MEYVE BAHÇELERİNDE DEĞİŞKEN DÜZEYLİ İLAÇLAMA İÇİN OTONOM TARIM ARACI TASARIMI

Eray ÖNLER

Doktora Tezi
Biyosistem Mühendisliği Anabilim Dalı
Danışman: Prof. Dr. İlker Hüseyin ÇELEN

2018

T.C.

TEKİRDAĞ NAMIK KEMAL ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

DOKTORA TEZİ

MEYVE BAHÇELERİNDE DEĞİŞKEN DÜZEYLİ İLAÇLAMA İÇİN OTONOM TARIM ARACI TASARIMI

Eray ÖNLER

BİYOSİSTEM MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMAN: PROF. DR. İLKER HÜSEYİN ÇELEN

TEKİRDAĞ - 2018

Prof. Dr. İlker Hüseyin ÇELEN danışmanlığında, Eray ÖNLER tarafından hazırlanan "MEYVE BAHÇELERİNDE DEĞİŞKEN DÜZEYLİ İLAÇLAMA İÇİN OTONOM TARIM ARACI TASARIMI" isimli bu çalışma aşağıdaki jüri tarafından Biyosistem Mühendisliği Anabilim Dalı'nda Doktora Tezi olarak oy birliği ile kabul edilmiştir.

Jüri Başkanı: Prof. Dr. İbrahim ÇİLİNGİR *İmza:*

Üye: Prof. Dr. Türkan AKTAŞ *İmza:*

Üye: Prof. Dr. Kubilay K. VURSAVUŞ *İmza:*

Üye: Prof. Dr. Yılmaz BAYHAN *İmza:*

Üye: Prof. Dr. İlker H. ÇELEN *İmza:*

Fen Bilimleri Enstitüsü Yönetim Kurulu adına

Prof. Dr. Fatih KONUKCU

Enstitü Müdürü

ÖZET

Doktora Tezi

MEYVE BAHÇELERINDE DEĞIŞKEN DÜZEYLI İLAÇLAMA İÇIN OTONOM TARIM ARACI TASARIMI

Eray ÖNLER

Tekirdağ Namık Kemal Üniversitesi
Fen Bilimleri Enstitüsü
Biyosistem Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. İlker Hüseyin ÇELEN

Son yıllarda tarımsal üreticiler; tarımsal işçi erişiminin belirsizliği, güvenli, erişilebilir ve yüksek kaliteli tarımsal ürünler konusunda artarak devam eden müşteri talebi, uluslarası üreticilerle olan rekabet ve karbon ayak izinin azaltılması ihtiyacı nedeniyle önemli zorluklarla karşı karşıyadır. Üreticilerin rekabetçi ve karlı üretimi devam ettirebilmeleri teknolojiye yatırım yaparak işçi maliyetlerini düşürüp, verimi arttırmalarından geçmektedir. Otonom tarımsal araçlar meyve bahçelerinde proseslerin otonom hale getirilmesi, verimliliğin arttırılması, bahçe yönetimi konusunda alınan kararlar için gerekli verilerin toplanması, işletme giderlerinin ve karbon ayak izinin azaltılması konularında önemlidir. Bu çalışmada meyve bahçelerinde değişken düzeyli ilaçlama sistemini taşımak için otonom araç tasarımı ve simülasyonu yapılmıştır. Meyve bahçelerinde odometri ve LIDAR sensörlerinden gelen verileri kullanarak meyve bahçesinin engel haritasını çıkarabilen, adaptif monte karlo lokalizasyon yöntemi ile LIDAR ve odometri sensörlerinden gelen verileri harita ile karşılaştırarak otonom aracın harita üzerindeki konumunu doğru olarak belirleyebilen, aracın harita üzerinde istenilen noktalara otonom olarak gitmesini sağlayan ve bulunduğu noktadan hedef noktaya giderken karşısına çıkan engellerden dinamik pencere yaklaşımı algoritmasını kullanarak sakınabilen bir yazılım geliştirilmiştir. Tasarlanan otonom araç değişken düzeyli ilaçlama sisteminin sadece istenilen lokasyon içerisinde ilaçlama yapmasını sağlamak için ilaçlama makinasını çalıştırma ve durdurma komutlarını verebilmektedir. Çalışmada tasarlanan otonom araç, GPS' in doğru şekilde çalışamayacağı üstü örtülü meyve bahçelerinde, çalışacak olması nedeniyle özgün bir çalışmadır. Ayrıca dışarıdan GPS verisi gibi herhangi bir veriye ihtiyaç duymayacağı için tam otonom bir araçtır. Elde edilen tasarımdan uygulamada daha az çevre kirliliği, daha az işletme gideri ve daha az iş gücü kullanılmasını sağlayarak daha yüksek verim elde edilmesi beklenmektedir. Haritalama uygulamasının başarısı, haritanın ne kadar yer değiştirme yapıldıktan sonra güncelleneceğine ve haritalamada kullanılan parçacık sayısına bağlıdır. Robotun 50 cm yer değiştirmesi ile haritanın güncellenmesi ve 30 parçacık kullanılması durumunda gerekli işlem gücü ve performans bakımından en uygun olan 3,03 entropi değeri elde edilmiştir. Lokalizasyon, robottaki sensörlerden alınan verilerin ve harita bilgisinin adaptif monte karlo lokalizasyon algoritması kullanılarak karşılaştırılması ile sağlanmıştır. Lokalizasyon başarısı, robota ait pozisyonun ne kadarlık yer değiştirme sonucu güncelleneceğine ve lokalizasyonda kullanılan parçacık sayısına bağlıdır. Konum güncellemesinin robotun 2 cm yer değiştirmesi ile yapıldığında ve minimum 500, maksimum 2000 parçacık kullanımında gerekli işlem gücü ve performans bakımından en uygun olan 3,50 cm ortalama hata elde edilmiştir. Rota planlama uygulaması, harita üzerinde lokalizasyonu sağlanmış robotun bulunduğu noktadan istenilen noktaya gidebilmesi için geliştirilmiştir. Rota planlama için Dijkstra algoritması kullanılmış, planlama global ve lokal planlama olarak iki aşamada yapılmıştır. Lokal planlamada kullanılan dinamik pencere yaklaşımı ile robotun önüne çıkan engellerden kaçabilmesi sağlanmıştır

Anahtar kelimeler: Otonom Araç, Lazer Sensör, Lokalizasyon, Haritalama, Navigasyon

2018, 122 sayfa

ABSTRACT

Ph.D. Thesis

AUTONOMOUS AGRICULTURAL VEHICLE DESIGN FOR VARIABLE RATE SPRAYING AT ORCHARDS

Eray ÖNLER

Namık Kemal University in Tekirdağ
Graduate School of Natural and Applied Sciences
Department of Biosystem Engineering

Supervisor: Prof. Dr. İlker Hüseyin ÇELEN

In recent years, agricultural producers have faced significant challenges due to the uncertainty of access to the agricultural labor force, the growing demand for safe, accessible and high-quality agricultural products, a high level of competition with international producers, and an increasing need to reduce their carbon footprint. The fact that producers can continue their competitive and profitable production only by reducing labor costs means they have to invest in technology and increase efficiency. Autonomous agricultural vehicles are important in autonomous processes in fruit orchards, increasing productivity, collecting necessary data for decisions on orchard management, reducing operating costs and carbon footprint. In this study, autonomous vehicle design and simulation were done in order to carry the variable level spraying system in the orchards. An autonomous vehicle can map the orchards by using the data from the odometry and LIDAR sensors, then accurately determine the position of the autonomous vehicle on the map by using the adaptive Monte Carlo localization method to compare data from the LIDAR and odometry sensors with the map. An autonomous vehicle can avoid obstacles by using the dynamic window approach algorithm. The designed autonomous vehicle is able to give commands for starting and stopping the spraying machine to ensure that the variable-level spraying system is sprayed only within the desired locations. The autonomous vehicle designed in the study is an original study because it will work in covered orchards where GPS cannot work properly. It is also a fully autonomous tool because it does not need any data, such as external GPS data. It is expected that higher efficiency will be achieved by providing less environmental pollution, lower operating expense and less labor force in practice. The success of the mapping application depends on the map being updated after the position displacement and the number of particles used in the mapping. When the map is updated with the 50 cm displacement of the robot and 30 particles are used, 3,03 entropy value was obtained which are the most suitable for the required processing power and performance. The localization is achieved by comparing the robot sensors' data and map information through an adaptive Monte Carlo localization algorithm. The localization success depends on a position update after the amount of displacement of the robot and the number of particles used in the localization. When the position update was made with the displacement of the robot by 2 cm and the minimum 500, maximum 2000 particles were used, a mean error of 3,50 cm was obtained which was optimal in terms of the required processing power and performance. The path planning application has been developed in order to reach the desired point from the location of the robot, which is localized on the map. Dijkstra algorithm was used for path planning, planning was done in two stages as global and local planning. The dynamic window approach used in local planning allows the robot to escape obstacles.

Keywords: Autonomous vehicle, Laser Sensor, Localization, Mapping, Navigation

2018, 122 pages

İÇİNDEKİLER

ÇİZELGE DİZİNİ

Sayfa

ŞEKİL DİZİNİ

Sayfa

Sayfa

Sayfa

SİMGELER ve KISALTMALAR DİZİNİ

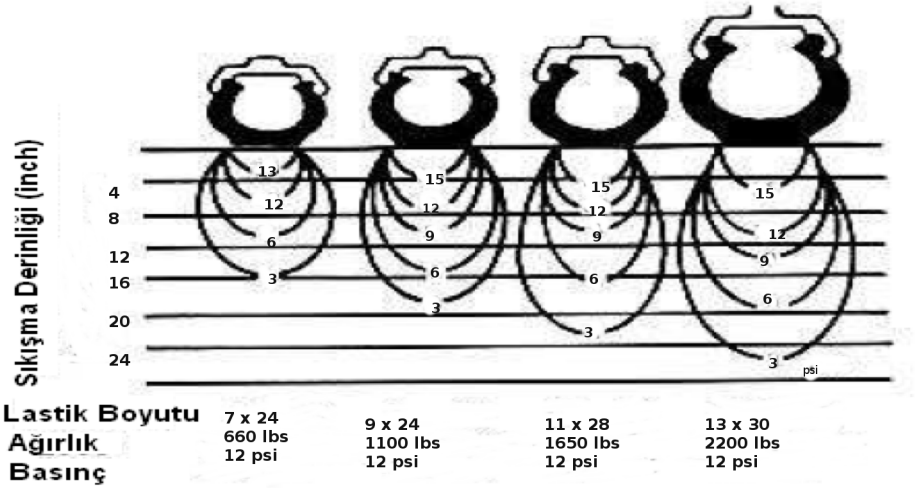
A*	: A Star Search Algorithm (A Yıldız Arama Algoritması)
CCW	: Counter Clock Wise (Saat Yönü Tersi)
CW	: Clock Wise (Saat Yönü)
cm	: Santimetre
GB	: Gigabayt
GPS	: Global Positioning System (Küresel Konumlama Sistemi)
Hz	: Hertz
LIDAR	: Light Detection and Ranging (Işık Algılama ve Mesafe Ölçümü)
m	: Metre
m^2	: Metrekare
mm	: Milimetre
OSRF	: Open Source Robotic Foundation (Açık Kaynaklı Robot Vakfı)
PF	: Particle Filter (Parçacık Filtresi)
PID	: Proportional, Integral, Derivative (Oransal-İntegral-Türevsel)
RGB	: Red, Green, Blue (Kırmızı, Yeşil, Mavi)
ROS	: Robotic Operation System (Robotik İşletim Sistemi)
ROSBAG	: Robotic Operation System Bag (Robotik İşletim Sistemi Kayıt Kütüğü)
RVIZ	: Robot Vizualization (Robot Görselleştirme)
SDF	: Simulation Description Format (Simülasyon Tanımlama Formatı)
SLAM	: Simultaneous Localization and Mapping (Eşzamanlı Lokalizasyon ve Haritalama)
XML	: Extensible Markup Language (Genişletilebilir İşaretleme Dili)
°	: Derece

1. GİRİŞ

2050 yılında dünya nüfusunun 9 milyara ulaşacağı, bu nüfusun gıda, beslenme, tekstil ve yakıt ihtiyaçlarını karşılamak amacıyla tarımsal üretimin iki katına ve üretimdeki verimin de %25 seviyesinde artırılmasına ihtiyaç duyulacağı projeksiyonları yapılmaktadır (Reid 2011). Ayrıca ülkemizin 2023 vizyonunda lokomotif sektörlerden olan tarım sektöründe dünyada ilk üç üreticiden biri olması hedeflenmektedir. Yüksek teknoloji uygulanarak verim arttırılmadan bu hedefe ulaşılamayacağı 2023 vizyon raporunda belirtilmiştir.

Tarım arazilerinin genişliği büyüdükçe ilaçlama gibi, bazı tekrarlı işlemlerin otomasyon ile gerçekleştirilmesi ihtiyacı doğmuştur. Toprak işleme, ekim ve hasat gibi yüksek güç kullanımı gerektiren işlerde, birçok durumda klasik tarım araçlarına göre hem boyut hem de motor gücü olarak daha sınırlı kapasiteye sahip tarımsal robotların kullanımı uygun olmayacaktır. Ancak nispeten daha az güç gerektiren, bitkilerin izlenmesi, haritalanması, hasat işlemini gerçekleştiren işçinin ve hasat edilen ürünün sürücüsüz bir araçla taşınması ve ihtiyaca uygun değişken düzeyli ilaçlama yapılması gibi uygulamalar için tarımsal robotlar çok uygundur.

Tarım alet ve makinelerinin ağırlıkları, tekerlek basıncı ve boyutlarının toprak sıkışmasına etkisi vardır (Şekil 1.1.). Bir tarlada meydana gelen toprak sıkışıklığının %96'dan fazlası traktörlerin düzensiz trafiği nedeniyle oluşmakta ve üretimde kullanılan enerjinin %90'ı bu sıkışıklığı gidermek için harcanmaktadır. Kök bölgesinde sıkışmanın meydana gelmesi, toprağın fiziksel ve mekanik özelliklerinde değişimlere yol açmaktadır. Sıkışma toprakların kütlesel yoğunluklarını arttırırken, toplam gözeneklilik, boşluk oranı, havalanma ve drenaj gözeneklerinde azalmalara neden olmaktadır. Toprak sıkışıklığının artışı bitki köklerinin gelişimine mekanik bir direnç oluşturmakta, köklerin ihtiyacı olan havanın alt katmanlara geçişini engellemekte, taban arazilerde ise drenaj problemlerine yol açmaktadır. Eğimli alanlarda sıkışma ile toprağa su girişi düşeceğinden, yüzey akış miktarı artacak, dolayısıyla erozyon tehlikesi ortaya çıkacaktır. Toprak sıkışması bitki gelişiminde azalmaya ve denitrifikasyona neden olmaktadır (Nolte ve Fausey 2001). Tarımsal robotlar klasik tarım araçlarına göre daha hafif olduklarından, daha az toprak sıkışmasına sebebiyet vermektedirler.



Şekil 1.1. Ağırlık tekerlek boyutu ve tekerlek basıncının toprak sıkışması üzerindeki etkisi (Nolte ve Fausey 2001)

Modern meyve bahçelerinde birçok bakım çalışması ağaç sıraları arasında düşük hızda ilerleyen bir araç ve bu aracı süren bir sürücü ile yürütülmektedir. Ağaç sıraları arasında otonom olarak ilerleyebilen bir araç, verimi artıracak ve üretim maliyetlerini azaltacak, tarım işçisini araç kullanmak gibi pasif bir rolden iş üreten bir pozisyona taşıyacaktır (Hamner ve ark. 2010).

Son yıllarda tarımsal üreticiler; tarımsal işçi erişiminin belirsizliği, güvenli, erişilebilir ve yüksek kaliteli tarımsal ürünler konusunda artarak devam eden müşteri talebi, uluslarası üreticilerle olan rekabet ve karbon ayakizinin azaltılması ihtiyacı nedeniyle önemli zorluklarla karşı karşıyadır. Üreticilerin rekabetçi ve karlı üretimi devam ettirebilmeleri teknolojiye yatırım yaparak işçi maliyetlerini düşürüp, verimi arttırmalarından geçmektedir. Otonom araçlar meyve bahçelerinde proseslerin otonom hale getirilmesi ve karar vermek için gerekli verilerin toplanması amacıyla önemlidir (Hamner ve ark. 2010).

Otonom sürüş sistemleri robotik dünyasında iyi çalışılmış bir konu olmasına rağmen, meyve bahçelerinde ve fidanlıklarda gidebilecek otonom sürüş sistemleri henüz tam anlamıyla çözülmüş değildir. Buğday tarımında kullanılan otonom traktör ve hasat makinaları geniş, açık ve engelsiz alanlarda, GPS verisini engelsiz bir şekilde alabilmenin avantajını kullanmaktadır. Bu çalışmalarda aracın navigasyonu GPS (Global Konum Belirleme Sistemi) ile yapılmaktadır (Lee ve Ehsani 2009, Belforte ve ark. 2006). Ancak meyve bahçelerinde telli terbiye yapılması, ağaçların kanopi yoğunluğu ve bahçe üzerinin ağaçları dolu gibi etkilerden korumak amacıyla

örtü ve onu taşıyan tellerle kaplanması nedeniyle GPS' e bağlı navigasyon sistemlerinin meyve bahçeleri için uygulanması oldukça zordur. Bunun yanı sıra özellikle sıra sonlarındaki dar hareket alanı, aracın önüne çıkabilecek insan vb. canlı engellerin olabilmesi meyve bahçesinde otonom olarak hareket edecek araçların çözmesi gereken problemler olarak karşımızda durmaktadır (Hamner ve ark. 2011).

Otonom tarım aracı konusunda John Deere, Class, Amazone gibi bazı firmaların yaptığı prototip çalışmaların olduğu bilinmektedir. Ülkemizde AKINSOFT firması tarafından prototip olarak üretilen elektrik enerjisiyle çalışan iki prototip robot modeli bulunmakla birlikte bu robotlar GPS yardımıyla navigasyonlarını sağlamakta ve ekim için kullanılmaktadır (Anonim 2018). GPS yardımıyla sürüş sağladıkları için tam anlamıyla otonom araçlar olarak sayılamazlar. Çünkü otonom sürüş için uydularla bağlantı içerisinde olmak zorundadırlar.

Bir çok araştırmacı otonom araç tasarlamak için çalışmalar yürütmektedir. 00558.STZ.2010-1 no' lu San-Tez projesi kapsamında (Önler ve ark. 2015) meyve bahçelerinde kanopi yoğunluğuna göre değişken düzeyli ilaçlama için geliştirilen elektronik sistemin çalıştığı 200 kg depoya sahip bir ilaçlayıcıyı taşıyabilecek kapasitede bir otonom araç tasarlanmıştır. Değişken düzeyli ilaçlama makinası tasarımının ekonomik açıdan uygunluğu, çevre ve su kaynaklarının optimum düzeyde korunması, tarımda kullanılan kimyasal miktarının bitkilerin isteklerine göre ayarlanması önemlidir (Doruchowski ve Holownicki 2000, Rüegg ve ark. 1999). Bunun sağlanması için ağaç kanopisinin karakterize edilmesi gereklidir. Geleneksel olarak manuel veya yaprakların koparılması yoluyla yapılan ölçümler bulunmaktadır. Manuel yapılan ölçümler fazla zaman almakta ve zorluğu nedeniyle alınan örnek sayısını sınırlamaktadır (Llorens ve ark. 2011, Zaman ve ark. 2006, Zaman ve Salyani 2004). Yaprakların koparılması ile yapılan ölçümler daha doğru sonuçlar vermesine rağmen bir kısım ağacın zarar görmesine neden olmaktadır. Yapılan çalışmalarda ultrasonik sensörlerin ağaç kanopi karakteristiklerinin ölçümünde yeterli sonuçlar verdiği görülmüştür (Önler ve ark. 2015).

1.1. Tarımda Otonom Navigasyon

Tarımda otonom navigasyon, mekanik direksiyon yönlendiricilerin kullanılmaya başlandığı 1950' li yıllara kadar gitmektedir (Richey 1959). Bu tür sistemler araca takılan temas sensörleri ile su borusu, sırt, tekerlek izi ve bitki sırası gibi sabit nesnelere bağlı olarak

pozisyonunu belirlemektedir. Ancak sıra üzerinde bitki olmaması durumunda ortaya çıkan ölçüm eksikliği ve bitkinin yapısı veya hasar görmesi durumlarında ortaya çıkan hatalı ölçümlerden ötürü çok kullanışlı değillerdir. Ayrıca sistem altyapısal olarak ekonomik ve genişletilebilir yapıda değildir. Bu yetersizliklere çözüm bulmak amacıyla elektronik beacon sistemleri geliştirilmiştir. Bu sistemlerde temas sensörleri yerine aracın pozisyonu üç ya da daha fazla beaconun lazer veya radar sensörleri kullanılarak nirengi noktası teşkil etmesiyle bulunur. Ancak sensör ve beacon arasındaki haberleşme bağlantısının kopması, sistemin esnek olmaması ve yüksek kurulum maliyeti gibi dezavantajları vardır.

Global Pozisyonlama Sistemi (GPS)'nin 1990'lı yıllarda ortaya çıkışıyla beacon temelli navigasyon sistemlerinin birçok yetersizliği çözülmüş tarımda, ormancılıkta ve bahçecilikte otonom navigasyon araştırmaları hız kazanmıştır (Larsen ve ark. 1994, Gan-Mor ve ark. 1997, Bell 2000, Stoll ve Kutzbach 2000, Heidman ve ark. 2002). Birçok firma Gerçek Zamanlı Kinematik GPS ve otomatik dümenleme özelliklerini taşıyan tarım araçları üretmeye başlamıştır. Günümüzde GPS teknolojisi toprak işlemede, ekim-dikimde, gübrelemede, ilaçlamada ve hasatta kullanılmaktadır. Uygulamadaki başarılarına rağmen GPS temelli navigasyonun bazı önemli limitleri vardır. Bunlardan birincisi sürekli ve hassas pozisyon tayini için GPS alıcısının uyduyu doğrudan bir hat içerisinde görmesi gerekmektedir. Ancak sinyalin ağaçlar veya diğer engeller nedeniyle engellenebildiği karmaşık tarımsal çevrede bu gereklilik problem oluşturmaktadır. Sinyal kalitesi kötü hava koşulları nedeniyle de etkilenebilmektedir. Tarımsal işlemlerde gerekli olan hassasiyeti sağlamak için 5'ten fazla uydu ile aynı anda bağlantı gereksinimi bu hassasiyet problemini daha da büyütmektedir. İkinci olarak ise sistemin doğasından kaynaklanan gecikme, otonom aracın kontrol sistemi için problem oluşturmaktadır. Üçüncü olarak ise uydu ve baz istasyonu gibi harici cihazlara ihtiyaç duyulduğu için GPS temelli navigasyon tam manasıyla bir otonomi sağlayamamaktadır.

1.2. Görüntü Temelli Navigasyon

Bilgisayar işlem gücündeki artış 1990'lı yıllarda tarım ve diğer alanlardaki otonom araçlar için görüntü sensörlerine olan ilgiyi artırmıştır. Görüntü temelli yöntemler siyah-beyaz veya renkli kameralardan alınan görüntüleri çeşitli görüntü işleme yöntemleri kullanarak robotun bulunduğu ortamdaki konumunu algılamak için kullanılmaktadırlar. Tarımda görüntü temelli navigasyon sistemleri çokça çalışılmıştır. Farklı yöntemlerle otonom navigasyon ile ilgili birçok derleme bulunabilir (Wilson 2000, Keicher ve Seufert 2000, Reid ve ark. 2000, Lee

ve Ehsani 2009). Bu sistemlerin birçoğunda bitki sıraları arka plandan ayrılarak sıra algılanmakta ve sıraya göre robotun konumu ve yönü bulunmaktadır. Bu algoritmalar sıra özelliklerini lineer regresyon gibi çizgi uydurma algoritmaları (Billingsley ve Schoenfisch 1997), k-means (Han ve ark. 2004) ve Hough Dönüşüm yöntemlerini kullanarak bulabilmektedir. Bu yöntemler kullanılarak birçok çalışma yapılmıştır (Reid ve Searcy 1986, Fujii ve Hayashi 1989, Marchant ve Brivot 1995, Marchant 1996, Hague ve Tillett 1996, Astrand 2005).

