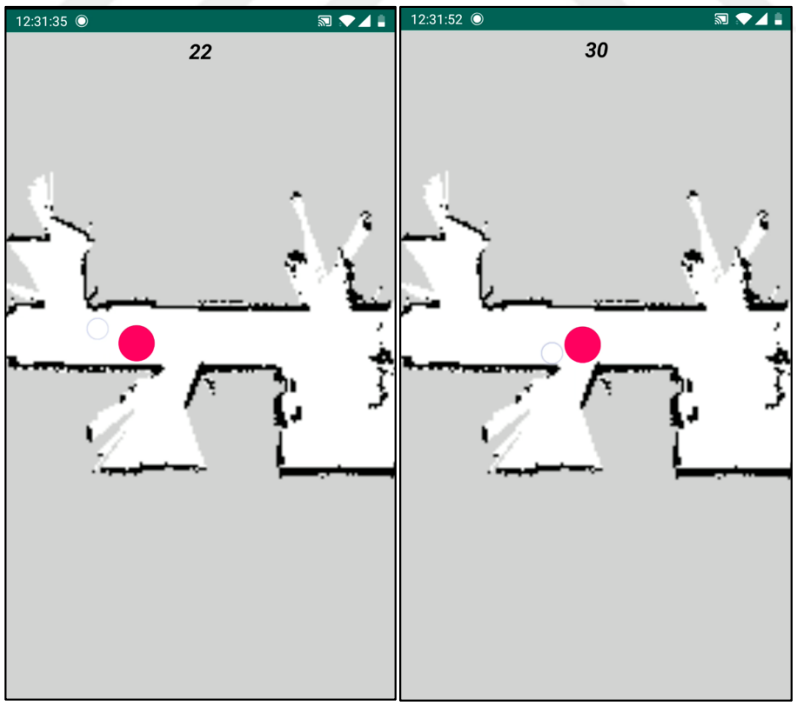
Şekil 3.33'de haritalama sonrasında elde edilen harita üzerinde konum tespiti yapılarak ortamdaki vericilerden yayınlanan sinyal güçleri ve robotun pozisyonu toplanmıştır. Bunun için robot üzerinde wireless_localization paketinin altındaki robot_wireless_localization.launch çalıştırma dosyası, master bilgisayarda ise remote_wireless_localization.launch çalıştırma dosyası çalıştırılmıştır. Bu çalıştırma dosyaları kodlanılan düğümleri çalıştırarak otonom bir şekilde haritada gezerek veri toplama işleminin gerçekleştirilmesini sağlar. Aynı zamanda robot üzerine yerleştirilmiş olan Oneplus 5 model telefonun da eş zamanlı olarak veri toplayabilmesi için ROS Wireless Localization adlı uygulaması açılmıştır ve veri toplaması sağlanmıştır.

Şekil 3.34'de robotun ortamdaki verileri toplaması için belirlenen limit olan her noktadan 200 veri toplama limiti ile 50 adımda (adım araları 10 cm olmak üzere) 5 metrelik yol kat edilerek ortamdaki veriler toplanmıştır.



Şekil 3.34. Robot üzerine yerleştirilmiş telefon

Verileri Android uygulamasına aktarılmıştır ve konum tespiti işlemine başlanmıştır. Konum tespiti için iki seçenek bulunmaktadır. Birinci seçenek robottan toplanan veriler ile konum tespiti yaparken ikinci seçenek ise telefon ile toplanan veriler ile konum tespiti sağlamaktadır. Şekil 3.35'de görülen şekillerdeki gibi konum tespitini Android platform üzerinde sağlanmıştır.

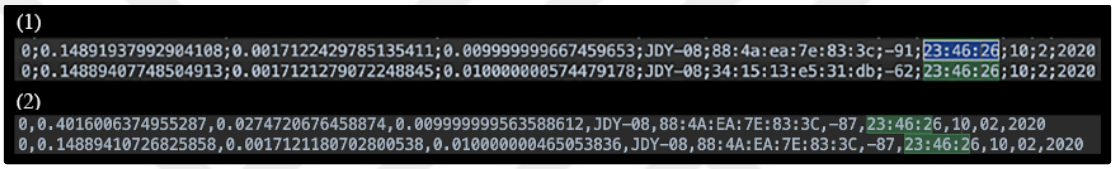


Şekil 3.35. Android ile konum tespiti (Kırmızı nokta tespit edilen konum, Beyaz nokta gerçek konum)

Yapılan gözlemlerde ise konum tespitinde ortalama 0 ile 5 adım arasında hata payı ile konum tespiti yapılması sağlanmıştır. 1 adım 10 cm olarak hesaplanır. Harita üzerinden toplanan veri sayısı arttırıldıkça ve Bluetooth vericisi sayısı arttıkça bu hata payının daha düşük rakamlara ineceğini tahmin edilmektedir.