1.3. Mesafe Sensörü Temelli Navigasyon

Görüntü sensörü temelli yöntemler ışık koşullarına karşı çok duyarlıdır. Dış ortamda aydınlık koşullarda meydana gelen değişimlere bağlı olarak birçok sistem farklı çalışma koşullarına göre sürekli kalibrasyon gerektirir. Lazer mesafe ölçer (LIDAR) teknolojisi ortam aydınlığından etkilenmediği için tarımsal uygulamalar için daha uygun olabilir. Ancak LIDAR teknolojisi tarımsal uygulamalarda yüksek maliyeti sebebiyle çok tercih edilmemiştir. Bu sensör teknolojisi fiyatları düştükçe daha fazla tercih edilecektir (Ahamed ve ark. 2004). Barawid ve ark. (2007) meyve bahçelerinde gerçek zamanlı navigasyon sağlayan LIDAR temelli otonom bir araç geliştirmişlerdir. Çalışmalarında geliştirdikleri yöntemin düz çizgi tanımayla sınırlı olduğunu ve kavisli sıralarda sıkıntı yaşadıklarını belirtmişlerdir. Bu yöntemin diğer bir zayıf yönü ise çizgi algılama algoritması doğru bitki sırasını algılamayı kaçırırsa araç sırayı tamamen kaybetmekte ve geri dönememektir. LIDAR ayrıca engel algılama ve engelden kaçma için kullanılmıştır (Subramanian ve ark. 2006). Weiss ve Biber (2011) mısır sıralarının tespit edilmesi için istatiksel modeli kullanan 3 boyutlu LIDAR tabanlı bir navigasyon yöntemi geliştirmişlerdir. LIDAR 3 boyutlu nokta bulutu verisinden zemine karşılık gelen kısımları çıkarmakta daha sonra istatistiksel model bitkiye karşılık gelen noktaları kümelemektedir. Sonuçlar ümit verici olmasına rağmen geliştirilen istatistiksel model mısır bitkisinin boyutuna ve şekline özel olduğundan yöntemin diğer bitkilerde kullanılabilmesi için genişletilmesi kolay değildir.

Kullanılan sensör tipinden bağımsız olarak ortamın belirsizliğinin açık şekilde ortaya konması ve yönetilmesi gerekmektedir. Hague ve Tillett (1996) olasılıksal bir yöntem olan Genişletilmiş Kalman Filtresi (EKF) yöntemini otonom tarımsal robot navigasyonu için kullanmışlardır. EKF yönteminin diğer bir uygulaması ise Southall ve ark. (2002) tarafından bitkilerin hücre desenine sahip şekilde büyüdüğü bir alanda navigasyon için kullanılmıştır.

Bu gelişmelere rağmen bu alanda daha birçok çalışmaya ihtiyaç duyulmaktadır. EKF yönteminin temel dezavantajı lineer olmayan sistemlerin lineer yaklaşığını kullandığı için bu tür sistemlerde hata, zamanın bir fonksiyonu olarak artmaktadır. İkinci olarak ise bu yöntemler spesifik bir özniteliğe dayalı ölçümlere bağlı olduğu için alınan veriler içerisinden önce bu özniteliğin çıkarılmasını gerektirir. Bu öznitelik çıkarma işlemi gerçekliğin bir özeti olacağı için belirsizliği arttıracaktır. Örneğin Hough Dönüşümünde düz çizgiyi bulmada yaşanan bir hata, navigasyonu başarısızlığa uğratacaktır. Bu probleme çözüm bulmak için Parçacık Filtresi (PF) geliştirilmiştir. PF, sistemi lineer olarak kabul etmez (Dellaert ve ark. 1999). Bu yüzden PF ortamın karmaşık ve ölçümlerin Normal Dağılıma sahip olmadığı ve hatta multimodal dağılıma sahip tarımsal uygulamalarda doğal bir seçim haline gelmektedir. Bergerman ve ark., (2012) PF temelli bir yöntemi meyve bahçelerinde otonom araç tasarımı için kullanmışlardır. Yapılan 300 km' lik test sürüşünün sonuçlarına göre gelecek vaadeden sonuçlar alınmıştır.

1.4. Olasılıksal Robotik

Robotik fiziksel dünyayı bilgisayar kontrollü cihazlar yardımıyla algılama ve manipüle etme bilimidir. Robotik sistemler sensörleri aracılığı ile dünyayı algılar ve uyguladıkları fiziksel kuvvetler ile manipüle ederler (Smith 2007). Bu görevler yerine getirilirken robotların fiziksel dünyadaki birçok belirsizlikle mücadele etmesi gerekmektedir. Bu belirsizlik kaynakları beş temel kısımda toplanabilir (Thrun 2005):

Çevre: Robotun çalıştığı çevre doğal yapısı gereği tahmin edilemez bir doğaya sahiptir (özellikle dış ortamda çalışan robotlar için). İnsanlarla iç içe çalışacak robotlarda belirsizlik oranı daha da yüksektir.

Sensörler: Sensörlerin algılayabildikleri şeyler sınırlıdır. Bu sınırlar çeşitli faktörlerden kaynaklanır. Sensörün menzili ve çözünürlüğü fiziksel sınırlarını oluşturur. Bunun yanı sıra sensörden alınan verilerde önceden tahmin edilemez şekillerde oluşan elektriksel gürültü sebebiyle, alınan verilerden çıkarılabilecek anlamlı bilgi de sınırlanmaktadır. Son olarak ise sensörler bozulabilir ve çalışma sırasında hatalı sensörlerin tespiti oldukça zordur.

Aktüatörler: Motor, solenoid valf gibi elemanları içeren aktüatörlerede oluşacak hatalara karşı önceden tahmin edilemez bir yapıya sahiptir. Kontrol gürültüsü, aşınma ve mekanik arızalar sebebiyle belirsizlikler ortaya çıkabilir.

Yazılım: Robotikteki belirsizliklerin bir kısmı da robot yazılımlarından kaynaklanır. Bütün modeller dünyanın basitleştirilmiş versiyonlarına göre elde edilmektedir. Sonuç olarak elde edilen model altta yatan fiziksel işlemin ve çevrenin sadece belirli bir kısmını ifade edebilir.

Algoritmik Kestirim: Diğer bir belirsizlik kaynağı ise algoritmik belirsizliktir. Robotlar gerçek zamanlı sistemlerdir. Bu nedenle yapabilecekleri bilgisayar hesaplamalarının bir sınırı vardır. Birçok popüler algoritma yaklaşık olarak kestirim yapmaktadır. Bu sayede işlem süresinden kazanılırken doğruluktan ödün verilmektedir.

Belirsizlik düzeyi uygulama alanına bağlıdır. İnsanların tüm çevreyi dizayn edebildiği üretim hattı gibi uygulamalarda belirsizlik sadece marjinal bir faktörken dış ortamda veya bina içerisinde çalışan robotlarda belirsizlik artmaktadır. Günümüzde robot sistemlerinde yukarıdaki sebepler ile açıklanan belirsizlik kaynakları robot sistemlerinin dizaynında dikkate alınması gereken en önemli faktörleri oluşturmaktadır (Burlina ve ark. 1991).

Olasılıksal robotik robotların algılama ve aksiyonlarındaki belirsizliği dikkate alan görece olarak yeni bir yaklaşımdır. Olasılıksal robotikteki temel yaklaşım olasılık teorisini kullanarak belirsizliği sayısal olarak ifade etmektir. Yani tek bir en iyi tahmine dayanmak yerine olasılıksal algoritmalar tüm tahmin uzayını olasılıksal dağılım şeklinde ifade eder. Böylelikle belirsizlik derecesi matematiksel olarak ortaya konmuş olur. Kontrol seçenekleri belirsizliğin oranına göre yapılabilir. Bu nedenlerden ötürü doğası gereği belirsizliğin yüksek olduğu dış ortam robotik uygulamalarında olasılıksal robotik yaklaşımı daha iyi sonuçlar vermektedir (Thrun 2005)

Olasılıksal robotik; robot modeli, kontrol uygulamaları ile sensör verilerinden kaynaklanan sınırlamaları aynı anda yöneterek birleştirir. Robotikte geleneksel olarak kullanılan model temelli hareket planlama veya reaktif davranış temelli yaklaşımların aksine olasılıksal robotik yaklaşımı sensör ve model sınırlamalarına karşı daha dayanıklıdır. Bu nedenle geçmiş paradigmalardan farklı olarak dış ortam uygulamalarında daha iyi bir şekilde ölçeklenebilirler. Sonuç olarak olasılıksal algoritmalar lokalizasyon ve büyük mekanların yüksek doğruluklu engel haritalarını çıkarmak konusunda yaşanan problemler için bilinen tek çözümdür (Ferreira ve Dias 2014).

Bununla beraber olasılıksal robotiğin bazı dezavantajları da bulunmaktadır. Bunlardan en sık karşılaşılan işlemsel karmaşıklıktır. Olasılıksal algoritmalar doğal yapıları gereği

rakiplerine göre bilgisayar işlem gücü kullanımı bakımından daha verimsizdir. Bu tek bir tahmin yerine bütün bir olasılıksal dağılımı göz önüne alınmasından kaynaklanmaktadır. Artan işlemci gücü nedeniyle bu sorun bir miktar çözülse de hala uygulamada karşımızda durmaktadır (Aznar ve ark. 2014).

Robotik alanı yazılım dizaynı açısından bir dizi paradigma serisini kullanmıştır. Bu paradigmalardan birincisi 1970' lerin ortalarında çıkan model temelli paradigmadır. Model temelli paradigma sürekli uzayda kontrol edilecek yüksek serbestlik derecesine sahip robotik manipülatörlerin kontrol zorluğunu gösteren çeşitli çalışmalarla başlamıştır (Reif 1979). Schwartz (1987) robot hareketinin karmaşıklığı konusundaki çalışmaları ve Latombe (1991) tarafından geliştirilen genel hareket planlama algoritması ile bu konudaki çalışmalar devam etmiştir. Yapılan bu erken dönem çalışmalarda sistemin belirsizliği büyük oranda göz ardı edilmiştir. Robot ve çevre modeli tam ve doğru kabul edilmiş ve robot deterministik olarak ele alınmıştır. Model yeterince doğru olmalıdır ve arta kalan belirsizlikler düşük seviye hareket kontrolörleri tarafından yönetilmeye çalışılmıştır. Manipülatörün kontrolü için birçok hareket planlama tekniği basitçe tekil bir referans yörüngesi üretirken görünmeyene cevap verebilmek için potansiyel alanlar (Khatib 1986) ve navigasyon fonksiyonları (Koditschek 1987) gibi olasılıksal çalışmalarda yapılmaya başlanmıştır.

Sensör geri bildirimi 1980 yılının ortalarından itibaren robotik çalışmaların odak noktasını oluşturmaya başlamıştır. Davranış temelli robotik çalışmalarında iç modeller yerine robotun fiziksel çevreyle interaksiyonu dikkate alınarak, karmaşık robot hareketleri üretilmiştir (Kaelbling ve Rosenschein 1990).

Anlık sensör bilgisinin kontrol için yeterli olacağı ve görece basit görevler için uygun olan davranış temelli paradigmanın daha karmaşık sistemler için uygun sonuçlar vermemesi üzerine model temelli ve davranış temelli paradigmaları aynı anda kullanan hibrid kontrol paradigmaları üzerinde çalışmalar yapılmıştır (Arkin 1998).

Modern olasılıksal robotik ise 1990' ların ortalarında ortaya çıkmıştır ancak temelleri Kalman filtresinin (Kalman 1960) bulunmasına dayandırılabilir. Olasılıksal robotikte aynı model temelli paradigmada olduğu gibi modeller olmasına rağmen bu modeller yetersiz ve eksik olarak kabul edildiği gibi, sensör temelli paradigmada olduğu gibi sensör ölçümleri olmasına rağmen bu ölçümlerde yetersiz ve eksik kabul edilmektedir. Kontrol aksiyonları

model ve sensör ölçümlerinin entegrasyonu ile sağlanmaktadır. İstatistik model ve sensör ölçümlerinin birbirine entegre edilmesi için bir yapıştırıcı katman görevi görmektedir.

Kalman filtre tekniğinin yüksek boyutlu algılamada kullanımı (Smith ve Cheeseman 1986), engel haritasının geliştirilmesi (Moravec 1989) ve parçalı gözlemle planlama tekniklerinin kullanımı (Kaelbling ve ark. 1998) olasılıksal robotik çalışmaları açısından önemlidir.

Parçacık filtresinin geliştirilmesiyle Kalman filtresi gibi parametrik filtrelerde ortaya çıkan multimodal algılama sorunlarına çözüm getirilmiştir (Dellaert ve ark. 1999). Bu algoritmaların yanı sıra algoritmaları daha etkin şekilde kullanmayı sağlayacak Bayes bilgi işleme odaklı programlama metodolojileri geliştirilmiştir (Thrun 1993, Lebeltel ve ark. 2004, Park ve ark. 2005)

1.5. Otonom Araçlarda Haritalama

Mobil otonom robotların en temel görevlerinden birisi harita oluşturmaktır. Literatürde mobil robot harita problemi eşzamanlı lokalizasyon ve haritalama (SLAM) problemi olarak ifade edilmektedir (Dissanayake ve ark. 2000, Murphy ve Russel 2001, Eliazar ve Parr 2004, Gutmann ve Konolige 1999, Hahnel ve ark. 2003, Montemerlo ve ark. 2003, Montemerlo ve ark. 2002). Robotun ortamdaki lokalizasyonu için haritaya, harita içinde lokalizasyona ihtiyaç duyulduğu için kompleks ve çözümü zor bir problemdir.

Moravec (1996) stereo kamera kullanarak 3 boyutlu çevreyi tanımlamışlardır. Thrun (1998) engel haritalarında çoklu sensör füzyonunu kullanmışlardır. Engel haritaları birçok farklı amaç için kullanılabilmektedir. Borenstein ve Koren (1991) engel haritalarını çarpışmadan kaçınma da kullanmayla ilgili çalışma yapmışlardır. Biswas ve ark. (2002) engel haritalarını dinamik ortamlarda hareketli nesnelerin şekillerini öğrenmek için kullanmışlardır. Dinamik nesnelerin hiyerarşik sınıf modelleri engel haritaları ile gösterilmiştir (Anguelov ve ark., 2004). Engel haritaları ayrıca eşzamanlı lokalizasyon ve haritalama problemlerinde kullanılmıştır.

Dış çevreyi hücreler halinde ifade etme mobil robotik literatüründe bulunan yersel gösterim yöntemlerinden sadece birisidir. Klasik hareket planlama çalışmalarında ortam poligonlar halinde gösterilir ancak bu modellerin eldeki verilerden nasıl elde edildiği ile ilgili

açıklar vardır (Schwartz ve ark. 1987). Kalman filtresinin sonar sensörden elde edilen bilgilere göre bir çizgi kestirimiyle ilgili ilk çalışma Crowley (1989) tarafından yapılmıştır. Anguelov ve ark. (2004) ham sensör verilerinden düz çizgili kapıları tanımlayarak algılama oranını arttırmıştır. Yersel gösterimde ilk olarak topolojik paradigma kullanılmıştır. Topolojik paradigmada dış çevre lokal ilişkiler ile ifade edilmiş bu sayede robot birbirine komşu lokasyonlar arasında hareket etmiştir (Kuipers ve ark. 2004). Engel haritaları ise metrik gösterime sahip tamamlayıcı bir paradigmadır. Metrik gösterim sayesinde robotunun bulunduğu ortam kesin koordinat sistemi üzerinde doğrudan tanımlanabilir. Thrun (1998) metrik engel haritasından topolojiik altyapıyı çıkararak daha hızlı hareket planlama yapmıştır.

Doucet ve ark. (2001), eşzamanlı lokalizasyon ve haritalama probleminin çözümü için Rao-Blackwellized parçacık filtresi çözümünü geliştirmişlerdir. Rao-Blackwellized yaklaşımının uygulamasındaki temel problem uygun bir harita geliştirmek için gerekli olan parçacık sayısına göre bilgisayarın işlemsel karmaşıklığının artmasıdır. Bu nedenle parçacık sayısının mümkün olduğunca az tutulması bu algoritmanın en temel problemidir. Ek olarak yeniden örnekleme sırasında doğru parçacıkların elenmesi söz konusu olabilir. Bu etki parçacık fakirleşmesi olarak bilinmektedir (Merwe ve ark. 2000).

1.6. Otonom Araçlarda Lokalizasyon

Lokalizasyon mobil robotlara otonom hareket sağlamakta karşılaşılan en önemli problemdir (Cox ve ark. 1990). Dış ortam robotlarında konum tahmini için EKF' nin kullanımı Dickmanns ve Graefe (1988) tarafından yapılan çalışmada kamera görüntülerinden otoyoldaki virajların tahmini için kullanılmıştır. Bu çalışmaların birçoğunda ortamın lokalizasyon için yapay işaretleyicilerle modifiye edilmesi gerekmiştir. Leonard ve Durrant-Whyte (1991) çalışmasında ortamın geometrik haritasından tahmin edilen noktalar ile sonar taramadan çıkarımlanan geometrik noktaları eşleştirmek için EKF kullanmıştır. Kullanımı ekonomik ve fizibil olan durumlarda yapay işaretleyicilerin kullanımı daha sonraki çalışmalarda da görülmüştür (Salichs ve ark. 1999).

Noktasal özniteliklere dayalı lokalizasyon çalışmalarından sonra araştırıcılar lokalizasyon için daha geometrik yöntemler geliştirmişlerdir. Cox (1991) ortamın doğru parçası ile infrared sensörden ölçülen mesafeyi eşleştiren bir algoritma geliştirmiştir. Weiss ve ark. (1994) mesafe ölçümlerini lokalizasyon için ilişkilendiren bir yöntem geliştirmişlerdir. Lokal

ve global engel haritasının karşılaştırılmasına harita eşleme denilmektedir (Moravec, 1989). Thrun (1993) tarafından geliştirilen gradyan inişli yer saptayıcı, harita eşleme fikrine dayanmaktadır. Schiele ve Crowley (1994) ultrasonik sensör ve engel haritalarına bağlı olarak robotun pozisyonunu takip etmekle ilgili çalışmalar yapmışlardır. Lokal engel haritası ve global hücre haritası karşılaştırıldığında bu eşleştirme her iki haritadaki özniteliklere uygun yapılmışsa benzer sonuçların elde edildiğini bulmuşlardır. Shaffer ve ark. (1992) harita eşleştirme ve öznitelik temelli teknikleri karşılaştırmış ve her iki yönteminde beraber çalıştığı deneysel sonuçların daha iyi olduğunu rapor etmiştir. Arras ve Vestli (1998) tarama eşleştirme ile yüksek doğruluklu olarak lokalizasyonun yapılabileceğini göstermiştir. Ortin ve ark. (2004) lazer mesafe ölçer verisi yanında kamera verisini de kullanarak tarama eşleştirmenin menzilini arttırmıştır.

Mobil robot lokalizasyonunda bulanık mantık kullanılmıştır (Saffiotti, 1997; Driankov ve Saffiottti, 2001). Simmons ve Koenig (1995) sonsal olasılıkları gösteren hücreleri kullanarak lokalizasyon yapan Markov Lokalizasyon yöntemini geliştirmişlerdir. Bu yöntemle çoklu hipotez tahminine olanak tanınmıştır. Hücre temelli yapılan ilk çalışmalarda görece olarak daha geniş hücreler kullanıldığı için bu hücreleri güncellemek yüksek hesap gücü gerektirmedir. Burgard ve ark. (1998) hücrelerin daha yüksek çözünürlükte olması için selektif güncelleme yöntemini geliştirmiştir. Bu sayede topolojik Markov lokalizasyonundan detaylı metrik lokalizasyona geçiş mümkün olmuştur (Thrun 2001). Bu algoritma lağım kanallarında çalışan robotlar (Hertzber ve Krichner, 1996), ofis ortamında lokalizasyonun sağlanması (Simmons ve Koenig, 1995) ve müzede çalışan robotlarda pozisyon tahmini (Burgard ve ark., 1998) gibi çalışmalarda kullanılmıştır. Konolige ve Chou (1999) hızlı konvolüsyonel tekniklerini robotun konum olasılıklarının hesaplanmasında kullanarak Markov Lokalizasyonda harita eşleştirmenin kullanılması fikrini ortaya atmışlardır. Burgard ve ark. (1998) yüksek doğruluklu takip ve global lokalizasyon imkanı sağlayan dinamik Markov lokalizasyon tekniğini geliştirmişlerdir. Hücre temelli paradigma için aktif lokalizasyon Fox ve ark. (1997) tarafından geliştirilmiştir.

Parçacık filtreleri takip ve lokalizasyon problemlerinde yaygın olarak kullanılmaktadır. Montemerlo ve ark. (2002) eş zamanlı lokalizasyon ve insan takibi için rekürsif olarak parçacık filtresi kullanmıştır. Parçacık filtresi birden fazla sayıda insanın takibi için Schulz ve ark. (2001) tarafından kullanılmıştır. Schulz ve ark. (2001a) bilinmeyen bir ortamda hareketli bir robot ile hareketli insanların güvenilir bir şekilde takip edilebileceğini göstermiştir.

1.7. Otonom Araçlarda Rota Planlama

Robotik içerisinde rota ve hareket planlama genellikle olasılıksal olmayan bir çerçeve içerisinde ele alınmıştır. Bu yöntemlerde robot ve çevresinin mükemmel bir şekilde bilindiği ve kontrol sonuçlarının deterministik olduğu kabul edilmektedir. Ancak durum uzayının sürekli ve çok boyutlu olması sebebiyle problemler ortaya çıkmaktadır (Latombe 1991). Bu konuda ilk çalışmalar Lozano-Perez ve Wesley (1979) tarafından görsel diyagramlar, Khatib (1986) tarafından potansiyel alan kontrolü ve Canny (1988) tarafından silüet algoritması ile yapılmıştır. Guibas ve ark. (1992) voronoi diyagramlarının efektif bir şekilde hesaplanmasıyla ilgili çalışma yapmışlardır. Kavraki ve ark. (1996) rotanın randomize tekniklerle hesaplanması konusunda çalışmıştır. Reif (1979) sürekli kontrol uzayının sonlu sayıda hücreye dekompozisyonuyla hareket planlama ile ilgili teknikler geliştirmiştir.

Robotikte rota ve hareket planlama algoritmaları çeşitli arama algoritmalarının (A* arama) farklı uygulamalarda kullanılmasıyla başlamıştır (Brooks ve Lozano-Perez 1985). Yapılan bu çalışmalarda çokyüzlü veya polygonal engeller için konfigürasyon uzayı serbest bölge, engeller ve karışık küboidlerden oluşan diyagramlara dönüştürülmüş. Daha sonra bu diyagramlar üzerinde A* arama algoritması uygulanarak serbest düğümler arasında lineer yollar aranmıştır (Brooks ve Lozano-Perez 1985, Fan ve Lui 1994, Kavraki ve Latombe 1994).

Takahashi ve Schilling (1989) dikdörtgen bir robot için genelleştirilmiş voronoi diyagramında buluşsal arama yaparak rota ve hareket planlama konusunda çalışma yapmıştır. Voronoi diyagramı (Donald 1984) yersel bir diyagram üzerinde terminal noktalarını seçerek oluşturulan bir serbest uzay diyagramıdır. Böylece boyut azaltılması sağlanır. Donald (1987) A* en kısa rota arama algoritmasını çok yüzeyli engellerde rota ve hareket planlama için kullanmıştır. Olasılıksal algoritmalar olan potansiyel alan algoritması, yol haritası algoritması, güncellenmiş A* arama algoritması ve genetik algoritmalar kullanılmaya başlanmıştır (Khatib 1986, Warren 1993).

1.8. Çalışmanın Hedefleri

Bu tez çalışmasında, olasılıksal robotik yöntemlerin, çekme ve taşıma gibi tarımsal işlemlerde, özellikle de değişken düzeyli ilaçlama uygulamasında kullanılabilecek otonom bir tarım aracının tasarımında kullanılması amaçlanmıştır.

Ortamın doğal belirsizliği nedeniyle tarımsal işlemlerde kullanılacak robotların otonom navigasyonu zor bir işlemdir. Mevcut sistemlerin en önemli dezavantajı bu belirsizliklere karşı dayanıklı olmamalarıdır. Bu çalışmada LIDAR (Lazer algılama ve mesafe ölçüm sensörü) ve odometri sensörü temelli bir yöntem kullanılarak bu probleme çözüm geliştirilmiştir. Navigasyonda amaç, dış ortamda bulunan tel terbiyeli bir meyve bahçesinde, otonom olarak iki ağaç sırası arasındaki koridorda gidebilecek ve verilen rotanın takip edilmesini sağlayacak bir yazılım sisteminin tasarlanmasıdır. Sistemin temelini parçacık filtresi yöntemine dayanan haritalama, lokalizasyon, rota planlama ve ilaçlama sisteminin kontrolü aşamaları oluşturmaktadır. Çalışma aşağıdaki maddeleri hedeflemiştir:

Haritalama: Gerçek zamanlı haritalama ve lokalizasyon yazılımı geliştirmek. Odometri ve LIDAR sistemine sahip otonom bir aracın çevresindeki engelleri ve bunlara olan mesafelerini algılayarak bu engellerin bir haritasını çıkarabilmesi

Konum Belirleme: Haritalaması yapılan ortamda LIDAR ve odometri sensöründen gelen verileri harita ile karşılaştırarak otonom aracın harita üzerinde lokalizasyonun sağlanması

Rota Planlama: Harita üzerinde lokalizasyonu yapılarak konumu belirlenen otonom aracın verilen hedef veya hedeflere gidebilmesi için rota planlaması yapabilmesi ve otonom araç, hedefine gitmek üzere rota üzerinde ilerlerken harita üzerindeki engellerden kaçınabilmesi

İlaçlama Sisteminin Kontrolü: Robotun harita üzerindeki konumunun hassas bir şekilde tespit edilerek haritanın belirli noktalarında değişken düzeyli ilaçlama sisteminin çalıştırılması diğer konumlarda ise kapalı kalması

2. KAYNAK ÖZETLERİ

Bu bölümde; Tarımda Otonom Navigasyon, Olasılıksal Robotik, Otonom Araçlarda Lokalizasyon, Otonom Araçlarda Haritalama, Otonom Araçlarda Rota Planlama ve Navigasyon ile ilgili çalışmalar hakkında bilgiler verilmiştir.

2.1. Tarımda Otonom Navigasyon

Singh ve ark. (2005) tarımsal üretimde verim üzerinde önemli etkisi olan ve karlılığı arttıran fungusit, pestisit ve gübre uygulaması için seralarda kullanılacak otonom bir robot geliştirmişlerdir. Otonom araç kullanımı ile uygulanan kimyasalın hedefe ulaştırılmasını sağlamak, işçi ihtiyacını ve işçilerin kimyasala maruz kalmaları nedeniyle uğrayacağı zararı minimize etmektir. Doğru akımlı elektrik motorları ile hareket verilen robot platformu ultrasonik sensörlerden gelen veriler ve bulanık mantık kullanılarak oransal kontrolör ile kontrol edilmiştir. Otonom robot sıra aralarında 2,5 cm ortalama hata ile hareket edebilmiştir.

Cheein ve Carelli (2013) tarımsal işlemlerin insansız robotik araçlarla yapılması olanakları konusunda bir araştırma yapmışlardır. Hassas otonom tarımın operasyon, navigasyon ve otonom araçların tarımsal işlemleri gerçekleştirmesi için kontrol aşamalarından oluştuğunu bildirmişlerdir. Otonom tarımsal araçların hassas tarım uygulamarının merkezinde bulunacağını belirtmişlerdir.

Malavazi ve ark. (2018) otonom tarımsal robot için LIDAR sensör temelli bir navigasyon algoritması geliştirmişlerdir. Yeni geliştirilen bu algoritma sensörden alınan ölçüm noktalarından bitki sıralarını çıkarmak için bir yöntem içermektedir. Çizgi tespitine dayalı bu navigasyon algoritması yabancı ot robotunda test edilmiştir.

Bonadies ve Gadsden (2018) insansız yer araçları ile bitki sıralarının otonom olarak takip edilmesinin tarımda kullanım alanlarını araştırmışlardır. Otonom araçların tarım endüstrisindeki insan kaynağı yetersizliğine çözüm olabileceğini ve tarımsal üretimdeki gıda güvenliğini sağlayabileceğini belirtmişlerdir. Bu araçların hayvan dışkısı tespiti, bitki büyümesinin izlenmesi, fırtına veya sel nedeniyle ortaya çıkan bitki zararının tespiti, istenmeyen zararlı ve mantari hastalıkların tespitinde kullanılabileceğini rapor etmişlerdir.

Fernandez ve ark. (2018) nokta dönüşlü tarımsal robotlar için dayanıklı bir dijital kontrol sistemi geliştirmek için çalışma yapmışlardır. Birinci derece model yaklaşımı kullanan

bu model ile kontrol sisteminin gerçek zamanlı gömülü sistemlerde çalışabilmesini sağlamışlardır. Sistem diferansiyel hızı sistem girdisi olarak almakta ve lateral pozisyonu çıktı olarak vermektedir. Dijital dayanıklı tasarım sayesinde farklı zemin özelliklerinde en iyi kontrol sonucu alınmaya çalışılmıştır.

Vroegindeweij ve ark. (2018) tavukçulukta kümes içi yumurta toplamak için kullanılan otonom bir robotun performans değerlendirilmesi konusunda bir çalışma yürütmüşlerdir. Yapılan çalışmada otonom robotun navigasyon ve yumurta toplama başarısı değerlendirilmiştir. Ayrıca karşılaşılan engellerden sakınma farklı rota tipleri üzerindeki tepkisi incelenmiştir. Yerden yumurta toplamada %46 başarı elde edilmiştir.

Gan ve Lee (2018) akıllı çiftlik için navigasyon sistemi geliştirme üzerine bir çalışma yapmışlardır. Çalışmada bir başlangıç istasyonundan hareket eden robotun turunç bahçesindeki herbir ağacı otonom olarak ziyaret etmesini sağlayacak ve yol üzerindeki engellerden sakınmasını sağlayacak bir sistem geliştirilmiştir.

Liu ve ark. (2018) tarımsal araçların otonom navigasyonunda küçük engellerden sakınma üzerine bir araştırma gerçekleştirmişlerdir. Otonom aracın statik engellere çarpmaması için minimum dönüş açısını hesaplayan bir sistem geliştirilmiş ve bir simülasyon programı üzerinde test edilmiştir.

Astolfi ve ark. (2018) bağlarda otonom navigasyon üzerine bir çalışma yapmışlardır. Çevre kirliliği ve verimi arttırması nedeniyle hassas tarımın bilinirliğinin artmasının tarımsal robotlarda hızlı bir gelişme sağladığını bildirmişlerdir. Atalet sensörü, GPS ve LIDAR sensör kullanarak bağın haritalanması sağlanmış ve adaptif monte karlo lokalizasyonu kullanılarak lokalizasyonu sağlanmıştır. Simülasyondan elde edilen sonuçlar arazi testlerinde doğrulanmıştır.

Gruzauskas ve ark. (2018) otonom araçlar kullanımında sürdürülebilirlik ve maliyet arasındaki ters ilişkiyi minimize etmek üzerine bir çalışma yapmışlardır. Tarımsal üretim zincirinde endüstri 4.0 uygulamalarının eksikliği tespit edilmiştir. Büyük veri analizi, bulut hesaplama, siber-fiziksel sistemlerin kullanıldığı otonom araçların uzun vadede işletmelere rekabet avantajı sağlayacağının altını çizmişlerdir. Otonom araçların kullanılmasının CO_2 salınımını %22 azaltacağını ortaya koymuşlardır.

2.1.1 Görüntü temelli navigasyon

Hiremath ve ark. (2014) görüntü temelli parçacık filtresi uygulamasının yarı düzenli tarımsal ortamda kullanmışlardır. Çalışmada parçacık filtresine dayalı otonom navigasyon algoritması geliştirilmiş ve ortam sadece monoküler kamera kullanılarak algılanmıştır. Parçacık filtresi kullanımı belirsizliği azaltmış ve güvenilir bir navigasyon algoritması geliştirilmiştir. Yapılan 5 km'lik arazi sürüş testinde robot bitki koridorunun içerisinden bitkiye dokunmadan geçebilmiştir.

Campos ve ark. (2016) tarımsal alanların video görüntüleri üzerinden mekan-zamansal analiz yöntemi kullanarak engel tespiti konusunda bir çalışma yapmışlardır. Otonom navigasyonda video görüntüsü üzerinden engel tespiti yapılması için gerçekleştirilen çalışmada mekan-zamansal analiz hareketli ve hareketsiz nesnelerin tespitini sağlamıştır.

Zhang ve ark. (2017) pirinç tarımında yabancı otla mücadele için kullanılan bir robot için alınan kamera görüntülerini değerlendirerek navigasyon için kullanan bir algoritma geliştirmişlerdir. Geliştirilen yöntem görüntülerin siyah beyaza dönüştürülerek segmentasyonuna dayanmaktadır. SUSAN köşe algılama algoritmasına göre navigasyon hattı belirlenmektedir. Sistem simülasyon ortamında tasarlanmış ve denenmiştir.

Radcliffe ve ark. (2018) şeftali bahçesinde robotik bir aracın otonom navigasyonu için görüntü temelli bir sistem geliştirmişlerdir. Kameranın gökyüzüne doğru konumlandırılması ile ağaç kanopisi ve gökyüzünü kullanarak navigasyon sağlanmış ve bu yeni yaklaşım ortaya konmuştur. Geliştirilen algoritma küçük boyutlu bir robot için meyve bahçesinde başarıyla uygulanmıştır. Sistem ağaç sıralarını ortalama 2,13 cm hata ile takip edebilmektedir.

Yang ve ark. (2018) görüntü temelli bir yöntem kullanarak mısır sıralarında orta noktanın tespiti üzerine bir çalışma yapmışlardır. RGB formatındaki görüntüler üzerinden robotun mısır sırası üzerindeki konumu belirlenmiş ve sıranın orta noktası bulunmuştur. Sistem %92 başarı ile orta noktayı bulabilmektedir. Bu yöntemin mısır ekili alanlarda kullanılacak otonom tarım araçları için uygun olduğu belirtilmiştir.

Ericson ve Astrand. (2018) iki farklı görsel odometri yönteminin tarımsal çevrede kullanılması ile ilgili bir çalışma yapmışlardır. Görsel odometri yöntemlerinin sonuçları değerlendirilmiştir. %3,76 pozisyon ve 0,0482 derece/dk oryantasyon hatası gözlenmiştir.

Zhang ve ark. (2018) binoküler görüntü ile traktörler için rota takip sistemi geliştirmişlerdir. Alınan binoküler görüntü ile bitki sıraları algılanarak PID kontrol algoritması ile bu sıraların takibi sağlanmıştır. Pamuk bitkisi için yapılan denemelerde 4 cm pozisyon ve 0,95 derece açısal hata tespit edilmiştir.

2.1.2. Mesafe sensörü temelli navigasyon

Weiss ve Biber (2011) tarımsal robotlarda 3 boyutlu LIDAR sensörü kullanarak bitki algılama ve haritalama konusunda araştırma yapmışlardır. Robotik uygulamalardaki 3 boyutlu sensör teknolojileri karşılaştırılmış düşük çözünürlükü 3 boyutlu LIDAR sensör ile güvenilir bir bitki algılama yöntemi geliştirilmiştir. Algoritma simülasyon ortamı Gazebo' da geliştirilmiştir. Karşılaştırılmalı deney sonuçları simülasyon ve gerçek dünyadan alınan sonuçlarla yapılmıştır.

Reina ve ark. (2015) robotik araçlarda ortam farkındalığının sağlanması için LIDAR, radar ve termografi sensörlerinin kullanımı ile ilgili çalışma yapmışlardır. Birden fazla sensör kullanımının sürüş güvenliğini arttırdığını belirtmişlerdir.

Underwood ve ark. (2016) badem ağaçlarının kanopi hacmi, çiçek, meyve ve verim haritalamasını gerçekleştirmek için LIDAR ve görüntü sensörü kullanımı konusunda çalışma yapmışlardır. Haritalama için uygun bir yazılım sistemi geliştirilmiştir. 580 ağaç üzerinde yılda 3 kez ölçüm alınarak yapılan ve 2 yıl süren çalışma sonunda kanopi hacmi ve ürün verimi arasında lineer bir ilişki tespit edilmiştir (R^2=0,77)

Meng ve ark. (2018) LIDAR ölçümleri ve spektral görüntüleme ile ağaç kanopisindeki zararlılar tarafından kaynaklanan yaprak dökülmelerinin ağaç bazında haritalanması konusunda çalışmışlardır.

Pierzchala ve ark. (2018) 3 boyutlu LIDAR ölçümleri ve grafik-SLAM algoritması kullanarak ormanların haritalanmasını sağlayacak insansız araç konusunda çalışmışlardır. Çalışmada grafik-SLAM algoritmasının orman haritalama için verimli bir yöntem olduğu ortaya konmuştur. Oluşturulan harita üzerinden ağaçların pozisyonları ve çapları elde edilebilmektedir. Ağaç çapının 2 cm hata ile ölçülebildiği görülmüştür.

Meichen ve ark. (2018) LIDAR, pusula ve atalet sensör verilerinin füzyonu ile dinamik engel tespiti konusunda çalışma yapmışlardır. Kalman filtresi temelli füzyon yöntemiyle bu üç sensörün verileri birleştirilmiş ve dinamik engellerin tespiti sağlanmıştır.

Westling ve ark. (2018) LIDAR sensör kullanarak avokado ağaçlarının ışık alma modellerini çıkaran bir çalışma yapmışlardır. Geliştirilen model meyve ağaçlarının ışığı absorbe etme ve dağıtma miktarını tahmin etmekte kullanılmaktadır. LIDAR sensör ölçümleri ile her bir ağaca ait geometri model için dijital bir değere dönüştürülmüştür. Tüm sezon boyunca ölçüm alınabilmektedir. Model başarısı R^2=0,854 olarak ölçülmüştür.

2.2. Otonom Araçlarda Haritalama

Cheein ve ark. (2011) hassas tarımda ağaç gövde tespitine göre haritalama için optimize edilmiş bir haritalama algoritması geliştirmişlerdir. Tarımsal işlemlerde kullanılan robotlar için harita kullanımının zorunlu olduğunu belirtmişler ve bu amaçla gerçek zamanlı haritalama ve lokalizasyon yapan bir algoritma ön tekerleklerinin döndürülmesiyle hareket eden araba tipi bir robot üzerinde test edilmiştir. Ortam haritası zeytin ağaçlarının gövdesi LIDAR ve monoküler görüntü sensörüyle tespit edilerek oluşturulmuştur. Oluşturulan ortam haritası ile robot ağaç sıraları arasında hareket edebilmiştir.

Shih ve Lin (2013) gerçek zamanlı haritalama ve lokalizasyon (SLAM) yöntemini kullanan otonom bir robot ile ortamın 3 boyutlu yeniden yapılandırmasını yapmak üzere bir çalışma yapmışlardır. Otonom robotun rota planlama, lokalizasyon ve haritalama gibi karmaşık işleri yapabilmesi için çevre yapısını anlaması gerekliliğini bildirmişlerdir. Geliştirilen yöntem ile 55,94 cm hata ile robot pozisyonu bulunabilmektedir. Bu yöntemin tarımda arazi inceleme, yapay gerçeklik ve coğrafi bilgi sistemlerinde kullanılabileceğini önermişlerdir.

Mutz ve ark. (2016) otonom bir araç için kompleks bir çevrede büyük boyutlu haritalama konusunda çalışma yapmışlardır. 3 boyutlu LIDAR sensörden alınan veriler ile 3 farklı ortam (park alanı, yaya üst geçiti ve şehir içi trafiği) için haritalama simülasyonu gerçekleştirilmiştir.

Andersone (2017) ultrasonik sensör kullanarak olasılıksal haritalama konusunda çalışmıştır. Ultrasonik sensörden alınan veriler haritalamadaki belirsizliği azaltmak için olasılıksal haritalama yöntemini kullanarak normal rastgele dağılıma göre modellenmiştir. Alınan verilerden hücresel engel haritası oluşturulmuştur.

2.3. Otonom Araçlarda Lokalizasyon

Gamallo ve ark. (2010) omnivizyon temelli Kullback-Leibler Distance (KLD)-Monte Karlo Lokalizasyon uygulaması geliştirmişlerdir. Monte Karlo Lokalizasyon algoritmasının LIDAR sensör ile başarıyla uygulandığı bildirilmiştir. Lokalizasyon ve robot kaçırma problemi üzerinde yapılan testler lokalizasyon sistemin gerçek uygulamada başarıyla kullanılabildiğini göstermiştir.

Guan ve ark. (2018a) doppler-azimuth radar kullanan bir robot için monte karlo lokalizasyon uygulaması geliştirmişlerdir. Geliştirilen çözüm ile doplller-azimuth radar ve odometri verileri kullanılarak lokalizasyon yapılabilmiştir. Adaptif Monte Karlo yönteminin kullanılmasının ihtiyaç duyulan bilgisayar işlem gücünü azalttığı rapor edilmiştir.

Guan ve ark. (2019) mobil robotlar için adaptif monte karlo lokalizasyon algoritmasını kullanmışlardır. Adaptif parçacık sayısı kullanımı ile lokalizasyonda performans artışı ve ihtiyaç duyulan parçacık sayısında azalma gözlenmiştir. Realistik bir senaryo için simülasyon ortamında odometri verisine patinaj eklenerek ölçümler yapılmıştır.

2.4. Otonom Araçlarda Rota Planlama

Henkel ve ark. (2016) mobil servis robotlarında lokal rota planlaması için enerji verimli dinamik pencere yaklaşımını kullanmışlardır. Lineer regresyon modeli ile rota planlama sırasında enerji tüketimi izlenmiş ve tüketilen enerjide %9,79 azalma olduğu görülmüştür.

Niu ve ark. (2018) insansız yer araçları için enerji verimli bir algoritma geliştirmişlerdir. Çalışmada voronoi diyagramı, dijkstra algoritması yaklaşımları enerji tüketimi bakımından karşılaştırılmıştır.

Lamini ve ark. (2018) otonom mobil robotlar için rota planlamada genetik algoritma yaklaşımını kullanan bir çalışma yapmışlardır. Statik ortamda genetik algoritma kullanarak rota planlama uygulaması yapılmıştır. İki nokta arasında engellerden sakınma sağlayacak ve rota uzunluğuna göre optimize edilmiş simülasyon sonuçları rota planlamada genetik algoritma yaklaşımının kullanılabileceğini göstermiştir.

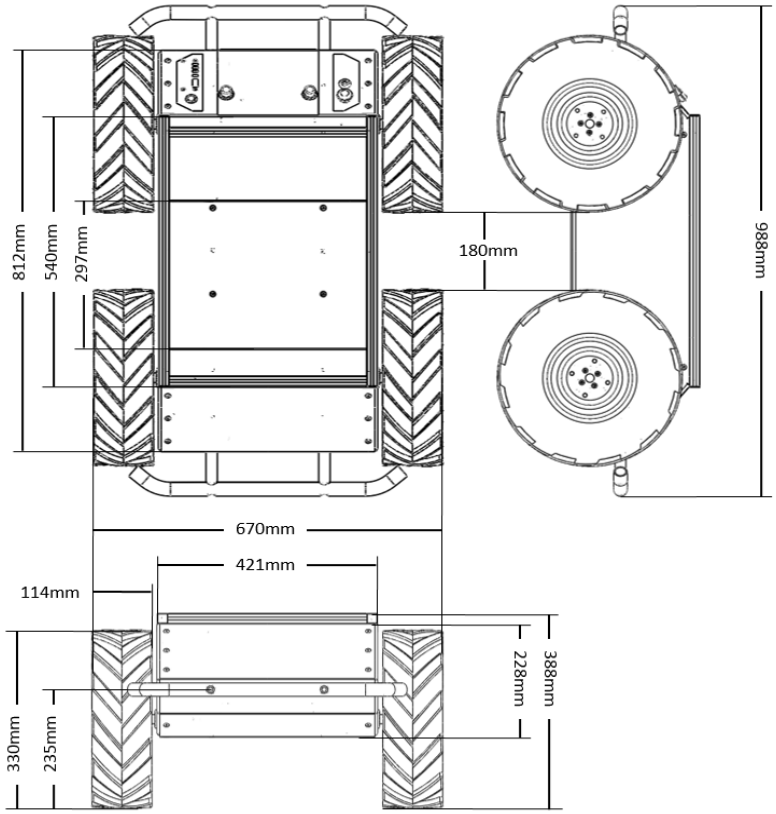
Prado ve ark. (2018) otonom araçların farklı zeminlerde performansını arttırmak amacıyla olasılıksal kendi-ayarlama yaklaşımı konusunda çalışma yapmışlardır. Çalışmada otonom araçların performansının tekerlek ve yüzey ilişkisine bağlı olduğunu ortaya koymuş ve

yapılacak olasılıksal kendi-ayarlama yaklaşımı ile uygulanan otonom kontrolün %75 iyileştirilebileceğini ortaya koymuşlardır.

Guo ve ark. (2019) ölçülebilir bozucu etkiler gözönüne alınarak otonom araç için model tahmin kontrol yöntemini kullanarak rota planlama yapmışlardır. Model tahmin kontrol yöntemi için bir tasarım şeması geliştirilmiştir. Farklı yol koşulları ve model kabulleri bozucu etkiler olarak gözönüne alınmıştır.

3. MATERYAL ve YÖNTEM

3.1. Simülasyonda Kullanılan Otonom Araca Ait Genel Ölçüler



Şekil 3.1. Tasarlanan otonom araca ait genel ölçüler

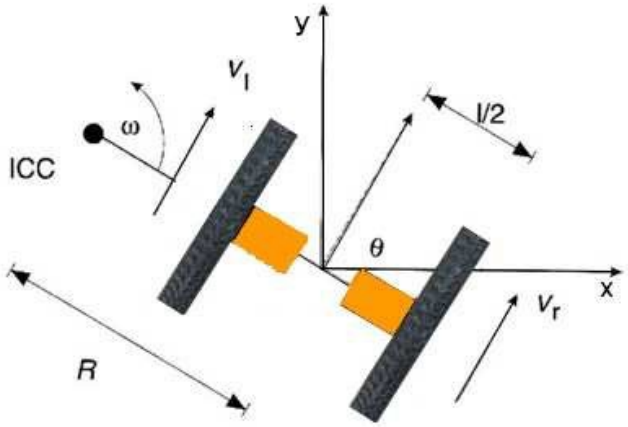
Tasarımı yapılan otonom araç platformu 990 mm uzunluğa, 670 mm genişliğe ve 390 mm yüksekliğe sahiptir. Dingil arası mesafe 512 mm, kullanılacak ekipmanlar ile baz ağırlığı 50, taşıdığı ağırlık 200 kg aynı şekilde simülasyon üzerinde tanıtılmıştır. Yerden yüksekliği 130 mm' dir (Şekil 3.1). İlaçlama deposu ve baz ağırlığı ile toplam 250 kg ağırlığa sahip olacak otonom araçta ortalama 3 saat sürekli çalışma için 6 kW batarya kullanımı yeterli olacaktır.

Simülasyonda hareket sol ve sağda bulunan tekerlekler için birer adet 1kW gücündeki fırçasız DC elektrik motorları ile sağlanmıştır. 330 mm çapa sahip, çapa makinası tekerleği kullanılmıştır. Tasarlanan otonom araç üzerinde odometri verisini sağlayacak atalet sensörü ve sol ve sağ taraftaki tekerleklere ait devir verilerini ölçmek amacıyla 2 adet devir ölçer sensör

kullanılmıştır. Çevredeki cisimlerin otonom araca olan mesafelerini ölçmek amacıyla LIDAR sensör kullanılmıştır.

3.2. Otonom Araca Ait Kinematik Model

Harekete etkili kuvvetleri dikkate almadan hareketi inceleyen matematik dalına kinematik denilmektedir (Joseph 2018). Şekil 3.2' de tasarlanan otonom araca ait kinematik değerler gösterilmiştir.



Şekil 3.2. Tasarlanan otonom aracın kinematik özellikleri

Otonom aracın sahip olduğu pozisyon X matrisi ile ifade edilmiştir.

$$X = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \tag{3.1}$$

Sağ ve sol tekerlek hızlarına göre otonom aracın ilerleme hızı, açısal hızı ve dönüş yarıçapı aşağıdaki eşitlikler ile ifade edilmiştir (Çelen ve ark. 2015).

$$\mathrm{v(t)} = \frac{1}{2}(\mathrm{v_r(t)} + \mathrm{v_l(t)} \tag{3.2}$$

$$\mu(t) = \frac{(v_r(t) - v_l(t))}{l} \tag{3.3}$$

$$R = \frac{(v_r(t)+v_l(t))}{2*(v_r(t)-v_l(t))} \quad (3.4)$$

Burada;

$v_r(t)$: Sağ tekerleğin doğrusal hızı (m/s),

$v_l(t)$: Sol tekerleğin doğrusal hızı (m/s),

v(t): Doğrusal hız (m/s),

μ: Açısal hız (rad/s),

l: Tekerlek eksenlerinin birbirine olan mesafesi (cm),

R: Otonom araç orta eksenine göre dönüş yarıçapıdır, (cm) (Çelen ve ark. 2015).

Otonom araca ait bu kinematik özellikler aracın simülasyonda tanımlanan hızlanma, maksimum hız, araç boyutları, dönüş merkezi, tekerlekler arası mesafe ve diferansiyel sürüş özellikleri sayesinde simülasyon ortamı tarafından otomatik olarak hesaplanmıştır.

3.2.1. Hız ve ivmelenme

Otonom aracın hız ve ivmelenme dinamikleri rota planlamanın ikinci aşamasını oluşturan lokal planlama aşaması için önemlidir. Lokal planlayıcı odometri verilerini girdi olarak almakta ve çıktı olarak hız ve dönüş komutu vermektedir. Otonom aracın maksimum ve minimum hız değerleri ile ivmelenme değeri için uygun değerlerin seçilmesi lokal planlamanın uygun yapılması için önemlidir. Kullanılan LIDAR sensör 1cm çözünürlüğe sahip olduğu için ilaçlamada yaygın olarak kullanılan 1 m/s - 2,5 m/s hızları tasarlanan sistem için uygundur. Ölçüm standardı sağlamak amacıyla simülasyonda otonom aracın hızı sabit 1 m/s' ye ayarlanmıştır. İvmelenme değeri 1 m/s^2 ' ye ayarlanmıştır.

3.2.2. Maksimum hız ölçümü

Prototip gerçekleştirilirken, maksimum hız ölçümü düz bir çizgi üzerinde giderken ulaşılabilen en yüksek hız gözlenerek maksimum hız belirlenebilir. Açısal hız ise otonom araç sabit hıza ulaştığında yapılan 360°' lik hareket süresi gözlenerek rad/s olarak bulunmuştur. Hem açısal hız hemde maksimum hız geliştirilen programdaki yayınlanan odometri mesajı izlenerek ölçülebilir. Bu değerler ayarlanırken güvenlik açısından maksimum değerlerinin altında bir değer seçilmesi performans bakımından olumludur.

3.2.3. Maksimum ivme ölçümü

Programdan yayınlanan odometri mesajları zaman etiketi de içermektedir. Maksimum doğrusal hıza (t_d) ve maksimum açısal hıza (t_r) ulaşmak için geçen süre birden fazla ölçümün ortalaması alınarak bulunmuştur. Bu sayede maksimum doğrusal ve açısal ivmelenme sırasıyla aşağıdaki gibi hesaplanmıştır (Joseph 2018).