Konum tespitinde robot ile toplanan veriler ile telefon ile toplanan veriler arasında gözle görülür bir fark oluşmadığı gözlemlenmiştir.

Şekil 3.36'da görüleceği üzere aynı zaman diliminde toplanmış olan örnek verilerin birbirlerine yakın veriler olması gözlemlenemeyen bu farkın nedenini ortaya koymaktadır.

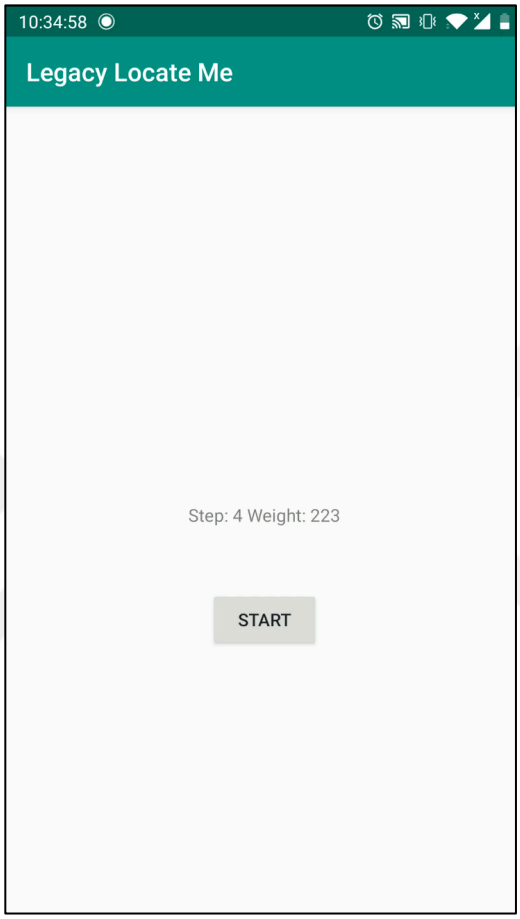


Şekil 3.36. Robot ile toplanan veri (1), Oneplus 5 ile toplanan veri (2)

Eski yöntemle de aynı ortamda veriler toplanmıştır. Bu veri toplama işlemi telefon ele alınarak yapılmıştır. Robotun topladığı kadar bir noktada durup veri toplamanın ne kadar güç olduğu gözlemlenmiştir. Aynı zamanda robot 10 cm hareket sağlarken her nokta arası bir adım boyu (30 cm) yol alması tercih edilmiştir. Noktalardan toplanılan veri sayısı ise 50 adet veri ile sınırlı tutulmuştur.

Eski yöntemle yapılan karşılaştırmalarda ise edinilen en önemli bulgu eski yöntemde veri toplama aşamasında verici ile alıcı arasına giren insan faktörü olmuştur. Cihazlar arasındaki insan engelinden dolayı sinyal seviyesinin düşmesi veriyi olumsuz etkilemiştir ve verinin toplanma aşamasında toplayan kişinin cihazlar arasındaki konumu konum tespitinde kişinin yine cihazlar arasındaki duruşuna göre farklılıklar oluşmasına sebep olmuştur. Yani toplama aşamasında vericiyle alıcı arasına insan girdiyse konum tespiti aşamasında yine aynı dizilimde konum tespiti nispeten daha güçlü yapılırken kişinin cihazlar arasından çıkması durumunda sinyal seviyesi değiştiği için hesapta olmayan bir faktör oluşmuştur ve hesaplamada farklılıklar oluşmuştur. Bunu faktör ile birlikte verilerin toplandığı doğrultuda yapılan konum tespitlerinde farklı sonuçlar ortaya çıkarken tersi doğrultuda farklı sonuçlar ortaya çıkmıştır. Aynı doğrultuda yapılan konum tespitlerinde ortalama 0 – 3 adım (0 – 90

cm) hata payı gözlemlenirken tersi doğrultuda bu değer 1 – 4 (30 – 120 cm) gibi bir değere ulaşmaktadır. Yine toplanılan veri miktarı ve adım sayısı da burada etken faktör olarak bulunmaktadır. Şekil 3.37'de geleneksel Pİ yöntemini gerçekleştirilmiş Android uygulamasının çalışma zamanını gösterilmiştir.