$a_{d,max} = \max d_v/d_t \sim v_{max} / t_d \quad (m/s^2)$ (3.5)

$a_{r,max} = \max d_\mu/d_t \sim \mu_{max}/t_r \quad (m/s^2)$ (3.6)

3.3. Simülasyonda Kullanılan Sensörler

3.3.1. LIDAR sensör

Simülasyonda lazer algılama ve mesafe ölçüm sensörü-LIDAR (LMS-111, Sick AG, Waldkirch, Almanya) kullanılmıştır (Şekil 3.3). Sensör otonom araç platformunun ön kısmına yerden 400 mm yükseklikte olacak şekilde, simülatör içerisinde monte edilmiştir. LIDAR lazer ışının gönderilmesi ve geri alınması arasında geçen uçuş süresi prensibine bağlı olarak ölçüm yapar. Bir lazer diyot aracılığıyla lazer ışın demeti yayılır. Lazer palsi bir nesneye çarpıp geri döndüğünde bu dönen pals sensör üzerindeki foto diyot ile algılanır. Palsin gönderilmesi ve yansıdıktan sonra algılanana kadar geçen süreden sensörün objeye olan mesafesi aşağıdaki eşitlik kullanılarak hesaplanır (Joseph 2018).

$$Mesafe = \frac{c*t}{2} \quad (3.7)$$

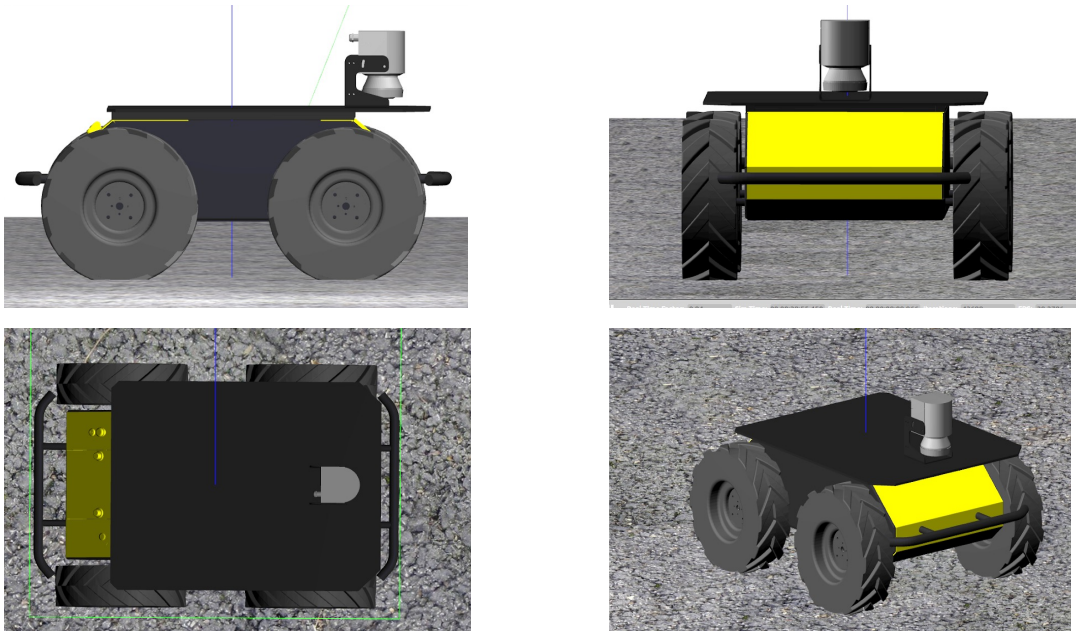
Bu eşitlikte;

c: Işık hızı,

t: Lazer ışınının gönderilmesi ve alınması arasında geçen süredir.

Şekil 3.3. LMS-111 LIDAR sensör (Anonim, 2018d)

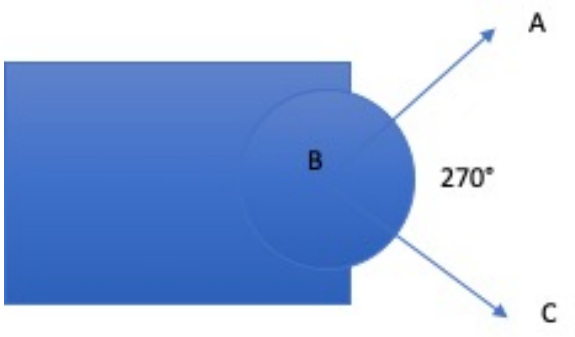
Simülasyonda kullanılan LİDAR tarafından yayınlanan lazer ışın demeti bir ayna yardımıyla deflekte edilerek 0,5° açısal çözünürlükle maksimum 270°' lik bir alanı tarayabilir ve maksimum 20 m mesafeden ölçüm alınabilir. LİDAR' ın maksimum tarama frekansı 50 Hz' dir ve 1 cm ölçüm çözünürlüğüne sahiptir. Kullanılan sensör modelinde gerçek dünyayı daha iyi yansıtabilmek adına normal dağılıma sahip (ortalama=0, standart sapma=0,03) hata simülasyonda sensörden okunan değerler üzerine rastgele olarak eklenmiştir.



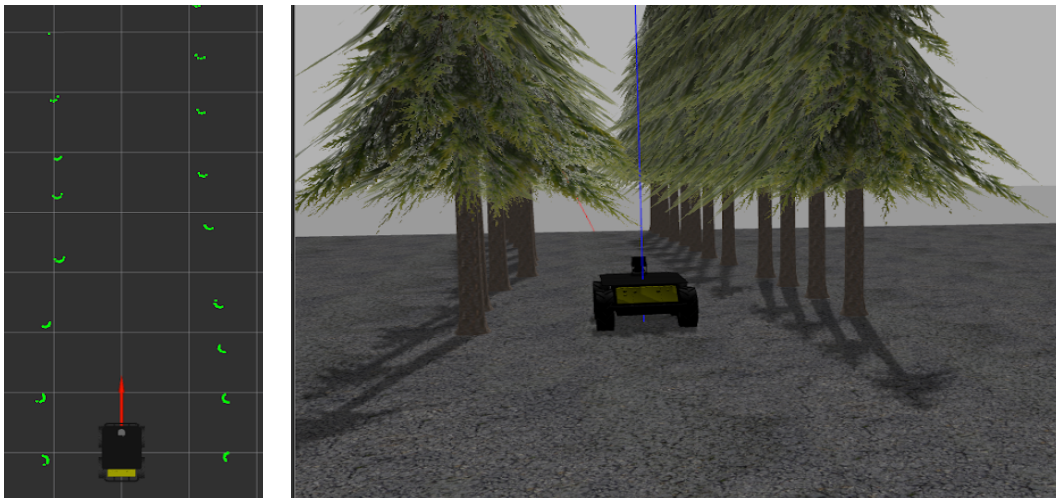
Şekil 3.4. Tasarlanan otonom araç 3 boyutlu gösterimi

Şekil 3.4' te otonom aracın üç görünüşü ve izometrik görünüşü bulunmaktadır. Sensörün ve otonom aracın boylamsal eksenleri hizalanmıştır. Geleneksel olarak tarama açısı -135' den (A noktası) başlar ve 135°' de (C noktası) biter (Şekil 3.5). LIDAR' dan herhangi bir t anında

571 noktadan ölçüm ($Z_t = (z_{(1)}, z_{(2)}, \ldots, z_{(541)})$ $\Theta = (\theta_{(1)}, \theta_{(2)}, \ldots, \theta_{(541)}) = (-135, -134{,}5, \ldots, 135)$ açılarına karşılık gelecek şekilde alınır. $z_{(j)}$, j ışını tarafından ölçülen nesneye (bitki yaprağı veya gövde) uzaklığı ifade etmektedir. Şekil 3.6' da otonom araç sıra arasındayken alınan bir tarama örneği görülmektedir. Veri noktaları (θ (j), z(j)) kartezyen koordinat üzerinde gösterilmiştir.



Şekil 3.5. LIDAR sensörün ölçüm açısı



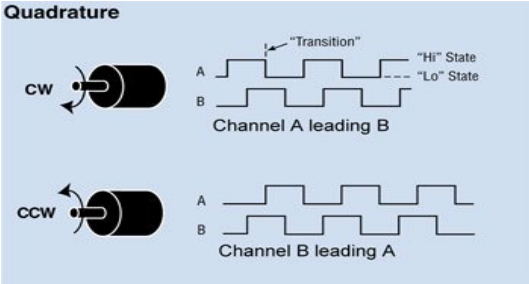
Şekil 3.6. LIDAR tarafından alınan verinin görselleştirilmesi

3.3.2. Odometri sensörleri

3.3.2.1. Devir ölçer

Simülasyonda otonom araç tekerleklerinde 78000 pals/dk çözünürlüklü optik dört evreli enkoderler kullanılmıştır. Dört evreli enkoderlerin içerisinde iki kanal (Kanal A ve Kanal B) içeren bir kod diski bulunmaktadır. Bu kanallar 90 adet faz dışı açıyı kodlamıştır. Dört evreli enkoder ile kanal A ve B kanalları arasındaki ilişkiye bakılarak dönüş yönü ve hızı belirlenebilir

(Şekil 3.7). Saat yönünde dönüşte Kanal A önden giderken, saatin ters yönünde dönüşte Kanal B önden gitmektedir. Dört evreli enkoderler dönüş yönü ve hızı haricinde 360° bir dönüş içerisinde şaftın spesifik lokasyonunu belirleyebilecek bir sinyal daha vermektedir. Dört evreli enkoderler çift yönlü pozisyon tespiti ve uzunluk ölçme uygulamalarında kullanılmaktadır. Kullanılan sensör modelinde gerçek dünyayı daha iyi yansıtabilmek adına normal dağılıma sahip (ortalama=0, standard sapma=0,2) hata simülasyonda sensörden okunan değerler üzerine rastgele olarak eklenmiştir.



Şekil 3.7. Dört evreli devir ölçer sensör (Anonim, 2018e)

3.3.2.2. Atalet sensörü

Simülasyonda otonom aracın üç boyutlu uzayda pozisyonunu algılamak için CHR-UM6 atalet sensörü (CH Robotik, Victoria, Avustralya) kullanılmıştır (Şekil 3.8). Atalet sensörü 3 eksenli ivme, 3 eksenli jiroskop ve barometrik yükseklik ölçebilmektedir. UM6 atalet sensörü 500 Hz çalışma frekansına sahiptir. Seri port ile haberleşme yapabilmekte ve jiroskopu ±2000 º/s, ivme ölçeri ise ± 2 g aralığında ölçüm yapabilmektedir. Kullanılan sensör modelinde gerçek dünyayı daha iyi yansıtabilmek adına normal dağılıma sahip (Ortalama=0, Standart Sapma=0,01) hata simülasyonda sensörden okunan değerler üzerine rastgele olarak eklenmiştir.

Şekil 3.8. Atalet sensörü (Anonim, 2018f)

3.4. Bilgisayar

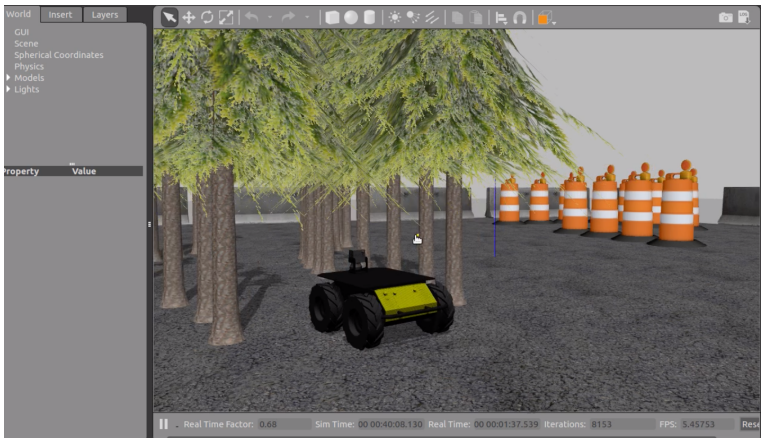
Geliştirilen programın çalıştırılması için Linux Ubuntu 16.04 LTS versiyon işletim sistemi kurulu olan 8 GB hafıza, 2,6 GHz hızında 4 çekirdekli işlemci, 2 GB hafızalı ekran kartına sahip bilgisayar kullanılmıştır.

3.5. Python Programlama Dili

Python genel amaçlı, yüksek seviyeli yorumlanabilir bir programlama dilidir. Guido van Rossum tarafından 1991 yılında geliştirilmiştir. Python nesne yönelimli programlama yapısı, hafıza kullanımını otomatik olarak yönetmesi, kolay okunabilir komut yapısı nedeniyle yaygın olarak kullanılmaktadır. Python yorumlayıcılar günümüzde kullanımda olan tüm işletim sistemleri ile çalışmaktadır. Web geliştirme, kullanıcı arayüzü programlama, gelişkin hafıza yönetimi, görüntü işleme, makina öğrenimi, derin öğrenme, büyük veri yönetimi gibi birçok fonksiyonu sağlayacak kütüphanelere sahiptir. Ayrıca kullanılan robotik yazılım geliştirme çerçeve programı tarafından desteklenen iki ana programlama dilinden birisi (Python ve C++) olması nedeniyle Python 3.0 programlama dili tercih edilmiştir. Geliştirilen program modülleri Python programlama dili ile yazılmıştır.

3.6. Simülasyon Programı

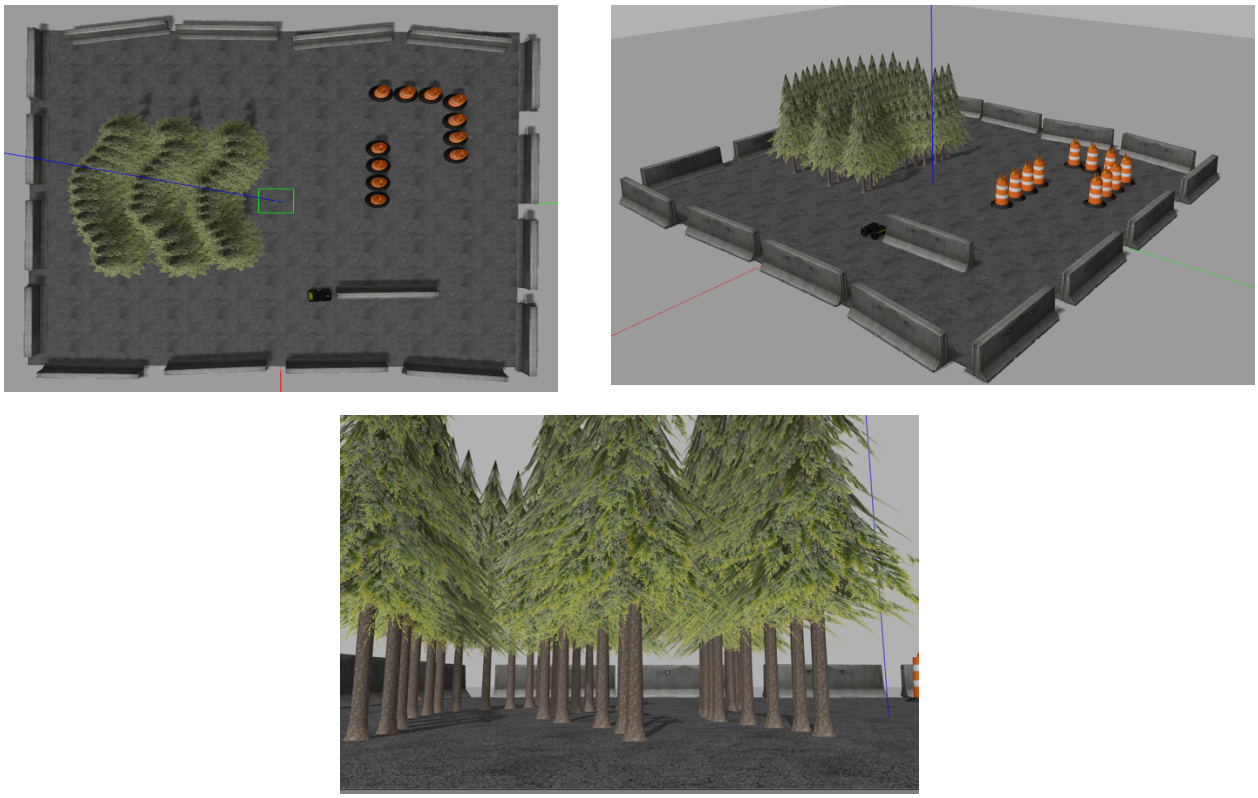
Robot geliştirmede simülasyon çok önemlidir. İyi tasarlanmış bir simulator ortamı ile hızlı test algoritmalarının geliştirilmesi, robot dizaynı, regresyon testi, realistik senaryolar içerisinde yapay zeka sistemlerinin test edilmesi gibi işlemler yapılabilir (Şekil 3.9) (Hsu ve Koenig 2012). Gelişkin fizik motoru, yüksek kaliteli grafikleri, programlamaya uygun grafik arayüzü, açık kaynak kodlu olması sebebiyle ücretsiz olması ve kullanılan robotik yazılım geliştirme çerçeve programıyla beraber çalışabilmesinden ötürü Gazebo 9.0.0 simülasyon programı kullanılmıştır.



Şekil 3.9. Gazebo simülasyon programı genel çalışma ekranı görüntüsü

3.6.1. Simülasyon ortamının ve otonom aracın oluşturulması

Tasarlanan otonom aracın engeller karşısındaki manevra kabiliyetini ölçmek amacıyla böyle bir çalışma ortamında karşılaşabileceği çeşitli engel varyasyonları Gazebo simülasyon ortamında hazırlanmıştır (Şekil 3.10).



Şekil 3.10. Simülasyonda kullanılan test alanı

Bu engeller birbirlerinden kavisli olarak ve sıra arası 1,5 m (±0,1 m) olacak şekilde ayarlanmış ağaç modelleri, sürüş sırasında dar bir kapıdan geçip daha geniş bir ağızdan çıkmayı temsil edecek koni modelleri ve büyük bir engellin etrafından dolaşmayı temsil edecek engel modeli kullanılmıştır. Ağaç modelleri özellikle kök taraflarında yaprak olmayan modeller olarak seçilmiş böylece sensörün algılayabileceği engel boyutu daraltılmış ve otonom sürüş ortamı daha zor hale getirilmiştir. Simülasyon ortamının etrafı engeller ile kapatılarak alan sınırlandırılmıştır. Tez çalışmasında tasarım kolaylığı ve dönüş yarıçapının daha kısa olabilmesi nedeniyle dört tekerlekli, sağ ve solunda iki elektrik motorundan tahrik alan bir otonom araç modeli kullanılmıştır.

Gazebo ortamında oluşturan otonom araç için devir ölçer sensör, UM6 atalet sensörü (devir ölçer ve atalet sensörü odometri verisini üretmektedir), LMS-111 LIDAR sensör kullanılmıştır. Bu ekipmanların seçilmesinin sebebi yaygın olarak kullanılmaları ve daha sonra geliştirilecek prototip üzerinde, geliştirilen bu yazılımın minimum değişiklikle uygulanmak istenmesidir.

Çizelge 3.1. Simülasyon ortamını tanımlayan world dosyası

```
<?xml version="1.0"?>
  <sdf version="1.4">
    <world name="default">
    <!-- Taban -->
    <include>
      <uri>model://ground_plane</uri>
    </include>
    <!-- Işık kaynağı -->
    <include>
      <uri>model://sun</uri>
    </include>
    <include>
      <uri>model://model_dosya_ismi</uri>
    </include>
    </world>
  </sdf>
```

Gazebo programında simülasyon ortamı world ve model dosyaları olmak üzere iki ana kısımdan oluşmaktadır. Bu her iki dosya da bir XML formatı olan Simülasyon Tanımlama Formatını (SDF) kullanmaktadır. Simülasyon içerisinde yer alan otonom araç, ışık kaynağı, sensör ve statik objeler oluşturulan world dosyasında bulunmaktadır. World dosyası içerisine eklenen modellerin konumları, boyutları, fiziksel özellikleri, hareket kabiliyetleri ayarlanmıştır. Gazebo sunucusu bu dosyayı okuyarak simülasyon ortamını oluşturmuştur. Çizelge 3.1' de taban, ışık kaynağı ve bir modelden oluşan örnek bir world dosya yapısı görülmektedir.

Çizelge 3.2. Simülasyona 3 boyutlu tasarımların eklenmesi

```
<visual name='visual'>
  <geometry>
    <mesh>
      <uri>model://robot_model/meshes/örnek.dae</uri>
    </mesh>
  </geometry>
</visual>
```

Simülasyon ortamına eklenecek otonom araç ve aracı oluşturan parçalar Google Sketchup Make programında 3 boyutlu olarak çizilmiş ve .dae uzantılı olarak kaydedilmiştir. Bu parçalar aşağıdaki örnek XML kodu formatında simülasyon ortamına eklenmiştir (Çizelge 3.2).

Otonom aracın hareketli parçaları, izin verilen hareket yönü, kütle ve parçanın hacmine göre Gazebo tarafından hesaplanan atalet özellikleri aşağıdaki örnek xml dosyası gibi tanımlanmıştır (Çizelge 3.3).

Son olarak Gazebo kütüphanesinde bulunan ve otonom araca eklenecek olan sensörler simülasyon ortamına eklenmiştir. Çizelge 3.4' te eklenen LIDAR sensöre ait XML kod örneği verilmiştir.

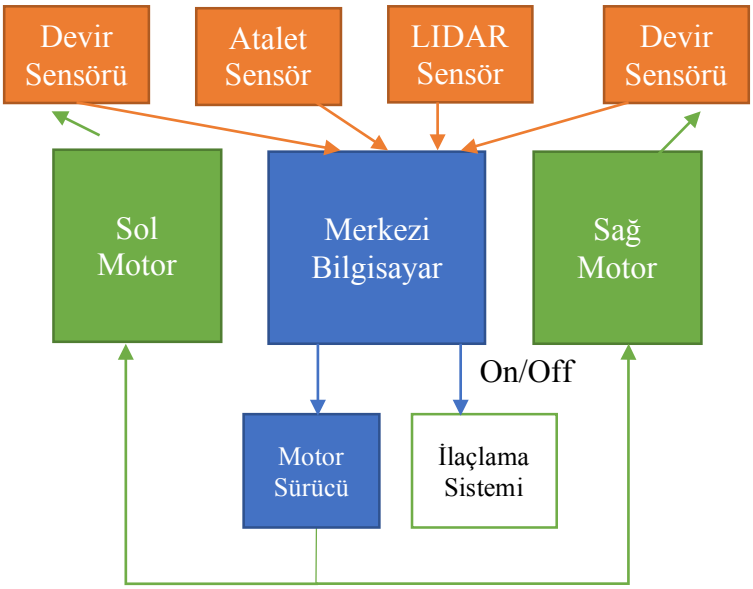
Çizelge 3.3. Otonom araç modeline eklenen parçaların atalet özelliklerinin tanımlanması

```
    <inertial>
      <mass>1.0</mass>
      <inertia>
         <ixx>0.083</ixx>
        <ixy>0.0</ixy>
        <ixz>0.0</ixz>
        <iyy>0.083</iyy>
        <iyz>0.0</iyz>
        <izz>0.083</izz>
      </inertia>
    </inertial>
    <collision name="collision">
      <geometry>
        <box>
          <size>1 1 1</size>
        </box>
      </geometry>
    </collision>
```

Çizelge 3.4. Simüle edilen otonom araca sensör eklenmesi

```
  <include>
    <uri>model://lms111</uri>
    <pose>0.2 0 0.2 0 0 0</pose>
  </include>
  <joint name="lms_joint" type="fixed">
    <child>lms111::link</child>
    <parent>chassis</parent>
  </joint>
```

Çizelge 3.4' teki örnek kodda eklenecek sensörün otonom araç üzerindeki konumu <pose>...</pose> etiketi içerisinde belirlenmiştir. <joint> etiketi içerisinde bu eklemin hareketsiz olduğu ve hangi parçaya bağlı olduğu belirtilmiştir.



Şekil 3.11. Donanım diyagramı

Sistemin temelini, geliştirilen otonom navigasyon programını çalıştıran bilgisayar oluşturmaktadır (Şekil 3.11). Bu program sensörlerden gelen verileri işleyerek haritalama, lokalizasyon, rota planlama ve ilaçlama sisteminin açılıp kapatılması işlerini yapmıştır. LIDAR sensörü çevresinden aldığı mesafe ölçüm verilerini navigasyon programına iletmiştir. Devir ölçer sensör ve atalet sensöründen alınan veriler navigasyon programına gönderilerek otonom aracın hareket hızı, yönü ve gittiği mesafenin hesaplandığı odometri mesajı elde edilmiştir. Elektrik motoru rota planlama uygulaması tarafından hesaplanan rotayı takip edecek şekilde navigasyon programından gelen doğrusal hız ve dönüş komutlarını motor sürücü aracılığı icra etmiştir.

3.7. Otonom Sürüş Geliştirme Programı

Çalışmada otonom sürüş geliştirme için, çerçeve programı olarak ROS Kinetic sürümü kullanılmıştır. ROS, robotların sıfırdan dizayn edilmelerini ve çeşitli işlemleri gerçekleştirebilmeleri için programlanabilmelerini sağlayan açık kaynak kodlu bir çerçeve programdır. Robot tasarımında geniş bir kullanıcı sınıfı tarafından ortak olarak kullanılabilecek

bir platform ihtiyacı nedeniyle ortaya çıkmıştır. Bu ortak platform sayesinde tasarımcılar kolayca prototip geliştirme imkânı bulmaktadırlar (Quigley 2012).

Çerçeve program sayesinde sık kullanılan aygıt sürücüleri otomatik olarak tanınmakta böylece donanımın sisteme tanıtılmasından çok robotik tasarım sürecine zaman ayrılabilmektedir. ROS'un sunduğu donanım ayırma katmanı sayesinde geliştiriciler arka planda kullanılan donanıma bağlı olmadan geliştirme yapabilirler. Robotik tasarım sürecinde haritalama, hareket planlama, sensör verilerinin yorumlanması, objelerin manipüle edilmesiyle ilgili ençok kullanılan algoritmalar ROS ortamında kolaylıkla uygulanabilmekte ve ihtiyaca uygun şekilde değiştirilebilmektedir. İş yükünün paralel bilgisayarlara bölünüyor olabilmesi tasarlanan sistemlerin kolaylıkla ölçeklenebilir olmasını sağlamaktadır. Bunun yanı sıra gelişmiş görselleştirme araçları, simülasyon ortamı, hata ayıklama sistemi ve sensör verilerinin kolaylıkla kaydedilebilmesi tasarlanan algoritmaların kolaylıkla test edilebilmesini ve uygulamaya hazır hale getirilmesini sağlamaktadır (Hsu ve Koenig 2012).

Modüler bir tasarım yapısına sahip ROS sayesinde bir robot için tasarlanan farklı kısımlar herhangi bir değişiklik yapılmadan veya küçük değişiklikler ile başka tasarımlar için de rahatlıkla kullanılabilir. Bu özelliği sayesinde prototip oluşturma süresi oldukça azalmaktadır (Bubeck 2013).

ROS'un akademi ve endüstride yaygın olarak kullanan aktif bir geliştirici ekosistemi vardır. Bu aktif geliştirici ekosistemi ROS'u diğer robotik çerçeve programlarından pozitif yönde ayırır. ROS projesi ilk defa 2007 yılında Stanford Üniversitesinde Switchyard ismi altında başlamıştır. Geliştirme çalışmaları 2008 yılında Willow Garage start-up firması tarafından devam ettirilmiş 2013 yılından beri ise Willow Garage tarafından kurulan Açık Kaynaklı Robotik Vakfı (OSRF) tarafından devam ettirilmektedir (Quigley 2012). ROS Ubuntu Linux işletim sistemiyle tam uyumlu olarak çalışmaktadır. Bunun yanı sıra Ubuntu ARM, Debian, Gentoo, Mac OS X, Arch Linux, Android, Windows ve Open Embedded işletim sistemleri için deneysel sürümleri de bulunmaktadır (Quigley ve ark. 2016).

ROS aşağıdaki nedenlerden dolayı çalışmada tercih edilmiştir (Joseph 2018).

Ortak Geliştirme: ROS açık kaynaklı bir yazılım çerçevesidir. Endüstri ve araştırmalarda kullanımı ücretsizdir. Geliştiriciler yeni paketler ekleyerek ROS'un

fonksiyonelliğini arttırabilir. ROS' taki tüm paketlerin donanım ayırma katmanı bulunduğu için mevcut bir paket rahatlıkla farklı robot konfigürasyonları için kullanılabilir.

Programlama Dili Desteği: ROS haberleşme çerçevesi modern programlama dillerinden birçoğu ile haberleşebilir. Mevcut durumda C++, Python ve List desteği bulunmakta Java ve Lua programlama dilleri için ise deneme sürümleri bulunmaktadır.

Kütüphane Entegrasyonu: ROS, Açık Kaynaklı Görüntü İşleme Kütüphanesi (OpenCV), Nokta Bulutu Kütüphanesi (PCL) gibi birçok üçüncü parti robotik yazılım kütüphaneleri ile arayüz oluşturabilmektedir. Geliştiriciler bu üçüncü parti yazılım kütüphaneleri ile kolay bir şekilde çalışabilmektedir.

Entegre Simülatör: ROS açık kaynaklı bir robot simülatörü olan Gazebo ile entegre çalışabilmektedir. Ayrıca Webots ve V-REP gibi diğer yaygın kullanılan robot simülatörleri ile arayüz oluşturabilmektedir.

Yazılım Testi: ROS içerisinde, yazılıma ait kodların belirlenen test yöntemleri açısından kalitesini test edip hataları yakalayabilen **rostest** aracı bulunmaktadır.

Genişleyebilirlik: ROS genişleyebilir bir mimaride geliştirilmiştir. Ağır hesaplamalar robot üzerinde yapılabileceği gibi bulut veya heterojen bilgisayar kümeleri üzerinde de yapılabilir.

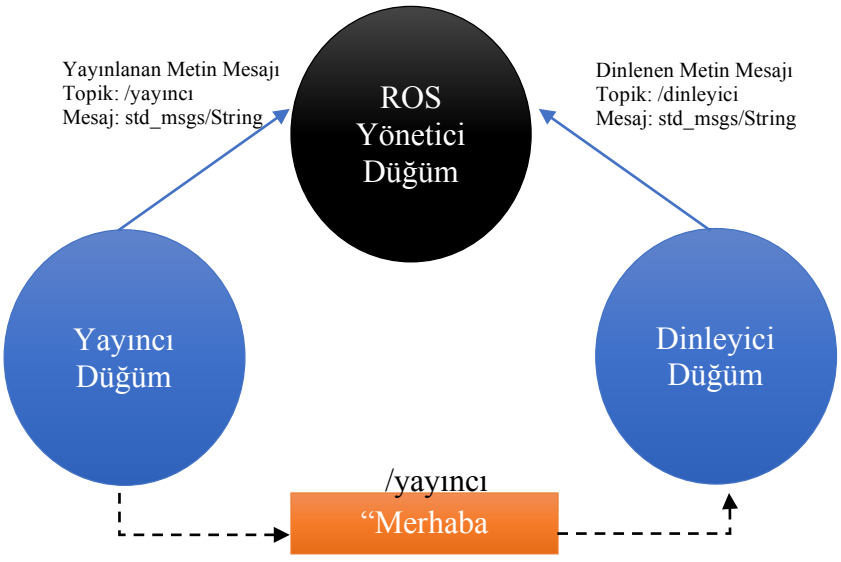
Kişiselleştirilebilirlik: ROS tamamen açık kaynak kodlu ve kullanımı ücretsiz olduğu için yapılan tasarıma özgü olarak özelleştirilebilir. Örneğin ROS' un sadece mesajlaşma platformunu kullanmak istendiğinde sadece bu platformu kullanıp diğer bütün özellikler kaldırılabilir.

Yaygınlık: ROS, OSRF tarafından desteklenen geniş bir kullanıcı topluluğuna sahiptir. Bu da güncel kalmasını, robotik çalışmalarda yaygın olarak kullanılan robot platformları ve sensörleri desteklemesini sağlamaktadır.

3.7.1. Geliştirilen otonom sürüş yazılımın temel yapısı

Otonom sürüş uygulaması Python programlama dili kullanılarak ROS dosya yapısına uygun şekilde modüller halinde programlanmıştır. Modüllerin birbirleriyle haberleşmesi için ROS tarafından sağlanan mesaj formatları kullanılmış böylece standardizasyon sağlanmıştır. Geliştirilen uygulamada odometri verileri için 3 eksendeki doğrusal ve açısal hareketleri ifade etmek için 6 eksenli bir odometri mesajı, LIDAR sensörden alınan ölçümleri uygulamada kullanmak için lazer tarama mesajı, oluşturulan haritadaki engellerin hücresel konumlarını ve değerlerini ifade eden matris formatındaki mesaj, sensörlerin aldığı ölçümleri otonom araç üzerindeki konumlarına göre otomatik olarak güncelleyen geometrik dönüşüm mesajı, lokalizasyon uygulaması tarafından tahmin edilen konum mesajı, rota boyunca izlenecek noktaları içeren global, lokal rota mesajları ve ilaçlama kontrol mesajı oluşturulmuştur.

Simülasyon ve navigasyon olmak üzere iki klasör içerisinde düzenlenmiş simülasyon ortamını tanımlayan tasarım ve program dosyaları simülasyon klasöründe, navigasyon ve ilaçlama için kullanılanacak program dosyaları navigasyon paketi içerisinde organize edilmiştir.



Şekil 3.12. ROS sisteminde düğümler arası haberleşme

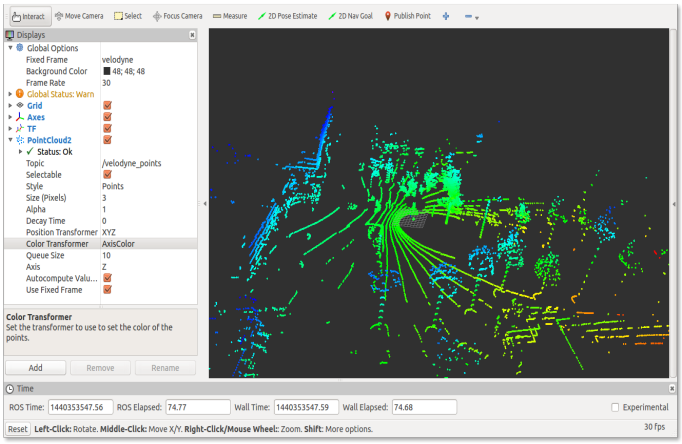
Odometri sensör verilerinin okunması, LIDAR sensör verilerinin okunması, harita oluşturma, lokalizasyon, rota planlama ve ilaçlama kontrolü işlemleri ROS program yapısına uygun olarak düğümler şeklinde modüler olarak programlanmıştır. ROS düğümleri yazılan program arayüzlerinin birbiriyle haberleşme birimleridir. Oluşturulan bu düğümlerin birbirleri ile bağlantısını sağlamak amacıyla ROS yönetici düğüm kullanılmıştır. ROS yönetici düğüm bir düğümden gelen veri paylaşım isteğini hedef düğümle paylaşır böylece bu iki düğüm haberleşmeye başlar. Şekil 3.12 'de düğümler arası haberleşme yapısını gösteren örnek diyagram görülmektedir.

Geliştirilen uygulamada LIDAR ve odometri sensörlerinden alınan ölçümler haritalama düğümü tarafından dinlenip harita oluşturmak için işlenmiştir. Lokalizasyon düğümü hem bu sensör verilerini dinlemiş hemde harita düğümünden yayınlanan haritayla karşılaştırarak otonom aracın harita üzerindeki konumunu bulmuştur. Rota planlama düğümü ise otonom aracın bulunduğu konum ve gitmek istediği konum arasında harita üzerindeki engelleri dikkate alarak rota hesaplamıştır. Hesaplanan rota mesajı otonom aracın motor kontrol ünitesine hareket mesajı olarak gönderilmiştir. İlaçlama düğümü ise lokalizasyondan aldığı konum bilgisini işleyerek ilaçlama yapılması istenen sınır koşullarında ilaçlamayı başlatacak, ilaçlama sınır bölgesinden çıktığı durumlarda ise ilaçlamayı durduracak açma kapama komutunu vermiştir.

3.7.2. ROS geliştirme araçları

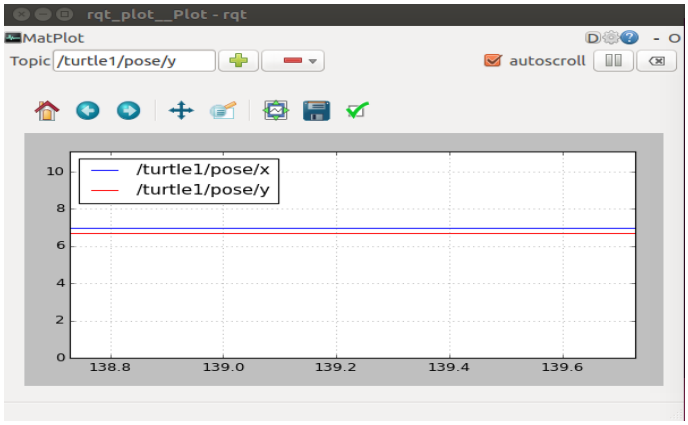
ROS ortamında geliştirilen yazılımı incelemek ve hata ayıklamak için çeşitli kullanıcı arayüzleri ve komut satırı araçları bulunmaktadır. Otonom araç geliştirmede tercih edilmesinin önemli sebeplerinden biri de sağladığı bu araçlardır. Görselleştirme ve hata ayıklama araçları sayesinde elde edilen ölçümler görselleştirilerek yazılımın test edilmesi sağlanmıştır.

ROS görselleştirme aracı (rviz): Rviz (Anonim 2018a) geliştirilen programa ait topik ve bunlara ait parametrelerin 2 veya 3 boyutlu olarak görselleştirilmesi için kullanılan bir grafik aracıdır. Otonom araç modelinin, LIDAR sensör verilerinin, ortam haritasının, engel katmanlarının, lokalizasyondaki parçacık dağılımının, odometri sensör verilerinin ve rota planının görselleştirilmesi için kullanılmıştır (Şekil 3.13).



Şekil 3.13. Görselleştirme Aracı Arayüzü

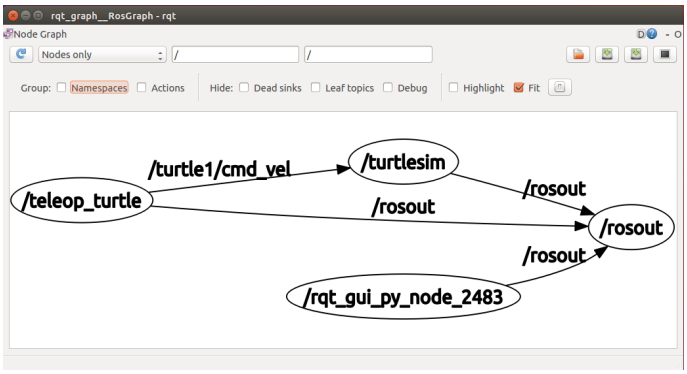
ROS grafik aracı (rqt_plot) : rqt_plot (Anonim 2018b) bir komut satırı aracıdır. Skaler formdaki ROS topiklerine ait grafiklerin görselleştirilmesi için kullanılan bir araçtır (Şekil 3.14.)



Şekil 3.14. rqt_plot örnek gösterimi

rqt_plot aracıyla gerçek konum ve tahmin edilen konumun görselleştirilmesi yapılarak lokalizasyon ve rota planlama uygulamalarının başarıları incelenmiştir.

ROS program görselleştirme aracı (rqt_graph): rqt_graph (Anonim 2018c) ROS düğümlerinin birbiriyle olan bağlantılarını incelemek için kullanılan bir kullanıcı arayüzüdür. Bu sayede düğümlerinin birbirleri ile olan ilişkileri görselleştirilmiştir (Şekil 3.15).



Şekil 3.15. rqt_graph örnek gösterimi

rqt_graph aracı ile geliştirilen haritalama ve lokalizasyon uygulamalarının diyagramları oluşturulmuştur. Bu diyagram sayesinde geliştirilen düğümler arasındaki ilişki ve birbirleriyle haberleşmek için kullandıkları mesaj türleri görülebilmektedir. Oluşturulan navigasyon uygulamasına ait program diyagramı rqt_graph aracı kullanılarak elde edilmiştir.

3.8. Parçacık Filtresi

Parçacık filtresi algoritması haritalama uygulamasında geliştirilen SLAM ve lokalizasyon uygulamasında kullanılan Adaptif Monte Karlo Lokalizasyon algoritmalarının temelini oluşturmuştur.

Otonom araç navigasyonu dinamik bir sistemdir. Otonom konumu (X_t= (x_t, yt, θ_t)^T) herbir zaman adımında değişir. Sensörlerin ve aktüatörlerin kusursuz olmaması bunun yanısıra çevreden kaynaklanan çeşitli elektriksel gürültüler nedeniyle hesaplanan otonom araç konumunun belirsizliği artar. Otonom konumunda meydana gelen bu belirsizlikle mücadele edebilmek için herhangi bir zamandaki pozisyona ait olasılıksal dağılım eşitlik 3.8' deki gibi ifade edilmiştir (Thrun 2005).

$$p(X_t \mid Z_{1:t}, U_{1:t}) \qquad (3.8)$$

Burada, $Z_{1:t}$ otonom araç tarafından t zamanına kadar yapılan ölçümleri, $U_{1:t}$ ise otonom araç tarafından t zamanına kadar yapılan kontrolleri (bu kontroller odometri sensörleri tarafından dolaylı olarak ölçülmektedir) ifade etmektedir. Bu olasılıksal dağılım (sonsal dağılım) her bir zaman adımı için değerlendirilmiştir. Sonsal dağılımın değerlendirilmesi

parçacık filtre algoritmasıyla yapılmıştır. Parçacık filtresinin temel amacı olası pozisyonları ifade eden sonsal dağılımın parçacık adı verilen rastgele örnekler ile temsil edilmesidir.

Bu parçacıklar tekrarlı olarak her yeni Z_t ölçümü için yenilenmiştir. Algoritma tahmin ve güncelleme olmak üzere iki adım içermektedir. Tahmin adımında parçacıkların yeni değerleri; anlık değerler ve otonom aracın hareket modeline bağlı olarak hesaplanmıştır. Güncelleme adımında ise tahmin edilen değerler; Z_t ölçümleri ve atanan ağırlık değerleriyle uyumluluk bakımından incelenmiş ve ardından parçacıklar normalize ağırlıklarına göre yeniden örneklenerek sonsal dağılımı aşağıdaki eşitlikteki gibi (Thrun 2005) oluşturmuştur.

$$p(X_{1:t} \,|Z_{1:t}, U_{1:t}) = \frac{p(Z_t|X_t)p(X_t|X_{t-1}U_t)}{p(Z_t|Z_{1:t-1})} p(X_{1:t-1}|Z_{1:t-1}, U_{1:t-1}) \qquad (3.9)$$

Yukarıdaki eşitlikte:

$p(Z_t \mid X_t)$: Ölçüm modeli tarafından verilen güncelleme adımını,
$p(X_t \mid X_{t-1}, U_t)$: Hareket modeli tarafından verilen tahmin adımını,
$p(Z_t \mid Z_{1:t-1})$ Normalleştirme katsayısını
$p(X_{1:t-1} \mid Z_{1:t-1}, U_{1:t-1})$: t-1 anındaki sonsal olasılık dağılımı göstermektedir.

Çizelge 3.5. Parçacık filtresi algoritması (Thrun 2005)

1: **Parçacık_Filtresi_Algoritması** (X_{t-1}, Z_t, U_t):
2: $\overline{X_t} = X_t = Ø$
3: **for** n=1 **to** N
4: $x_t^n \sim p(x_t|x_{t-1}^n, u_t)$
5: $w_t^n = p(z_t|x_t^n)$
6: $\overline{X_t} = \overline{X_t} + \langle x_t^n, w_t^n \rangle$
7: **endfor**
8: **for** n=1 **to** N
9: draw n with high probability w_t^n
10: $X_t = X_t + x_t^n$
11: **endfor**
12: **return** X_t

Parçacık filtresi uygulaması geliştirilen yazılımda Çizelge 3.5' te gösterildiği şekliyle kullanılmıştır. Parçacık filtresi parametrik olmayan bir bayes filtresi uygulamasıdır. Sonlu sayıdaki parametreleri (belirli aralıklarla alınan sensör ölçümleri ve uygulanan kontroller) kullanarak konuma ait sonsal dağılımı tahmin etmekte kullanılmıştır. Parçacık filtresindeki sonsal dağılım parçacıklar olarak ifade edilir ve aşağıdaki şekilde gösterilir:

$$X_t = x_t^1, x_t^2, \dots, x_t^N \tag{3.10}$$

Herbir parçacık t zamanında gerçek durumu ifade eden sistem durumunun bir hipotezidir. Burada N, X_t parçacık setindeki toplam parçacık sayısını ifade etmektedir. Çizelge 3.5' te parçacık filtresine ait algoritma verilmiştir. Bütün Bayes filtrelerinde olduğu gibi X_t rekürsif olarak bir önceki zaman adımına ait set olan X_{t-1}, den elde edilmiştir. Algoritmanın girdileri sistemin t-1 zamanındaki durumunu gösteren X_{t-1}, en son uygulanan kontrol u_t ve en son yapılan ölçüm z_t' dir. Öncelikle X_{t-1} 'deki sistem durumuna göre geçici bir $\overline{X_t}$ seti oluşturmuştur. Bu işlem X_{t-1} setini oluşturan x_{t-1}^n parçacıklarının herbirinin işlenmesiyle elde edilmiştir.

Algoritmanın 4. satırında t zamanı için x_t^n hipotez durum, x_{t-1}^n ve u_t kullanılarak oluşturulmuştur. Sonuç n indeksiyle indekslenerek bu parçacığın X_{t-1} 'in n' inci parçacığı kullanılarak oluşturulduğu belirtilmiştir. Algoritmanın 5. Satırında her bir parçacık için w_t^n olarak gösterilen bir önem katsayısı hesaplanmıştır. Bu katsayı yapılan ölçümlerin parçacık setine aktarılabilmesi için kullanılmıştır. Bu nedenle önem katsayısı parçacık x_t^n için z_t ölçüm değerinin alınma olasılığıdır. Algoritma satırları 8-11 arasında ise yeniden örnekleme ve önem katsayısı hesaplaması yapılmıştır. Önem katsayısına göre yüksek olasılıklı parçacık X_t 'ye eklenmiştir.

3.9. Ortam Haritasının Oluşturulması

Bir aracın otonom olarak hareket edebilmesi için öncelikle nasıl bir ortamda bulunduğunu bilmesi gerekir. Bunun için bir haritaya sahip olması ve bu haritanın üzerinde nerede olduğunun otonom araç tarafından algılanabilmesi otonom navigasyon için zorunludur.

Eğer otonom araç tasarımında kullanılan sensörler kusursuz olsaydı ve verilen hareket komutları kusursuz olarak uygulanabilseydi otonom navigasyon için kullanılacak harita oluşturma işlemi ve otonom araç tasarımı oldukça kolay olurdu. Sensörlerden alınan veriler otonom aracın pozisyonu ve geometrisi kullanılarak bir global koordinat çerçevesine dönüştürülmekte, haritaya kaydedilmekte böylece kolay bir şekilde ortama ait harita oluşturulmaktadır. Ancak sensör ve aktüatörler kusursuz olmadıkları ve çevre şartlarına bağımlı olarak değişen düzeyde hatalı ölçüm ve kontrol yaptıkları için otonom navigasyonda kullanılacak harita oluşturma işlemi zor bir adımdır. Otonom araç değişken dış ortam şartları nedeniyle nasıl hareket ettiğini kesin bir doğrulukla algılayamamaktadır.

3.9.1 Eşzamanlı lokalizasyon ve haritalama (simultaneous localization and mapping-SLAM)

Haritalama için ortamı hücreler halinde ifade eden ve otonom araç konumunu parçacık filtresi yöntemini kullanarak hesaplayan SLAM algoritması kullanılmıştır (Grisetti ve ark., 2007). Bu yöntemin temel fikri (Murphy 1999); harita (M) ve otonom aracın yörüngesi

$$X_{1:t} = X_{1,}\ldots,X_t \tag{3.11}$$

arasındaki bileşik sonsal dağılımın

$$p(X_{1:t}, M \mid Z_{1:t}, U_{1:t-1}) \tag{3.12}$$

yukarıdaki eşitlik ile (Thrun 2005) tahmin edilmesidir. Bu tahmin otonom araç tarafından alınan lazer sensör ölçümleri

$$Z_{1:t} = Z_1,\ldots,Z_t \tag{3.13}$$

ve uygulanan kontroller

$$U_{1:t-1} = U_1,..,U_{t-1} \tag{3.14}$$

yardımıyla yapılmıştır. Kullanılan parçacık filtresi temelli haritalama uygulaması aşağıdaki faktörizasyonu (Thrun 2005) gerçekleştirmiştir.

$$p(X_{1:t}, M|Z_{1:t}, U_{1:t-1}) = p(M|X_{1:t}, Z_{1:t})p(X_{1:t}|Z_{1:t}, U_{1:t-1}) \quad (3.15)$$

Bu faktörizasyon ile öncelikle otonom aracın yörüngesi daha sonra ise bu yörüngeye uygun harita hesaplanmıştır. Harita otonom aracın pozisyon tahminine yüksek derecede bağımlıdır.

Haritaya ait sonsal dağılım

$$p(M|X_{1:t}, Z_{1:t}),\ X_{1:t} \text{ ve } Z_{1:t} \quad (3.16)$$

olarak bilindiği için, bilinen pozisyonlarla haritalama kullanılarak (Moravec 1989) analitik olarak hesaplanabilir. Potansiyel yörüngeleri bulmak için $p(X_{1:t}, M|Z_{1:t}, U_{1:t-1})$'in sonsal dağılımı parçacık filtresi yöntemi kullanılmıştır. Her bir parçacık otonom aracın potansiyel yörüngesini ifade etmekte ve her potansiyel yörüngeye ise bir harita karşılık gelmektedir. Harita, ilgili parçacığın sağladığı yörünge ve ölçümler ile oluşturulmuştur. Ortamdan alınan ölçüm sayısı arttığında oluşturulan haritanın doğruluğu artmıştır.