Şekil 3.37. Android cihaz ile geleneksel Pİ yöntemi

4. SONUÇLAR VE ÖNERİLER

Bu çalışmada iç mekan konum tespiti için stabil ve belli doğrultuda gürültüden en az şekilde etkilenerek veri toplanmasını ve bu verilerin Android platformunda işlenmesini ve kullanılması sağlanmıştır. Stabil ve belli doğrultuda gürültüden olabildiğince az etkilenecek yöntemi geliştirmek için bunları yapabilecek bir ROS çatısı altında çalışan ortam geliştirilmiştir. Geliştirilen ortam Turtlebot 3 Burger ile birlikte gerçek dünya ortamında çalıştırarak veriler toplanmıştır. Toplanılan veriler Android platformu için geliştirilmiş olan ortamda işlenmiştir ve son kullanıcıya 2 boyutlu bir harita olarak sunulmuştur. Aynı zamanda toplanmış olan RSSI sinyallerini de kullanarak Pamakizi yöntemi ile geçmiş deneyimlerden çıkarım yaparak kullanıcının iç mekandaki konumunu tespit ederken ayriyetten Trilateration yöntemi ile de kullanıcının 3 ayrı BLE cihazına olan uzaklığında konumunu tespit ederek harita üzerinde gösterilmesi sağlanmıştır. Pamakizi yöntemi için hem Turtlebot 3 robottan hem de Android işletim sistemine sahip olan bir cihazdan veri toplanmıştır. Android cihazdan veri toplarken aynı zamanda kullanıcıyı bilgilendirecek ve robotun sensör görüşünü kullanıcıya sunacak bir yazılım geliştirilmiştir.

Yapılan çalışmada geliştirilen robot yazılımı ile bir ortamdan belirli bir örüntüde veri toplamayı sağlayacak olan yazılımlar, Parmakizi / Sahne Analizi teknikleri ile gerçekleştirilen iç mekan konum bulma yöntemleri için hassas ve çok büyük miktarda veri toplanılabilmesini sağlamışlardır. Bu yazılımlar ile toplanılan veri setleri üzerinde hem Parmakizi / Sahne Analizi tekniklerinden olan Parmakizi yöntemi gerçekleştirilmiştir hem de Üçgenleme tekniklerinden olan Trilateration toplanılan veri seti üzerinde gerçeklenerek görselleştirerek kullanıcıların anlayabileceği veriler olarak sunulmuştur.

Bu çalışmanın sonucunda elde edilen deneyim, veriler ve oluşturulan yazılımlar / geliştirme ortamları ile iç mekanlarda veri toplanmıştır ve toplanan veriler işlenip kullanılabilir veriler haline getirilmiştir. Burada kullanılan bazı araçlar için ayrıca satın alınması ve önceden ortama yerleştirilmeleri gerekmektedir. Bu çalışmayı daha efektif ve maliyeti düşük çözümler ile desteklemek daha da kullanılabilir olmasını

sağlayacaktır. Ortamda hali hazırda bulunan WiFi vericilerinin kullanılması ile çalışma daha maliyetsiz bir çözüme kavuşabilir. Bu çalışmadaki amaç bu yöntemi stabil geliştirmek olduğu için maliyeti çok göz önünde bulundurmadan BLE cihazları ile gerçekleştirilmiştir. Anlattılan yöntemler farklı yöntemlerin kullanılabilmesi için altyapı niteliği taşımaktadır. Düğümler vasıtası ile katmanlı bir yapı oluşturulmuş olduğu için robot tarafında sadece bir düğüm üzerine eklenecek WiFi tarama yapısı eklendiğinde çalışma çok fazla değişiklik gerektirmeden WiFi vericileri ile Pamakizi yöntemi olarak kullanılabilir. Android geliştirme ortamında ise yine aynı şekilde Bluetooth tarama fonksiyonları WiFi tarama fonksiyonlarına evrilerek aynı şekilde çalışmanın direkt olarak WiFi üzerinden gerçekleştirilmesi sağlanabilir.