Çizelge 3.6' da geliştirilen haritalama uygulamasına ait algoritma gösterilmiştir. Algoritmada kullanılan lazer sensörün özelliklerine bağlı olarak seçilen T_{mesafe}(2 cm), $T_{açı}$ (0.5°), $T_{yeniden\_örnekleme}$ eşik değerleri bulunmaktadır.

Çizelge 3.6. SLAM algoritması (Thrun 2005)

1: SLAM Algoritması (T_{mesafe}, $T_{açı}$, $T_{yeniden\_örnekleme}$, N, Harita Boyutu, Z_t, U_t)
2: **while True:**
3: $\langle \boldsymbol{U_{x,t}}, \boldsymbol{U_{y,t}}, \boldsymbol{U_{\theta,t}} \rangle \leftarrow \boldsymbol{U_t}$
4: **if** $\sqrt{U_{x,t}^2 + U_{y,t}^2} > T_{mesafe} || u_{\theta,t} > T_{açı})$ **then**
5: $S_t = \{\ \}$
6: **for** all $s_{t-1}^i \in S_{t-1}$ **do:**
7: $\langle x_{t-1}^i, w_{t-1}^i, M_{t-1}^i \leftarrow s_{t-1}^i \rangle$
8: $x_t'^i = x_{t-1}^i \oplus u_t$
9: // **Tarama eşleştirme**

10: $\widehat{x_t^i} = argmax_x p(x \mid M_{t-1}^i, z_t, x_t'^i)$

11: **if** $\widehat{x_t^i} = failure$ **then**

12: $x_t^i \sim p(x_t | x_{t-1}^i, u_t)$

13: $w_t^i \sim w_{t-1}^i p(z_t | M_{t-1}^i, x_t)$

14: **else**

15: $x_t^i \sim p(x_t | M_{t-1}^i, x_{t-1}^i, z_t, u_t)$

16: $w_t^i \sim w_{t-1}^i p(z_t | M_{t-1}^i, x_t, u_t)$

17: **end if**

18: // **Harita güncelleme**

19: $M_t^i = \int M_{t-1}^i, x_t^i, z_t$

20: // **Örneklem setini güncelleme**

21: $s_t^i \leftarrow \langle x_t^i, w_t^i, M_t^i \rangle$

22: $S_t = S_t \cup s_t^i$

23: **end for**

24: // **Gerekliyse yeniden örnekleme**

25: $N_{eff} = \frac{1}{\sum_{i=1}^{N} (\widetilde{w_t^i})^2}$

26: **if** $N_{eff} < T_{yeniden\ örnekleme}$ **then**

27: $S_t = yeniden\_örnekle(S_t)$

28: **end if**

26: $t = t + 1$

27: **end if**

28: **end**

Kullanılan SLAM algoritmasında her bir parçacık bir küme şeklinde tanımlanmıştır.

$$s = (X, W, M) \tag{3.17}$$

Burada;

X: otonom aracın konumunu ve yönünü ifade eden pozisyon vektörü (x, y, θ),

W: skaler değere sahip parçacık önem katsayısını,

M: harita matrisini,

s_t^i : i indeksine ve t zamanına ait parçacığı göstermek için kullanılmıştır.

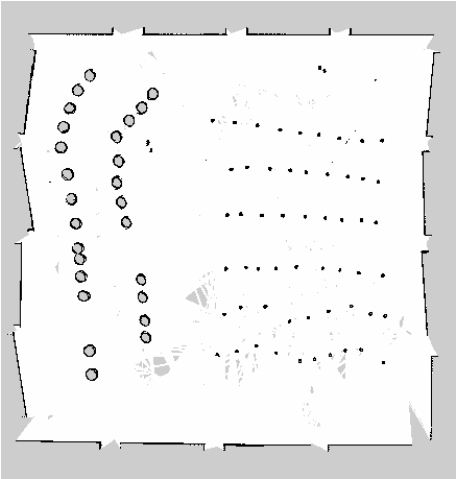
Algoritmada pozisyonu ifade eden X, (x, y, θ) koordinatlarından oluşur. Harita ise $M \in \mathbb{R}^{mxn}$ boyutlarında bir matristen oluşmaktadır. Matrisin herbir hücresi beyaz (0 değeri alır) ise engel yok, siyah ise engel var (255 değerini alır) ve gri ise bilinmiyor (-1 değerini alır) olarak kodlanmıştır. Lazer mesafe ölçüm sensör değerlerini içeren Z ise bir vektördür. Bu vektörün her bir indisi spesifik bir açıya karşılık gelen ölçüm değerini vermektedir. Odometri ölçümlerini gösteren u ise (x, y, θ) koordinatlarındaki değişimlere karşılık gelen doğrusal hız v_t ve açısal hız μ_t' ye karşılık gelmektedir. Pozisyon birleştirme operatörü $\oplus$ odometri ölçümleri ile pozisyon verilerini birleştirmek için kullanılmıştır. Parçacık fakirleşmesini önlemek amacıyla yeniden örnekleme ancak N_{eff}, $T_{\text{yeniden örnekleme}}$ eşik değerinin altına düştüğünde yapılmıştır. Bu eşik değeri başta kullanılan parçacık sayısının yarısı olarak belirlenmiştir (Lu ve Milios 1997). Herbir parçacık otonom aracın olası pozisyonu ifade eden bir hipotezdir.

3.9.2. Geliştirilen haritalama uygulaması

ROS ortamında geliştirilen haritalama uygulamasında JPEG veya PNG formatında bir harita ve harita ile ilgili çözünürlük, büyüklük ve eşik değerlerinin tanımlandığı bir öznitelik dosyası oluşturulmuştur (Çizelge 3.7). Oluşturulan harita, engelleri ifade eden pikseller şeklinde gösterilmiş ve her bir piksel üzerinde engel olma olasılığını gösteren bir değer almıştır.

Şekil 3.16' da doğrudan otonom aracın simülasyon ortamında LİDAR ve odometri sensör verilerinden öğrenilen bir harita gösterilmiştir. Beyaz bölgeler engel içermeyen, siyah bölgeler engel içeren ve gri bölgeler ise belirsiz alanları ifade etmektedir.

Çizelge 3.7' de geliştirilen haritalama uygulamasında kullanılan map.yaml dosyasında görüldüğü gibi harita dosyası map.png ismiyle ve png formatında kaydedilmiştir. Her bir piksel 2 cm' lik bir uzunluğu ifade etmektedir ve orijin noktası (0, 0, 0)' dır. Eğer pikselin parçacık filtresi tarafından hesaplanan engel içerme olasılığı %65'in üzerindeyse dolu, pikselin engel içerme olasılığı %19.6' nın altında ise boş olarak kabul edilmiştir. Bu eşik değerleri dış ortam haritalamasında uygundur (Joseph 2018).



Şekil 3.16. ROS ortamında sensör ölçümlerinden üretilen harita

Çizelge 3.7. Harita öznitelik dosyası

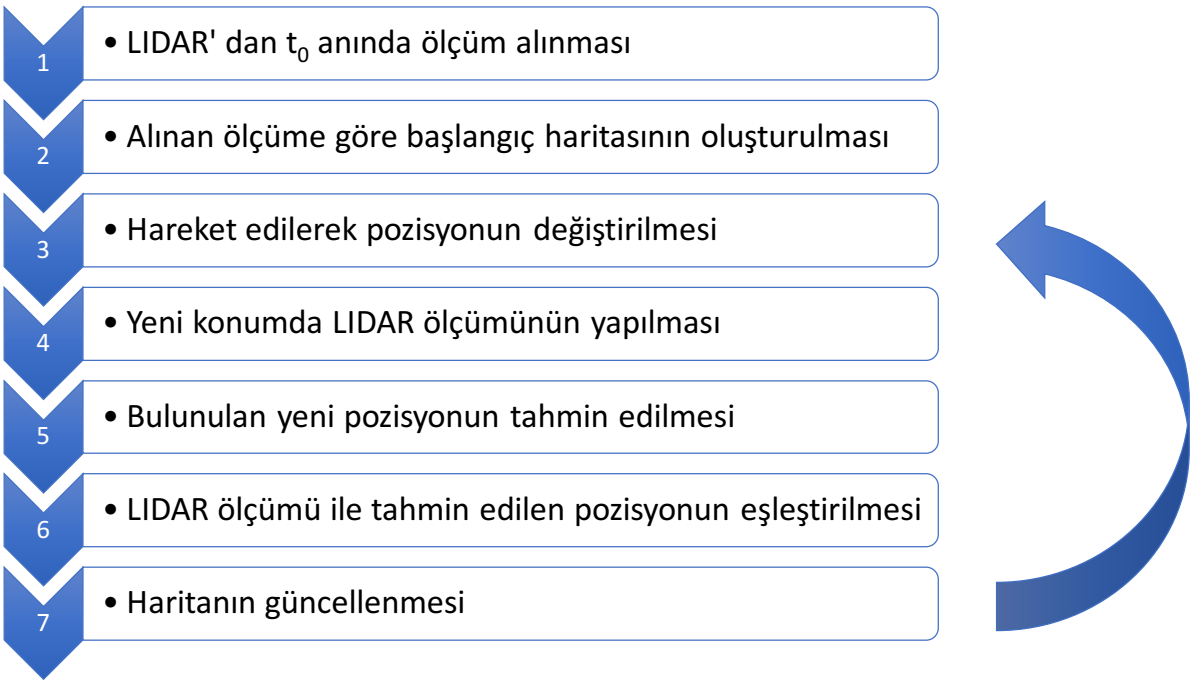
map.yaml
Image: map.png
Resolution: 0.02
Origin: [0.0, 0.0, 0.0]
Occupied_thresh: 0.65
Free_tresh: 0.196
Negate: 1

Harita çözünürlüğü haritayı oluşturan herbir pikselin metrik olarak ifade ettiği boyuta karşılık gelmektedir. Harita çözünürlüğü işlemsel yükü ve rota planlamayı etkilemektedir. Düşük çözünürlük (>= 5 cm) değerlerinde dar geçiş noktalarında engeller birbiri üzerine çakıştığından dolayı rota planlamak mümkün olmaz (Joseph 2018). Bu çözünürlüğün belirlenmesinde kullanılan LIDAR sensörün çözünürlüğü birinci derecede belirleyicidir. Eğer kullanılan LIDAR sensör çözünürlüğü seçilen harita çözünürlüğünden daha düşük bir değere sahipse LIDAR sensör ölçümleri tüm alanı kapsamadığı için harita üzerinde birçok bilinmeyen nokta bulunacaktır. Çalışmada kullanılan LIDAR sensörün çözünürlüğüne uygun olarak her bir hücre 2 cm uzunluğa karşılık gelmektedir.

Oluşturulan haritalar görüntü formatında saklandığı için herhangi bir görüntü editörüyle (örneğin Microsoft Paint) düzenlenebilir. Bu sayede sensör verileri tarafından oluşturulan haritaların istenilen kısımları manuel olarak düzeltilebilir. Sadece düzeltme işlemi değil aynı zamanda otonom aracın harita üzerinde gitmesi istenmeyen bölgeler var ise bu bölgelerin engel ifade eden siyah çizgiler ile bloke edilerek rota planlamasında gözardı edilmesi sağlanabilir.

Bazı senaryolarda otonom aracın araziye çıkarak tekrarlı bir şekilde ölçüm alması mümkün olmayabilir. Bu durumda ROS yayınlanan sensör mesajlarını kaydebilecek rosbag aracını sunmaktadır. Bu araç ile sensör verileri kaydedilerek daha sonra oynatılabilir. Böylece algoritmalarda yapılacak değişiklikler otonom aracı tekrardan çalıştırmadan kayıtlı sensör verileri üzerinden test edilebilirler. Geliştirilen uygulamada bilgisayar işlem gücü açısından bir yetersizlik olmadığı için kaydedilen veriler üzerinden değil sürüşle eş zamanlı olarak haritalama yapılmıştır.

Tasarlanan sistemde SLAM algoritması kullanılarak haritalama yapılmıştır. Kullanılan algoritma parçacık filtresi yöntemini kullanarak sensör verileri ve haritanın o ana kadar oluşturulmuş bölümlerine göre otonom aracın olası konumlarını takip ederek ortam haritasını oluşturmuştur (Şekil 3.17).



Şekil 3.17. Haritalamanın adımları

Harita oluşturma işleminde otonom araç, haritası çıkarılacak ortam içerisinde hareket ettirilirken mümkün olduğunca fazla alanın kapsanması ve aynı noktanın birden fazla ziyaret edilmesinin, alan üzerinde ölçüm yapılmamış yer olmaması ve hatalı ölçümlerin yeni ölçümler ile düzeltilmesi açısından önemli olduğu görülmüştür. Haritanın gerçek ortamı hangi oranda ifade edebildiği bilgi teorisindeki aşağıdaki verilen entropi eşitliği (Dissanayake ve ark. 2000) ile ölçülmüştür. Burada; otonom aracın konumunu ifade eden parçacıklara ait sonsal dağılımın entropisi Eşitlik 3.18 ile bulunmuştur. Sonsal dağılım otonom araç konumu hakkında %100 kesinliği ifade eden 1' e yaklaştıkça entropi değeri 0' a yaklaşacaktır. Bu ölçüm işlemi ve gözle kontrolden üretilen harita çok doğru görünmüyorsa bir miktar daha yeni veri kaydedilebilir ve harita verisi toplanma aşamasında otonom araç daha yavaş sürülebilir.

$$H(X) = -\int_i^{\cdot} p(X = i) log p(X = i) di \qquad (3.18)$$

Haritalama işlemine başlamadan önce harita boyutunun ve harita çözünürlüğünün ne olacağı yazılımda belirtilmelidir. Farklı dönüş ve sıra takip işlemlerinin test edileceği simülasyon ortamının ölçüleri 20 m x 20 m ve hücre çözünürlüğü ise kullanılan sensörün ölçebileceği minimum uzaklık değeri olan 2 cm olarak alınmıştır. Harita boyutu uygulamaya göre istenilen büyüklükte ayarlanabilir.

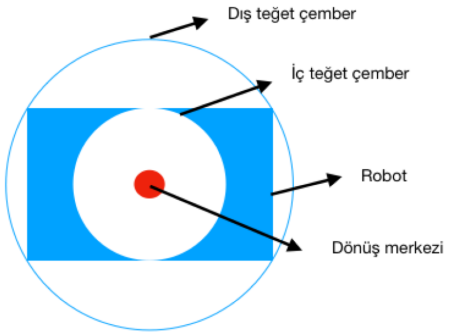
Üretilen haritanın kalitesi kullanılan sensörlerin ölçüm çözünürlüğü ve hassasiyeti ile doğrudan ilişkilidir. Ayrıca otonom aracı ortamda daha yavaş sürerek (özellikle dönüşler sırasında) üretilen engel haritasının kalitesi arttırılabileceği görülmüştür.

3.9.3. Otonom aracın fiziksel özelliklerine bağlı olarak navigasyonda kullanılacak engel katmanlarının harita üzerinde oluşturulması

Oluşturulan harita otonom aracın bulunduğu ortamdaki engelleri göstermektedir. Ancak otonom navigasyon için gerekli rota hesaplamalarını yaparken otonom aracın fiziksel özellikleri (izdüşümü ve dönüş yarıçapı dikkate alınarak bu engellere hangi mesafede kalması gerektiğini belirleyen bir tolerans katmanı oluşturulmuştur. Otonom araç izdüşümü aracın fiziksel sınırlarını ifade etmektedir. Yazılım içerisinde herbir (x, y) çifti otonom aracın bir köşesini ifade edecek şekilde aşağıda verildiği gibi iki boyutlu bir matris ile gösterilmiştir.

$$\begin{vmatrix} x_0 & y_0 \\ x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \end{vmatrix} \tag{3.19}$$

Bu izdüşüm iç teğet ve dış teğet çemberlerin yarıçaplarını hesaplamak için kullanılmıştır. Hesaplanan bu yarıçaplar ile tolerans katmanı tanımlanmıştır. Otonom aracın izdüşümünü hesaplamak için CAD çizimlerinden elde edilen ölçüler kullanılmıştır (Şekil 3.18).



Şekil 3.18. Otonom araca ait izdüşüm ve teğet çemberler

Harita oluşturmada engel olan hücreler 255, engel olmayan hücreler 0 ve ölçüm yapılmamış belirsiz hücreler -1 değeri almıştır. Otonom aracın izdüşüm ölçülerine göre engel etrafında otonom araç iç teğet çemberi içerisinde kalacak hücreler engel değerinin %75' ini, belirtilen tolerans katsayısı dış teğet çember ile çarpılarak hesaplanan tolerans katmanı içerisinde kalan hücreler ise engel değerinin %50' sini almıştır (Joseph 2018).

Her bir hücre için hesaplanan bu değerler rota planlama aşamasında kullanılmıştır. Rota hesaplanırken rota üzerindeki hücrelerin değerleri yazılım tarafından toplanmış ve en az değere sahip varyasyon otonom araç yazılımı tarafından seçilmiştir. Otonom aracın hareket ederken bir engele çarpmaması için dönüş merkezinin engel ile olan mesafesinin, iç teğet çember yarıçapından daha az olmaması gerekmektedir.

İç teğet çember ve tolerans katmanı hücre değerlerinin çok küçük veya büyük seçilmesinin global rota planlayıcı tarafından hesaplanan rotanın kalitesini etkilediği görülmüştür. Bu yüzdelik değerler çok büyük seçildiğinde hücre değerleri doğrusal olarak artacağından aslında otonom aracın geçmesi mümkün olan dar koridorlardan geçecek şekilte rota planlaması yapılamamıştır.

Harita oluşturduktan sonra bu haritanın tasarlanan lokalizasyon, rota planlama ve ilaçlama uygulamaları içerisinde kullanılacak şekilde, oluşturulan bir sunucu tarafından otonom navigasyon sistemine verilmiştir. Bu bilgiler oluşturulan lokalizasyon, rota planlama ve ilaçlama modülleri tarafından kullanılmış ve harita rviz aracılığı ile görselleştirilmiştir.

3.10. Otonom Aracın Harita Üzerinde Lokalizasyonunun Sağlanması

Otonom bir aracın yapabileceği en temel işlem çevresindeki ortamda dolaşabilmektir. Bunu efektif bir şekilde yapabilmek için otonom araç çevresinde nasıl bir ortam olduğunu ifade eden bir haritaya sahip olmalı, bu ortamın neresinde olduğunu bilmelidir. Bilinen bir harita üzerinde otonom aracın nerede olduğunu bilme işlemine lokalizasyon denilmektedir (Dellaert ve ark. 1999).

3.10.1. Adaptif monte karlo lokalizasyon algoritması

ROS ortamında adaptif monte karlo lokalizasyon algoritması kullanılarak otonom araç için bir lokalizasyon uygulaması geliştirilmiştir. LIDAR ve odometri sensörlerinden alınan veriler ve harita verileri karşılaştırılarak otonom aracın harita üzerindeki konumu bulunmuştur.

Monte Karlo Lokalizasyon algoritması, parçacık filtresi lokalizasyonu olarak ta bilinmektedir (Rekleitis 2004). Bu algoritma parçacık filtresi yöntemini kullanarak (Dellaert ve ark. 1999, Thrun ve ark. 2001) otonom aracın harita üzerindeki konumu bulmaktadır. Haritası verilmiş bir ortamda parçacık filtresi algoritması, otonom araç hareket edip çevresini algıladıkça otonom aracın harita üzerindeki pozisyonunu tahmin eder (Thrun 2005). Algoritma parçacık filtresi kullanarak otonom aracın olası pozisyonlarının dağılımlarını ifade etmektedir. Her bir parçacık otonom aracın bulunabileceği olası pozisyonu ifade eden bir hipotez anlamına gelmektedir (Thrun 2005). Algoritma genellikle durum uzayı (harita üzerinde) üzerinde parçacıkların rastgele ve tekdüze dağılımı ile başlar. Rastgele ve tekdüze dağılım otonom aracın ortamla ilgili hiçbir bilgisi olmadığı ve durum uzayının herhangi bir yerinde bulunma ihtimalinin birbirine eşit olması anlamına gelir (Thrun 2005). Otonom araç hareket ettikçe parçacıklar hareket kadar ötelenir ve hareketten sonraki yeni konumu tahmin eder. Otonom araç bir engel algıladığı zaman parçacıklar yeniden örneklenir tahmin edilen pozisyonla ölçülen

verinin ilişkilendirilmesi sağlanır. Sonuç olarak parçacıklar otonom aracın gerçek pozisyonuna doğru yakınsar (Thrun 2005).

Adaptif Monte Karlo Lokalizasyon Algoritması parametrik olmayan bir yöntem olduğu için tercih edilmiştir. Böylece parametrik olan Kalman Filtresi gibi başka lokalizasyon algoritmalarında karşılaşılan normal dağılıma sahip durumlarda iyi performans gösterirken multimodal durumlarda (Örneğin oda da bulunan kapıları algılayabilir ancak iki kapı arasındaki farkı algılayamaz) yeterli performans vermeme sorununa çözüm bulunmuştur (Dellaert ve ark. 1999).

Kullanılan lokalizasyon algoritmasının işlemsel karmaşıklığı (O(n)) parçacık sayısı ile doğrusaldır. Doğal olarak daha fazla parçacık daha kesin konum sonucu verirken işlem hızını azaltmaktadır. Bu nedenle hız-kesinlik arasında optimum bir N (parçacık sayısı) değeri aranmıştır.

Parçacık sayısını seçerken kontrol komutu (u_t) ve LIDAR sensör okuması (z_t) gelene kadar ekstra parçacık üreten adaptif yöntem kullanılmıştır (Thrun ve ark. 2005). Bu sayede otonom aracın diğer fonksiyonlarına engel olmadan bilgisayar işlem gücü ihtiyacını düşük tutarken, mümkün olan en fazla sayıda parçacık üretilmiştir. Aynı zamanda bu uygulama adaptif yapıda olduğu için daha hızlı bir işlemci kullanıldığında da daha fazla parçacık üretilebilir ve daha doğru konumlama sağlanabilir.

Klasik Monte Karlo Lokalizasyon algoritması uygulamanın zayıf yanlarından birisi otonom aracın hareket etmeyip sabit bir noktada beklediği durumlardır. Parçacıkların tümünün hatalı bir duruma yakınsadığında veya parçacıklar gerçek pozisyona yakınsadıktan sonra dışarıdan bir müdahale ile otonom aracın bir yerden alınıp yeni bir konuma konulduğunda tekrar doğru konumun bulunması mümkün olmayacaktır. Bu problem genellikle parçacık sayısının 50'den az olduğu ve büyük bir durum uzayına yayıldıkları durumlarda karşılaşılır (Thrun 2005).

Bu problemi ortadan kaldırmak için monte karlo lokalizasyon algoritmasının yeniden örnekleme bölümünde adaptif parçacık sayısı kullanımı sağlanmıştır. Aynı zamanda her iterasyonda ekstra parçacık eklenerek otonom aracın herhangi bir zamanda küçük bir olasılıkla

haritanın rastgele bir noktasına olabilecek kaçırma durumuna karşı dayanımlı olması sağlanmıştır (Thrun, 2002, Guan ve ark. 2019).

Meyve bahçesi üzerinde hareket edecek otonom aracın pozisyonu için sıra üzerindeki konumunu ifade eden x ve y koordinatları ile ve yönünü ifade eden θ açısı olmak üzere üç sayısal değer ile ifade edilmiştir (x, y, θ). Kullanılan lokalizasyon algoritmasında otonom aracın herhangi bir t zamanında olabileceği olası pozisyonlar parçacık olarak isimlendirilmektedir ve aşağıdaki gibi gösterilmiştir.

$$X_t = x_{t[1]}, x_{t[2]}, \ldots, x_{t[N]} \quad (3.20)$$

Harita üzerinde daha fazla parçacık içeren bölümler otonom aracın olma ihtimali daha yüksek olan, az parçacık içeren kısımlar ise bulunma ihtimali daha az olan yerleri ifade eder. Kullanılan lokalizasyon algoritması bulunulan konumun sadece bir önce bulunulan konuma bağlı olduğunu kabul eder (Çizelge 3.8). Yani X_t sadece X_{t-1}' e bağlıdır (Thrun ve ark. 2005).

Odometri sensörlerinden ölçülen kontrol komutlarını (u_t) ve LIDAR sensörden gelen veriyi (z_t) alır. Algoritma çıktı olarak t anındaki pozisyona ait olasılıksal dağılımı (X_t) verir.

Çizelge 3.8. Adaptif monte karlo lokalizasyon algoritması (Thrun 2005)

Adaptif Monte Karlo Lokalizasyon Algoritması(X_{t-1}, u_t, z_t, M):
  $\overline{X_t} = X_t = \emptyset$
  for n=1 **to** N
    $x_t^n \sim p(x_t | x_{t-1}^n, u_t)$ // hareket güncellemesi
    $w_t^n = p(z_t | x_t^n, M)$ // sensör güncellemesi
    $\overline{X_t} = \overline{X_t} + \langle x_t^n, w_t^n \rangle$
  endfor
  for n=1 **to** N
    en yüksek olasılığa sahip parçacıkların seçilmesi w_t^n
    $X_t = X_t + x_t^n$
  endfor
  return X_t

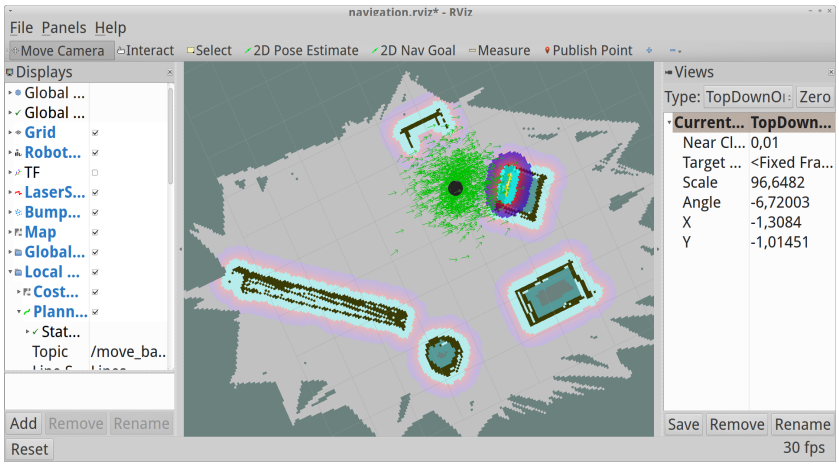
Hareket güncellemesi sırasında otonom araç yeni konumu verilen kontrol komutuna göre her bir parçacığa bu hareket uygulanarak güncellenmiştir (Rekleitis 2004). Hareket güncellemesi sırasında sistemin doğasından kaynaklanan ve önlenemeyen hatalar nedeniyle hareket güncellemesi sırasında parçacıklar sapma gösterir ve otonom aracın pozisyonu hakkındaki kesinliği azalmaktadır.

Sensör güncellemesi sırasında otonom araç, olası konumunu ifade eden parçacıkları, yeni hesaplanan konuma ve haritaya göre güncellemiştir. Her bir parçacık için sensörden alınan ölçüme göre olasılık hesaplanmış ve bu değerle orantılı olacak her parçacığa bir $w_{t[i]}$ katsayı atanmıştır. Daha sonra $w_{t[i]}$ katsayısı ile doğrusal şekilde bir önceki konuma ait olasılıksal dağılımdan rastgele olarak N yeni parçacık çekilmiştir. Sensörden alınan ölçümlerle uyumlu olan parçacıkların seçilme olasılıkları daha fazladır, uyumsuz parçacıkların ise seçilme olasılığı daha az olduğundan ölçüm sonuçlarına göre parçacıkların belirtilen limitler içerisinde otomatik olarak değişmesi sağlanmıştır. Otonom araç simülasyon alanı üzerinde ilerleyip ölçümler aldıkça ve haritayla karşılaştırdıkça tahmin edilen konum gerçek pozisyonuna doğru yakınsamıştır.

3.10.2. Geliştirilen lokalizasyon uygulaması

Geliştirilen lokalizasyon uygulaması verilen başlangıç konumuna göre parçacıkları (tüm aday konumları) hesaplamıştır. Lokalizasyon uygulamasının görselleştirilmesi için rviz aracı kullanılmıştır. Şekil 3.19' da denemeler sırasında simülasyon ortamından alınan örnek bir ekran görüntüsü görülmektedir ve görülen yeşil oklar lokalizasyon yazılımı tarafından tahmin edilen parçacıkları göstermektedir. Otonom araç hareket ettikçe sensörlerden gelen ölçümler ile olası konumu ifade eden parçacıkların haritayla karşılaştırılması yapılmıştır. Her bir aday konum haritayla karşılaştırıldığında uyum yüksekse parçacığın olasılıksal değeri artmakta, uyumlu değil ise olasılık değeri azalmaktadır. Adaptif Monte Karlo Lokalizasyonu kullanıldığı için otonom araç ilerledikçe çok düşük olasılık değerine sahip parçacıklar lokalizasyon uygulaması tarafından otomatik silinmiş ve tahmin edilen konum otonom aracın gerçek

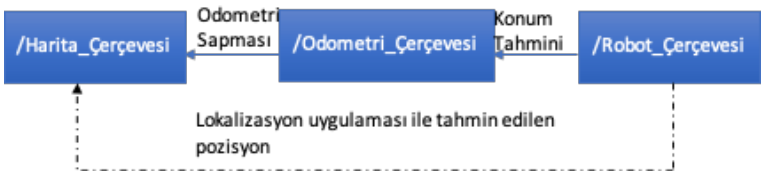
konumuna yakınsamıştır. Parçacıklar otonom aracın hareket yönünde ve ölçülen hareket kadar ilerlemektedir.



Şekil 3.19. Otonom araca ait lokalizasyonun görselleştirilmesi

Başlangıç konumunu doğru verebilmek için LIDAR sensörden okunan veriler harita ile rviz ekranında karşılaştırılarak pozisyon doğruluğu kontrol edilmiştir. Başlangıç pozisyonu seçildikten sonra otonom araç harita üzerinde ilerleyip ölçümler aldıkça gerçek pozisyonuna yakınsamıştır. Eğer otonom aracın harita üzerindeki pozisyonunu bulmak mümkün değilse parçacıklar başlangıç durumunda tüm harita üzerine tekdüze olarak dağıtılmıştır. Bu durumda otonom aracın gerçek pozisyonuna yakınsaması başlangıç konumu verilmesine göre biraz daha fazla zaman almıştır.

Kullanılan lokalizasyon yöntemi LIDAR sensöre sahip otonom araçlarda kullanılmak üzere tasarlanmıştır. Odometri koordinat çerçevesinden alınan ölçümlerden otonom aracın harita üzerinde konumunu belirlemek için otonom aracın fiziksel ölçülerine ve dönüş merkezinin konumuna göre geometrik bir dönüşümün yapılması gerekmektedir. Odometri ölçüm verileri ve otonom araç bilgileri girildikten sonra ROS' ta bulunan geometrik dönüşüm sunucusu bu dönüştürme işlemini otomatik olarak yapmıştır (Şekil 3.20).

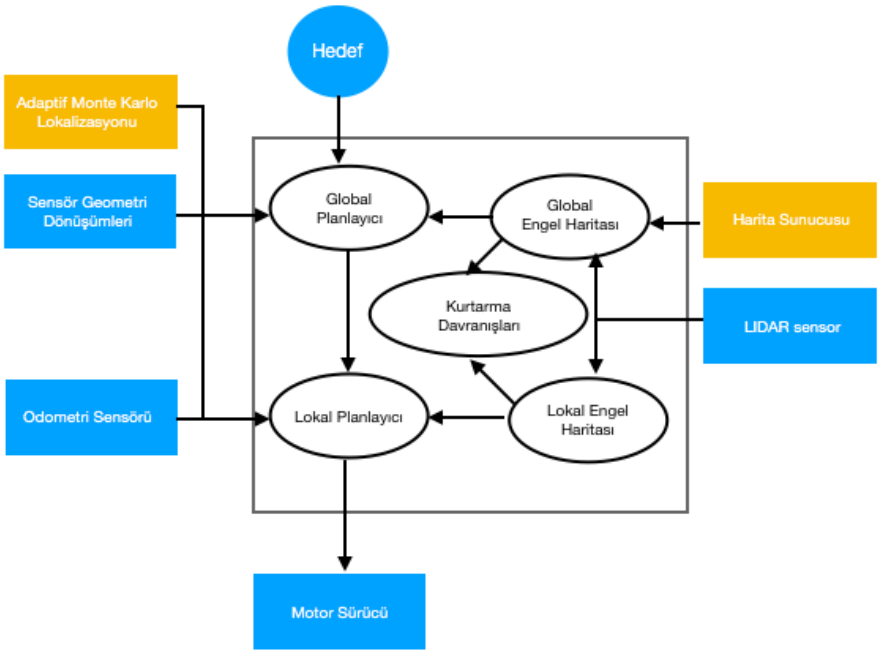


Şekil 3.20. Odometri verilerinin otonom aracın harita üzerindeki konumuna dönüştürülmesi

3.11. Rota Planlaması ve Otonom Sürüş

Otonom aracın bulunduğu ortamı ifade eden haritanın elde edilmesi ve aracın bu harita üzerinde nerede olduğunun lokalizasyonla belirlenmesinden sonra otonom sürüş için son adım olarak rota planlamasının yapılması ve bulunulan noktadan hedef noktaya/noktalara hareket edilmesini sağlayacak rota planlama uygulaması geliştirilmiştir.

Navigasyon aşamasında ortam haritası, otonom aracın bulunduğu konum, LIDAR ve odometri sensörlerinden gelen veriler birleştirerek bulunulan pozisyondan hedef pozisyona en kolay şekilde nasıl gidileceği hesaplanmış ve bu rota otonom aracın fiziksel yetenek ve limitlerine göre en optimum şekilde izlenmeye çalışılmıştır. Otonom aracın harita üzerinde ilerlerken karşısına çıkan engellerden sakınması sağlanmıştr. Otonom araç bir engel üzerinde sıkışıp kalması durumunda belirlenmiş kurtarma manevralarını yapacak ve bu engelden kurtulmak mümkün değilse güvenlik açısından duracak şekilde programlanmıştır.



Şekil 3.21. Navigasyon sistemin genel çalışma planı

Rota planlama ve navigasyon sistemi Şekil 3.21 ' de gösterildiği haliyle şu şekilde çalışır.

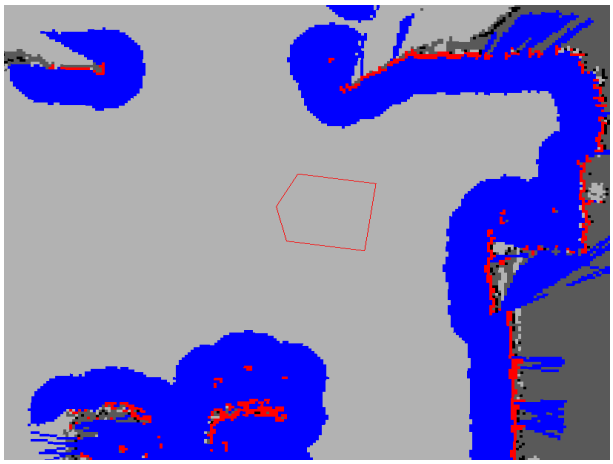
- Gidilmek istenilen hedef navigasyon sistemine girdi olarak verilmiştir.

- Global planlayıcı lokalizasyon uygulaması ile belirlenen otonom araç pozisyonu, geometrik olarak otonom araca göre dönüştürülmüş odometri ve LİDAR sensör verileri ve harita verilerini girdi olarak almıştır.
- Hedef konuma ulaşmak için en kısa yol global planlayıcı tarafından hesaplanmıştır.
- Yapılan bu plan lokal planlayıcıya gönderilmiş ve lokal planlayıcı LIDAR sensörden aldığı verilere göre önünde bulunan engellerden sakınmıştır. Lokal planlayıcı bir problemle karşıltığında (kaçınılmaz bir çarpışma algılarsa) ve ilerleme sağlayamazsa global planlayıcıya geri dönerek yeni bir plan yapılmasını sağlayacak şekilde programlanmıştır.
- Planlanan hareket otonom araca hareket veren motor sürücülerinin anlayacağı bir mesaj olarak gönderilmiştir.
- Otonom araç hedefe ulaştığında bir sonraki hedefi alana kadar bu konumda kalmıştır.

3.11.1. Geliştirilen rota planlama uygulaması

Otonom aracın istenilen bir A noktasından B noktasına gitmesi problemine rota planlama denilmektedir. Rota planlamada otonom aracın mümkün olan en kısa sürede, harita üzerindeki engellere çarpmadan ilerlemesi ve ilerleme sırasında önüne çıkan engellerden de kaçınması gerekmektedir (Dolgov ve ark. 2010, Zheng ve ark. 2016).

Bu adımları gerçekleştirmek amacıyla rota planlama, global ve lokal olmak üzere iki kısımda programlanmıştır. Rota planlama oluşturulan engel haritası, harita üzerindeki herbir hücre için hesaplanmış olan hücre değerleri dikkate alınarak yapılmıştır (Şekil 3.22). Harita üzeride oluşturulan harita katmanları sayesinde otonom aracın dönüş merkezinin engellere belirlenen tolerans değerlerinden fazla yaklaşmaması sağlanarak engellere çarpmadan hareket etmesi sağlanmıştır.



Şekil 3.22. rviz ortamında harita üzerindeki tolerans katmanının görselleştirilmesi

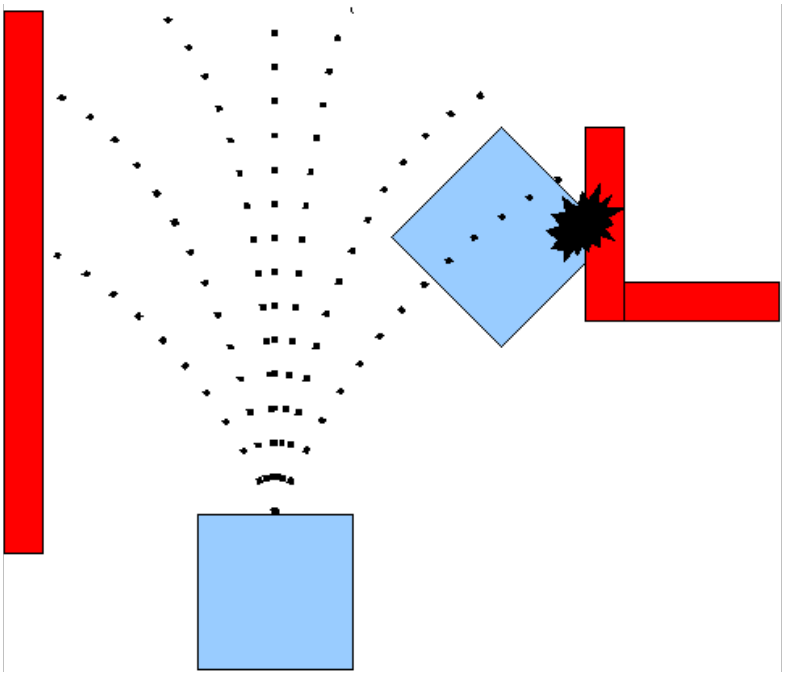
Global rota planlayıcı: Global rota planlama adımında engel haritasında otonom aracın bulunduğu noktadan hedef noktaya/noktalara gitmesini sağlayacak rota planlama uygulaması geliştirilmiştir. Bu amaçla rota planlamada Dijkstra Rota Planlama algoritması kullanılmıştır (Sadeghi-Niaraki ve ark. 2011).

Çizelge 3.9. Dijkstra en kısa rota algoritması (Sadeghi-Niaraki ve ark. 2011)

```
1:  Dijsktra Algoritması (harita, başlangıç)
2:    dist[başlangıç] = 0      //Başlangıç noktasından başlangıç noktasına olan mesafe 0
      olarak ayarlanır
3:    for v in harita:          // Başlatma ayarları
4:      if v ≠ başlangıç
5:        dist[v] = ∞  // Başlangıçtan diğer noktalara olan mesafe sonsuz olarak ayarlandı.
6:      add v to Q          // Bütün noktalar Q kümesine eklendi
7:    while Q is not empty:    // Ana döngü
8:      v = min dist[v] in Q
9:      remove v from Q
10:     for each neighbor u of v:
11:       alt = dist[v] + length(v, u)
12:       if alt < dist[u]:   // v' den u noktasına en kısa yol bulundu
13:         dist[u] = alt  // u noktasına olan mesafeyi güncelle
14: return dist[]
```

Çizelge 3.9.' da çalışma şekli gösterilen dijkstra algoritmasında dist, başlangıç noktasının harita üzerindeki diğer tüm noktalara olan uzaklığını gösteren bir kümedir. Başlangıç noktasından diğer tüm noktalara olan mesafe öncelikle ∞ olarak atanmıştır. Başlangıç ve komşu noktalar arasındaki hücrelerin değerlerine göre çevredeki tüm komşu noktaların mesafe değerleri bulunmuş ve dist kümesi içerisinde saklanmıştır. Böylece verilen bir başlangıç noktasından bitiş noktasına en kısa yol bulunmuştur

Lokal Rota Planlayıcı: İki aşamalı olarak programlanan rota planlama uygulamasında global rota planlayıcı tarafından hesaplanan genel plan, otonom aracın LIDAR sensör görüş alanı içerisinde engellerden kaçmasını sağlayacak şekilde bir lokal planlayıcı tarafından uygulanmıştır. Lokal rota planlaması için dinamik pencere yaklaşımı kullanılmıştır (Fox ve ark. 1997). Global plandan alınan rota bilgisi LIDAR sensörün görüş alanı içerisindeki engeller gözönüne alınarak tekrar ele alınmış ve lokal planlayıcıdan elde edilen hareket komutu otonom araca uygulanmıştır. Lokal planlamanın başarısı otonom aracın fiziksel özelliklerine uygun şekilde ayarlanması gereken simülasyon süresi değişkenine bağlıdır (Fox. ve ark. 1997).



Şekil 3.23. Lokal rota planlayıcada simülasyon aşaması

Dinamik Pencere Yaklaşım algoritması aşağıdaki şekilde çalışmaktadır:

- Otonom aracın kontrol uzayı (d_x, d_y, d_θ) maksimum hız değerine göre ayrık olarak rastgele örneklenmiştir.
- Otonom aracın bulunduğu durumdan belirtilen simülasyon süresince örneklenen hızda ilerlemesi durumunda gerçekleşecek sonuç tahmin edilmiştir (Şekil 3.23).
- İkinci adımda yapılan simülasyona göre elde edilen olası rotalar engele olan mesafe, hedefe olan mesafe, global rotaya olan mesafe ve hız değerlerine göre karşılaştırılmıştır. Bir engelle çarpışmaya sebep olan rotalar gözardı edilmiştir.
- En yüksek skora sahip rota kabul edilerek hız komutu otonom aracın hareket donanımlarına gönderilmiştir.
- Bu işlem hareket süresince tekrarlanmıştır.

Dinamik pencere yaklaşımında otonom aracın bulunduğu konuma göre optimal rotayı verecek (v (doğrusal hız m/s), μ (açısal hız rad/s)) çiftleri üretilmiştir (Guan ve ark. 2018).

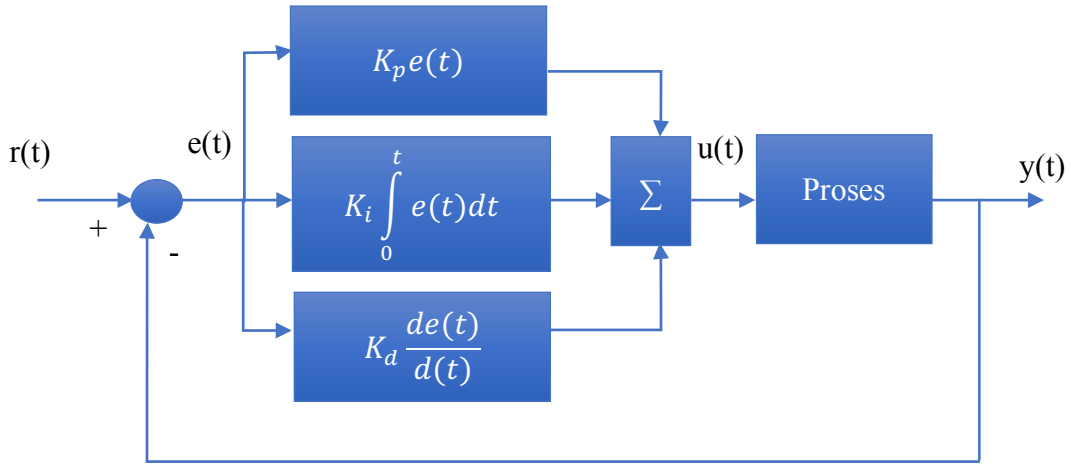
Dinamik pencere yaklaşımı algoritması ile programlanan lokal planlayıcı, otonom aracın kontrol uzayından rastgele olarak örneklediği hız değerlerini almış ve bu örnekler simülasyonda belirtilen süre boyunca uygulandığı kabul edilerek elde edilerek rotalar elde edilmiştir. Her rota program tarafından, rotanın geçtiği hücre değerleri toplanarak karşılaştırılmıştır. Bir engel ile kesişen rotaları üreten hızlar elemine edilmiş, en düşük değere sahip rota lokal rota planlayıcı tarafından seçilmiştir. Rotalar örneklenmiş hızın, program içerisinde simülasyon süresi parametresiyle belirtilen süre boyunca uygulandığı kabul edilerek üretilmiştir. Simülasyon süresi için verilen süre arttıkça ihtiyaç duyulan bilgisayar hesaplama gücünün doğrusal olarak arttığı gözlenmiştir (Tomovic 2014).

3.11.2. Otonom aracın donanım seviyesinde kontrolü

Otonom aracın hareketi diferansiyel sürüş yöntemi ile sağlanmıştır. Otonom aracın sağ ve sol tarafında birer adet bulunan elektrik motorlarına bir motor sürücü aracılığı ile gönderilen doğrusal ve açısal hız komutunun uygulanması ve uygun sınırlar içerisinde kalmasını sağlamak amacıyla kullanılacak motor sürücü tarafından sağlanan adaptif Oransal-Integral-Türevsel (PID) kontrol uygulanmıştır. PID kontrol yöntemi endüstride proses kontrolünde yaygın bir

şekilde kullanılmaktadır. PID kontrolörler tasarım esnekliği ve güvenilirlikleri nedeniyle yaygın bir şekilde kullanılmaktadır (Dorf ve Bishop 2017).

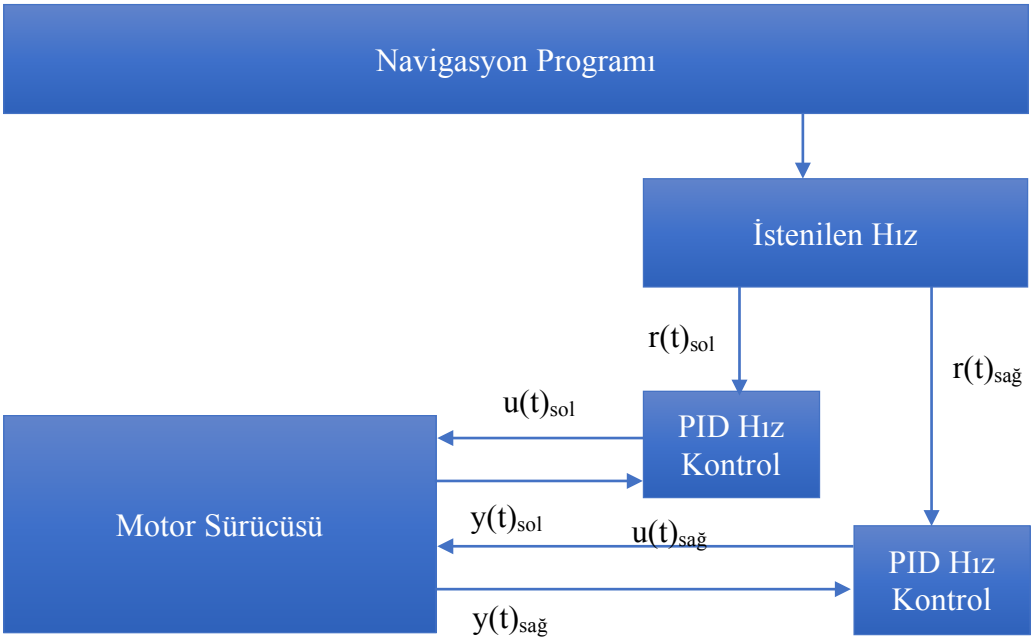
PID kontrol yöntemi ile navigasyon uygulaması tarafından otonom araca gönderilen hareket komutunun istenilen sınırlar içerisinde uygulanması sağlanmıştır. Şekil 3.24' te görülen PID kontrol diyagramında



Şekil 3.24. PID kontrol diyagramı (Dorf ve Bishop 2017)

r(t) ulaşılmak istenilen referans doğrusal ve açısal hareketi,
y(t) otonom aracın sahip olduğu doğrusal ve açısal hareketi,
e(t) referans ve çıkış arasındaki farkı,
u(t) ise elde edilen hataya göre motor sürücü tarafından motorlara uygulanacak düzeltmeyi ifade eder.

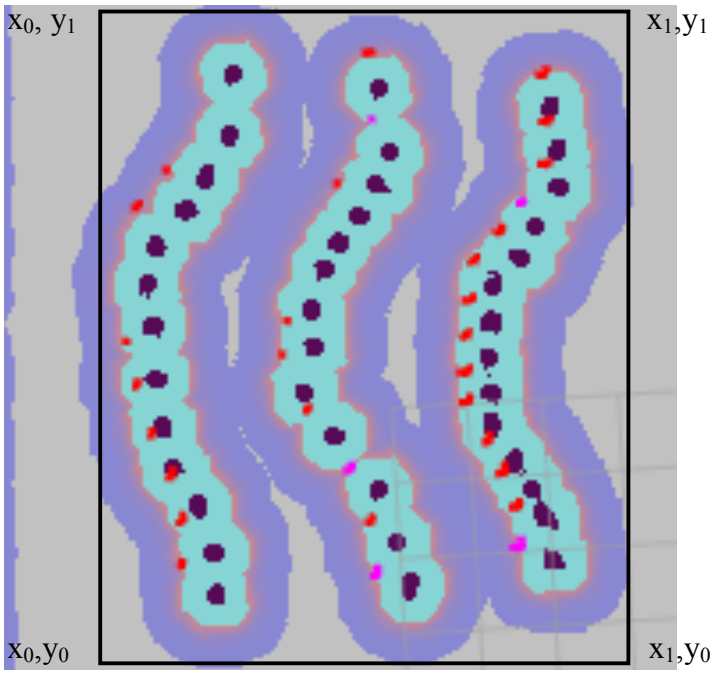
Geliştirilen programda hedefe ulaşmak için üretilen hız komutu motor sürücüsünde entegre olarak bulunan PID hız kontrol ünitesine gönderilmektedir (Şekil 3.25). Hızın istenilen referans sinyalle karşılaştırılması diferansiyel sürüşlü bir otonom araç tasarımı kullanıldığı için sağ ve sol motor için birbirinden bağımsız olarak yapılmaktadır.