Kaynak kodlarına kaynakça kısmında ulaşarak çalışmanın geliştirilmesine destek olup farklı yöntemler ile desteklenebilir.

KAYNAKLAR

[1] Wang Y., Ye Q., Cheng J. & Wang L., RSSI-Based Bluetooth Indoor Localization, *Proceedings - 11th International Conference on Mobile Ad-Hoc and Sensor Networks,* 2016, **1**(1), 165–171.

[2] Gonzalez J. L. S., Morillo L. M. S., Alvarez-Garcia J. A., De Salamanca Ros F. E. & Ruiz A. R. J., Energy-Efficient Indoor Localization WiFi-Fingerprint System: An Experimental Study, *IEEE Access*, 2019, **7**(1), 162664–162682.

[3] Karabey I., Wi-Fi Tabanlı Parmak İzi Yöntemi Kullanarak İç Ortam Konumlandırma, Yüksek Lisans Tezi, Atatürk Üniversitesi, Fen Bilimleri Enstitüsü, Erzurum, 2015, **1**(1), 415340.

[4] Zafari F., Gkelias A. & Leung K. K., A Survey of Indoor Localization Systems and Technologies, *IEEE Communications Surveys and Tutorials*, 2019, **21**(3), 2568–2599.

[5] Bruno L. & Robertson P., WiSLAM: Improving FootSLAM with WiFi, *2011 International Conference on Indoor Positioning and Indoor Navigation*, 2011, **1**(1), 1–10.

[6] Xu S. & Chou W., An Improved Indoor Localization Method for Mobile Robot Based on Wifi Fingerprint and Amcl, *Proceedings - 2017 10th International Symposium on Computational Intelligence and Design*, 2017, **1**(1)**,** 324–329.

[7] Lee Y. C., Park B. & Park S., Coarse-to-Fine Robot Localization Method Using Radio Fingerprint and Particle Filter, *IEEE International Conference on Automation Science and Engineering*, 2014, **1**(1), 290–296.

[8] Coşgun E., Korkmaz H., Toker K., An Embedded System Design to Build Real-Time 2D Maps for Unknown Indoor Environments, *Sakarya Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, 2019, **23**(4), 617-632.

[9] Zhang W., Liu K., Zhang W., Zhang Y. & Gu J., Deep Neural Networks for Wireless Localization in Indoor and Outdoor Environments, *Neurocomputing*, 2016, **194**(1), 279–287.

[10] Varshavsky A., de Lara E., Hightower J., LaMarca A. & Otsason V., GSM Indoor Localization, *Pervasive and Mobile Computing*, 2007, **3**(6), 698–720.

[11] Farid Z., Nordin R. & Ismail M., Recent Advances in Wireless Indoor Localization Techniques and System, *Journal of Computer Networks and Communications,* 2013, **1**(1), 1-12.

[12] Jin G. Y., Lu X. Y. & Park M. S., An Indoor Localization Mechanism Using Active RFID Tag, *Proceedings - IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*, 2006, **1**(1), 40–43.

[13] Güvenç I. & Chong C. C., A Survey on TOA Based Wireless Localization and NLOS Mitigation Techniques, *IEEE Communications Surveys and Tutorials*, 2009, **11**(3), 107–124.

[14] Zhang D., Xia F., Yang Z., Yao L. & Zhao W., Localization Technologies for Indoor Human Tracking, *5th International Conference on Future Information Technology, FutureTech 2010 - Proceedings*, 2010, **1**(1), 1–6.

[15] Jung S. Y., Hann S. & Park C. S., TDOA-Based Optical Wireless Indoor Localization Using LED Ceiling Lamps, *IEEE Transactions on Consumer Electronics*, 2011, **57**(4), 1592–1597.

[16] Wu C., Yang Z., Liu Y. & Xi W. (2013). WILL: Wireless Indoor Localization without Site Survey, *IEEE Transactions on Parallel and Distributed Systems*, 2011, **24**(4), 839–848.

[17] Wu K., Xiao J., Yi Y., Chen D., Luo X. & Ni L. M., CSI-Based Indoor Localization, *IEEE Transactions on Parallel and Distributed Systems*, 2013, **24**(7), 1300–1309.

[18] Zhu X. & Feng Y., *RSSI-based Algorithm for Indoor Localization, Communications and Network*, 2013, **05**(02), 37–42.

[19] Chen Y., Lymberopoulos D., Liu J. & Priyantha B., FM-Based Indoor Localization, *In Proceedings of the 10th international conference on Mobile systems, applications, and services*, 2012, **1**(1), 169-182.

[20] Chintalapudi K., Iyer A. P. & Padmanabhan V. N., Indoor localization without The Pain, *Proceedings of the Annual International Conference on Mobile Computing and Networking*, 2010, **1**(1), 173–184.

[21] Xiong J., Sundaresan K. & Jamieson K., ToneTrack: Leveraging Frequency-Agile Radios for Time-Based Indoor Wireless Localization, *Proceedings of the Annual International Conference on Mobile Computing and Networking*, 2015, **1**(1), 537–549.

[22] Martin E., Vinyals O., Friedland G. & Bajcsy R., Precise Indoor Localization Using Smart Phones, *MM'10 - Proceedings of the ACM Multimedia 2010 International Conference*, 2010, **1**(1), 787–790.

[23] Urano K., Hiroi K., Yonezawa T. & Kawaguchi N., Poster: Basic Study of BLE Indoor Localization using LSTM-based Neural Network, *MobiSys 2019 - Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*, 2019, **1**(1), 558–559.

[24] Chen Z., Zou H., Yang J., Jiang H. & Xie L., WiFi Fingerprinting Indoor Localization Using Local Feature-Based Deep LSTM, *IEEE Systems Journal*, 2019, **1**(1), 1–10.

[25] Mehryar S., Malekzadeh P., Mazuelas S. & Spachos P., Belief Consideration Filtering for RSSI-Based State Estimation in Indoor Localization, Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, Canada Concordia Institute for Information System Engineering, Concordia Unive, 2019, **1**(1), 8385–8389.

[26] Zhou M., Yuan H., Wang Y., Tan W. & Tian Z., Indoor UAV Localization using Manifold Alignment with Mobile AP Detection, *IEEE International Conference on Communications*, 2019, **1**(1), 1–6.

[27] Luo J., Zhang Z., Wang C., Liu C. & Xiao D., Indoor Multifloor Localization Method Based on WiFi Fingerprints and LDA, *IEEE Transactions on Industrial Informatics*, 2019, **15**(9), 5225–5234.

[28] Teoman E., Indoor Positioning in The Presence of An Uncertain Reference Point, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul, 2016, 553133.

[29] Pajaziti A. & Petrit A., SLAM–Map Building and Navigation via ROS, *International Journal of Intelligent Systems and Applications in Engineering*, 2014, **2**(4), **1**(1), 71-75.

[30] Ashutosh K., Pumendu K. & Shyam C., Optimization Technique Based Localization in IEEE 802.11 WLAN, *Recent Advances and Innovations in Engineering (ICRAIE)*, 2014, **1**(1), 1-5.

[31] URL-1: http://wiki.ros.org/Nodes, (Ziyaret tarihi: 10 Aralık 2019)

[32] URL-2: https://github.com/mayuce/ros_indoor_localization_data_collector, (Ziyaret tarihi: 01 Ocak 2020)

KİŞİSEL YAYIN VE ESERLER

Yüce M. A., İnner A. B., ROS Tabanlı Robotlar İçin Özellik Eşleme ile Otonom Hedef Tespiti Yapan Android Uyglama, *International Marmara Science And Social Sciences (IMASCON)*, Kocaeli, Türkiye, 23 Kasım – 25 Kasım 2018

ÖZGEÇMİŞ

Muhammet Ali Yüce, 1995 yılında İstanbul'da doğdu. 2017 yılında Kocaeli Üniversitesi Bilgisayar Mühendisliği bölümünden mezun oldu. 2017 yılının güz döneminde Kocaeli Üniversitesi Bilgisayar Mühendisliği anabilim dalında yüksek lisans eğitimine başladı. Android Yazılım Geliştiricisi olarak profesyonel iş hayatına devam etmektedir.