Şekil 3.25. ROS ortamında PID kontrol uygulaması

3.12. İlaçlama Sisteminin Kontrolü

Otonom araç değişken düzeyli ilaçlama sistemini taşıyacaktır. Kullanılacak değişken düzeyli ilaçlama sistemi sıra merkezine olan mesafeye göre ağaç kanopi hacmini hesaplamakta ve hesaplanan değere göre farklı miktarda ilaç püskürtmektedir. Bu nedenle ilaçlama kollarının sıra merkezine olan mesafesinin bilinmesi önemlidir (Önler ve ark. 2015). Bu değişken düzeyli ilaçlama sisteminin elektriksel olarak açılıp kapanmasını sağlayan bir program geliştirilmiştir. Ortam haritası oluşturulduktan sonra ağaç sıralarını içerisine alan bir dörtgen belirlenmiştir (Şekil 3.26). Otonom araç ortam üzerinde ilerlerken lokalizasyon uygulaması tarafından algılanan otonom araç konumunun bu dörtgen sınırları içerisinde olup olmadığı ((x_0, y_0) ve (x_1, y_1)) kontrol edilmiştir. Eğer otonom araç bu sınırlar içerisinde hareket ediyorsa değişken düzeyli ilaçlama sistemine açma komutu, sınırlar dışında hareket ediyorsa ise kapama komutu verecek bir mesaj oluşturulmuştur.



Şekil 3.26. İlaçlama sınırlarının belirlenmesi

3.13. Analizler

Haritalamada farklı parçacık ve güncelleme değerlerine göre elde edilen entropi değerleri, lokalizasyonda farklı parçacık ve güncelleme değerlerine göre tahmin edilen ve gerçek konum arasındaki fark değerleri ve rota planlamada tahmin edilen konum, gerçek konum ve rota arasındaki fark değerlerinin tanımlayıcı istatistiksel analizleri pandas 0.23.4 versiyon Python veri analiz kütüphanesi kullanılarak yapılmıştır. Grafikler matplotlib 3.0.0 ve seaborn 0.9.0 versiyonlu Python görselleştirme kütüphaneleri kullanılarak oluşturulmuştur.

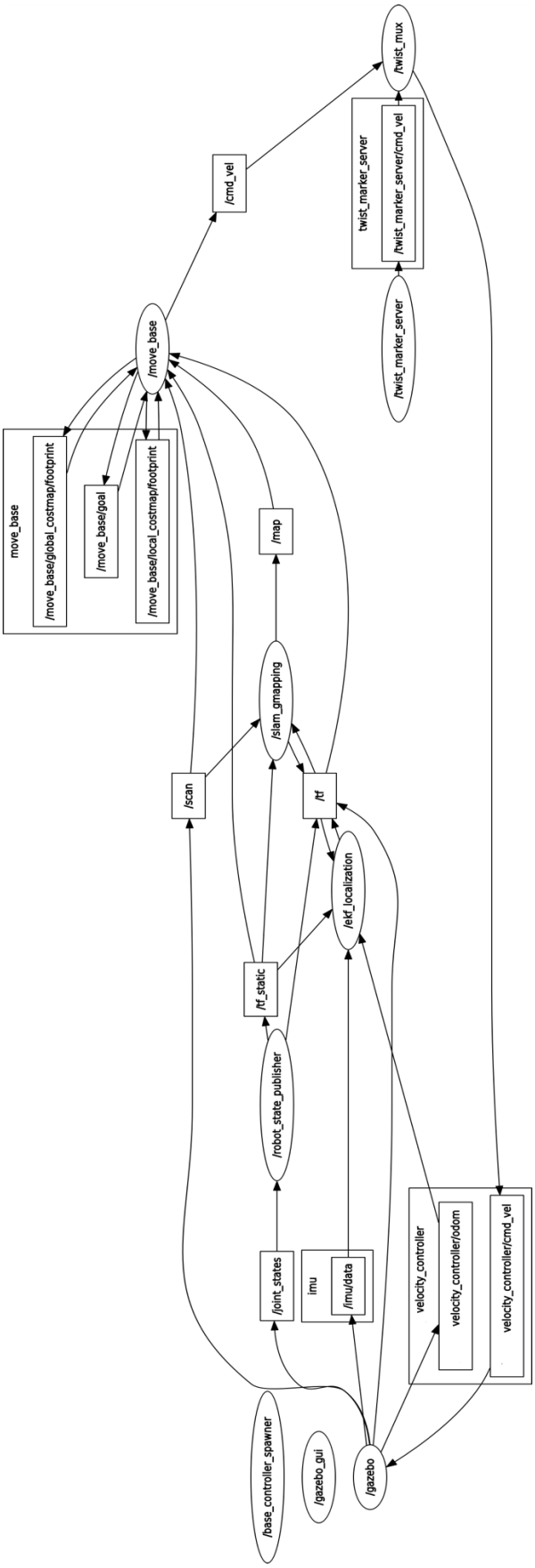
4. BULGULAR ve TARTIŞMA

LIDAR, odometri (atalet ve devir ölçer) sensörleri olan diferansiyel sürüş özelliğine sahip bir otonom araç ortamdaki engellerin haritasını çıkaran, geliştirilen bu harita üzerinde otonom aracın konumunu belirleyen, bulunduğu noktadan hedef nokta / noktalara gitmesini sağlayan ve sıra aralarında ilaçlama işlemini açan, sıra dışında ilaçlama işlemini durduran sinyal üreterek bunu ilaçlama sistemine gönderebilen bir yazılım geliştirilmiş ve simülasyon üzerinde farklı yazılım parametreleri ile test edilmiştir.

4.1. Haritalama

Bir otonom aracın bulunduğu ortamda otonom navigasyon yapabilmesi için ilk gereklilik bu ortamda bulunan sabit engellerin yerleşimlerini ve birbirlerine olan mesafelerini gösteren bir haritaya sahip olmasıdır. Bunun için öncelikli olarak SLAM algoritmasını kullanan bir haritalama uygulaması geliştirilmiştir. Haritalama uygulaması otonom navigasyon yapılacak ortamda otonom aracın dolaştırılması yoluyla, otonom araç tarafından otomatik olarak oluşturulmuştur. Harita otonom aracın sensörlerinden aldığı ölçümler aracılığıyla ortamı tanımlayan bir gösterim olarak *png veya *jpg formatında kaydedilmiştir. Resim formatındaki bu haritada her bir piksel belirtilen çözünürlükteki metrik değere karşılık gelmektedir. LMS-111 LIDAR sensörü 2 cm çözünürlükte ölçüm alabildiği için 2 cm x 2 cm piksel harita çözünürlüğü kullanılmıştır.

Şekil 4.1' de haritalama sistemini oluşturan yazılım kısımları görülmektedir. Otonom araç ve bu araca ait tüm sensör verileri simülatör (/gazebo) ortamında simüle edilmiştir. Haritalama modülünde (/slam_gmapping) LIDAR sensörden alınan veriler (/scan), odometri (atalet ve devir ölçer) sensörlerinden alınan alınarak birleştirilen veriler (/ekf_localization) ve dönüş merkezine göre sensörlerden alınan verilere uygulanacak geometrik düzeltme (/tf) girdi olarak işlenmiş ve buradan elde edilen sonuca göre harita (/map) üretilmiştir. Otonom aracın alan üzerinde ilerlemesi kullanıcı tarafından yapılan kontrol ile (/cmd_vel) gerçekleştirilmiş ve yeni konum tekrardan simüle edilmiş otonom araca iletilerek (/gazebo) bu döngünün devam etmesi sağlanmıştır. Harita üzerinde belirsiz yer kalmayana kadar bu işlem devam ettirilmiştir.



Şekil 4.1. Geliştirilen haritalama uygulamasına ait diyagram

Oluşturulan haritanın kalitesi bilgi teorisindeki entropi kavramıyla ölçülmüştür. Engel haritası parçacık filtresi yöntemini kullanan SLAM algoritması ile oluşturulmuştur. Elde edilen haritanın sonsal dağılımı kullanılarak haritaya ait entropi değeri haritalama işlemi süresince kaydedilmiştir.

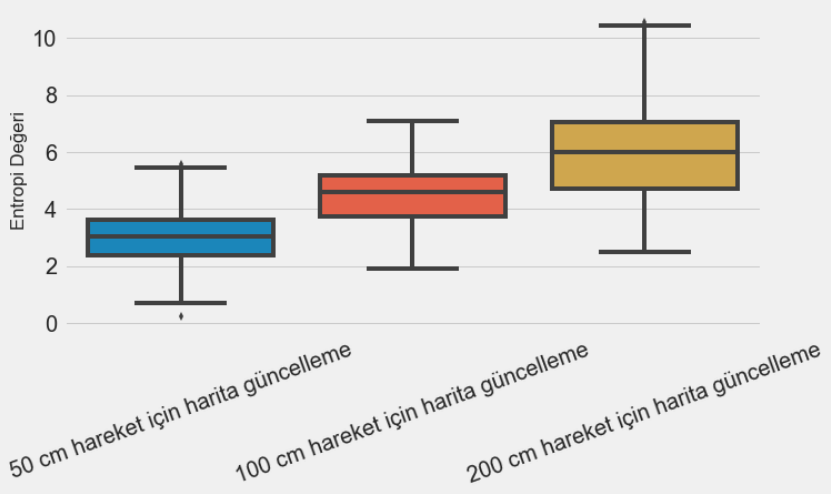
Haritanın bulunulan ortamı kesin olarak yansıttığı durumlarda entropi değeri 0' a eşit olmakta, haritanın ortamı yansıtmaktaki belirsizliği arttıkça ise bu değer büyümektedir. Entropi değeri haritanın ne kadar harekette güncelleneceği ile haritalama algoritmasında kullanılan parçacık sayısına bağlıdır. Farklı değerler için ortamın engel haritası çıkarılmasında elde edilen entropi değerleri karşılaştırılmıştır.

Haritanın güncellenmesi için gereken hareket değerleri 50 cm, 100 cm ve 200 cm seçilmiştir (Joseph 2018). Güncelleme değerinin küçültülmesi bilgisayar işlem gereksinimini arttırırken çok büyük seçilmesi ise doğru sonuçlar elde edilmesini önlemektedir. Haritalamada farklı güncelleme değerlerinin entropi üzerindeki etkisi araştırılırken parçacık sayısı bu gibi çalışmalarda önerilen 30 parçacık olarak kullanılmıştır (Grisetti ve ark. 2007)

Çizelge 4.1. Farklı güncelleme değerlerine göre harita oluşturmada ölçülen entropi

	Entropi (50 cm Hareket için Harita Hüncelleme)	Entropi (100 cm Hareket için Harita Güncelleme)	Entropi (200 cm Hareket için Harita Güncelleme)
Ortalama	3,03	4,48	5,81
Standard Sapma	1,13	1,17	1,95
Minimum Değer	0,2	0,62	1,70
Ortanca Değer	2,95	4,56	5,82
Maksimum Değer	5,39	7,62	11,92

Üç farklı konfigürasyon için simülasyon ortamında haritalama yapılmış ve entropi değerleri bakımından sonuçları karşılaştırılmıştır. Çizelge 4.1' de görüldüğü üzere en düşük ortalama entropi değeri 3,03 ile 50 cm hareket için güncelleme değerine sahip konfigürasyonda elde edilmiştir. En yüksek ortalama entropi ise 5,81 ile 200 cm hareket için güncelleme değerine sahip olan konfigürasyonda elde edilmiştir. Harita oluşturmada ilerleme hızına uygunluğu ve işlem gücü performans oranı en uygun olduğu için 50 cm harekette harita güncellemesi yapan konfigürasyon haritalama yazılımında kullanılmıştır.



Şekil 4.2. Farklı güncelleme değerlerine göre harita oluşturmada elde edilen entropi dağılımına ait kutu grafik

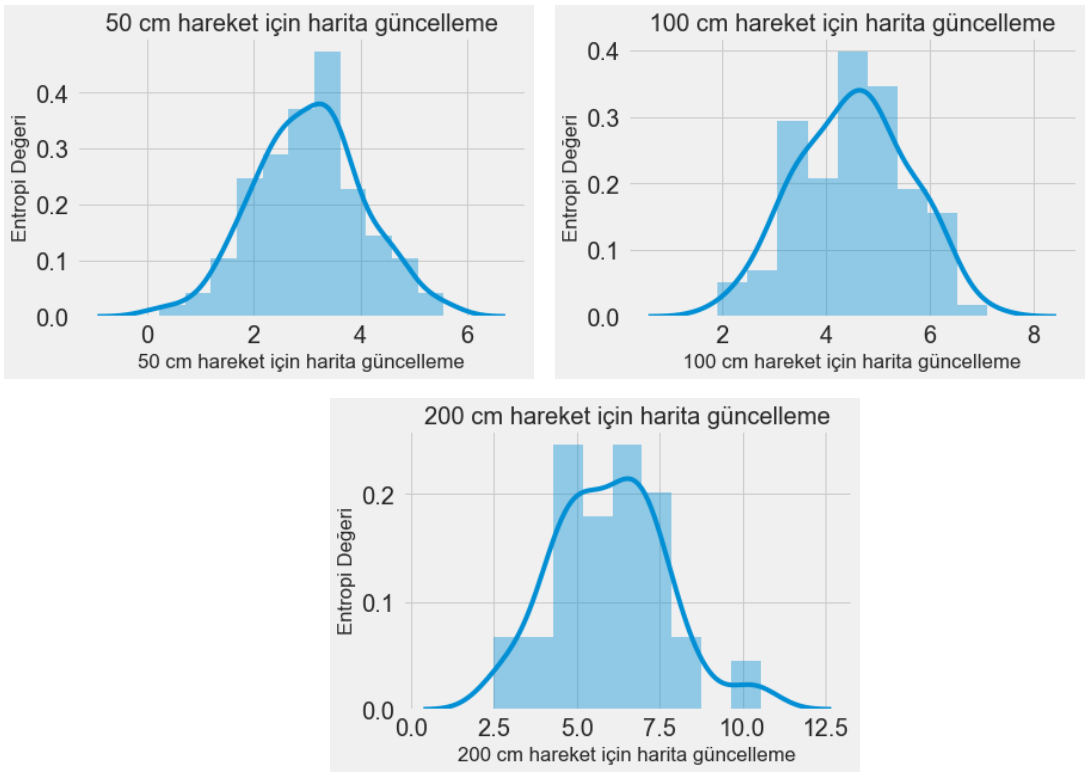
Şekil 4.3. Farklı güncelleme değerlerine göre harita oluşturmada elde edilen entropi dağılımına ait viyolin grafik

Yukarıdaki kutu ve viyolin grafik şekillerinde üç farklı konfigürasyon için entropi değerlerinin dağılımı görülebilir (Şekil 4.2, Şekil 4.3) Kutu grafik, değerlerin dağılımını görmek için uygundur ancak her bir grubun maksimum, minimum ve ortanca nokta arasındaki örneklem dağılımını görmek için viyolin grafik kullanılmıştır.

Harita güncellemesini 200 cm hareket edildikten sonra yapan konfigürasyonda entropi değerlerinde en yüksek varyans elde edildiği kutu ve viyolin grafiklerde görülmektedir. Örnekleme dağılımının 50 cm hareket için harita güncelleme değerine sahip konfigürasyonda

ortanca değer olan 2,95 entropi değeri etrafında, 1m hareket için harita güncelleme değerine sahip konfigürasyonda ortanca değer olan 4,56 entropi değeri etrafında üçüncü konfigürasyonda ise varyansı daha yüksek bir şekilde maksimum ve minimum noktalar arasında dağıldığı görülmektedir.

Üç farklı kongifürasyon için entropi değerlerinin dağılımları incelendiğinde normal dağılma sahip oldukları görülmektedir (Şekil 4.4). Gerekli bilgisayar işlem gücü ve entropi değeri dikkate alındığında 50 cm hareket için harita güncelleme yapan konfigürasyon geliştirilen uygulamada kullanılmıştır.



Şekil 4.4. Farklı güncelleme değerlerine göre harita oluşturmada elde edilen entropi değerlerinin dağılımı

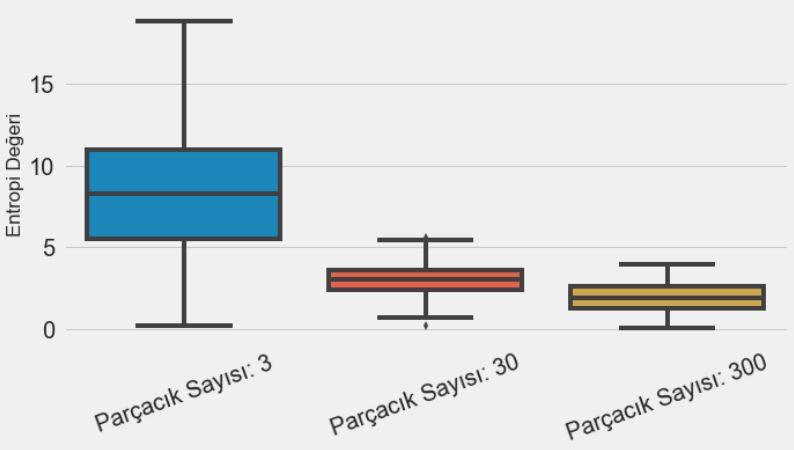
Haritalama algoritması, haritayı üretebilmek için parçacık filtresi temelli bir yöntem kullanmaktadır. Bu yöntemde haritaya ait sonsal olasılık dağılımı sensörlerden alınan ölçümlere göre parçacık adı verilen olası pozisyon (harita) hipotezleriyle oluşturulur. Ardışık olarak alınan ölçümler sonucunda sonsal olasılık dağılımı doğru harita versiyonuna doğru yakınsamaktadır. Sonsal olasılık dağılımını ifade edebilmek için kullanılan parçacık sayısı elde edilen haritanın kalitesini dolayısıyla entropisini doğrudan etkilemektedir. Parçacık sayısının entropi üzerindeki

etkisini ölçmek amacıyla 3, 30 ve 300 parçacık kullanılan üç farklı konfigürasyon üzerinden entropi ölçümü alınmıştır. Üretilen haritanın kalitesine pozitif etki sağlayan parçacık sayısı artışı diğer yandan ihtiyaç duyulan bilgisayar işlem gücünü doğrusal olarak arttırmaktadır. Bir önceki adımda bulunan 50 cm'lik yer değiştirmede harita güncelleme yapılması uygulanmıştır.

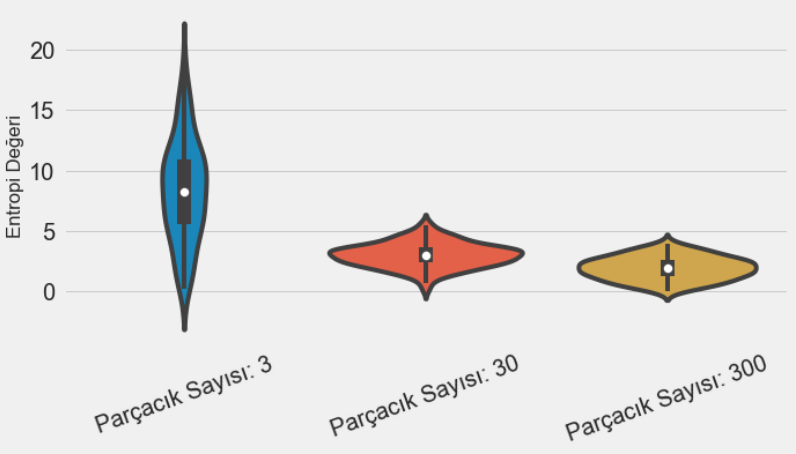
Çizelge 4.2. Farklı parçacık sayısı değerlerine göre harita oluşturmada elde edilen entropi

	Entropi (Parçacık Sayısı: 3)	Entropi (Parçacık Sayısı: 30)	Entropi (Parçacık Sayısı: 300)
Ortalama	6,61	3,03	2,06
Standard Sapma	3,99	1,13	0,98
Minimum Değer	0,13	0,20	0,12
Ortanca Değer	6,11	2,95	2,08
Maksimum Değer	17,28	5,39	4,25

Üç farklı değer seti için haritalama yapılmış ve entropi değerleri bakımından sonuçları karşılaştırılmıştır, Çizelge 4.2' de görüldüğü üzere en düşük ortalama entropi değeri 2,06 ile 300 parçacığa sahip konfigürasyonda elde edilmiştir. En yüksek ortalama entropi değeri ise 6,61 ile 3 parçacığa sahip konfigürasyonda elde edilmiştir. Harita oluşturmada 300 parçacık kullanımı ile 30 parçacık kullanımı arasında entropi ortalamaları bakımından %47 fark olmasına rağmen bilgisayar işlem gücü ihtiyacını 10 kat arttırdığı için geliştirilen uygulamada 30 parçacıklı versiyon kullanımı uygun bulunmuştur.



Şekil 4.5. Farklı parçacık sayısı değerlerine göre harita oluşturmada elde edilen entropi dağılımına ait kutu grafik

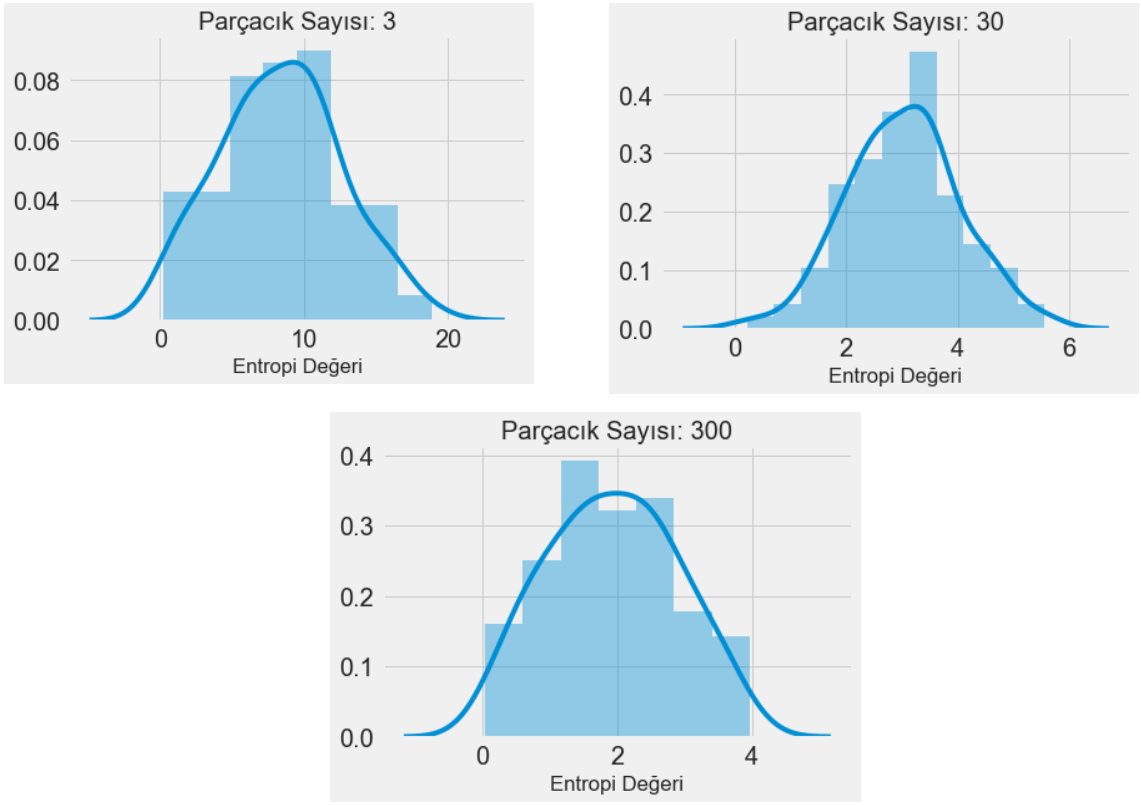


Şekil 4.6. Farklı parçacık sayısı değerlerine göre harita oluşturmada elde edilen entropi dağılımına ait viyolin grafik

Yukarıdaki kutu ve viyolin grafik şekillerinde üç farklı konfigürasyon için entropi değerlerinin dağılımı görülebilir (Şekil 4.5, Şekil 4.6). Kutu grafik, değerlerin dağılımını görmek için uygundur ancak her bir değerin maksimum, minimum ve ortanca nokta arasındaki örneklem dağılımını görmek için viyolin grafik kullanılmıştır.

En yüksek varyansın parçacık sayısının 3 olduğu konfigürasyonda olduğu kutu ve viyolin grafiklerde görülmektedir. Örnekleme dağılımının 300 parçacık kullanılan konfigürasyonda ortanca değer olan 2,08 etrafında, 30 parçacık kullanılan konfigürasyonda ortanca değer olan 2,95 etrafında, 3 parçacık kullanılan üçüncü konfigürasyonda ise varyansı daha yüksek bir şekilde maksimum ve minimum noktalar arasında dağıldığı görülmektedir.

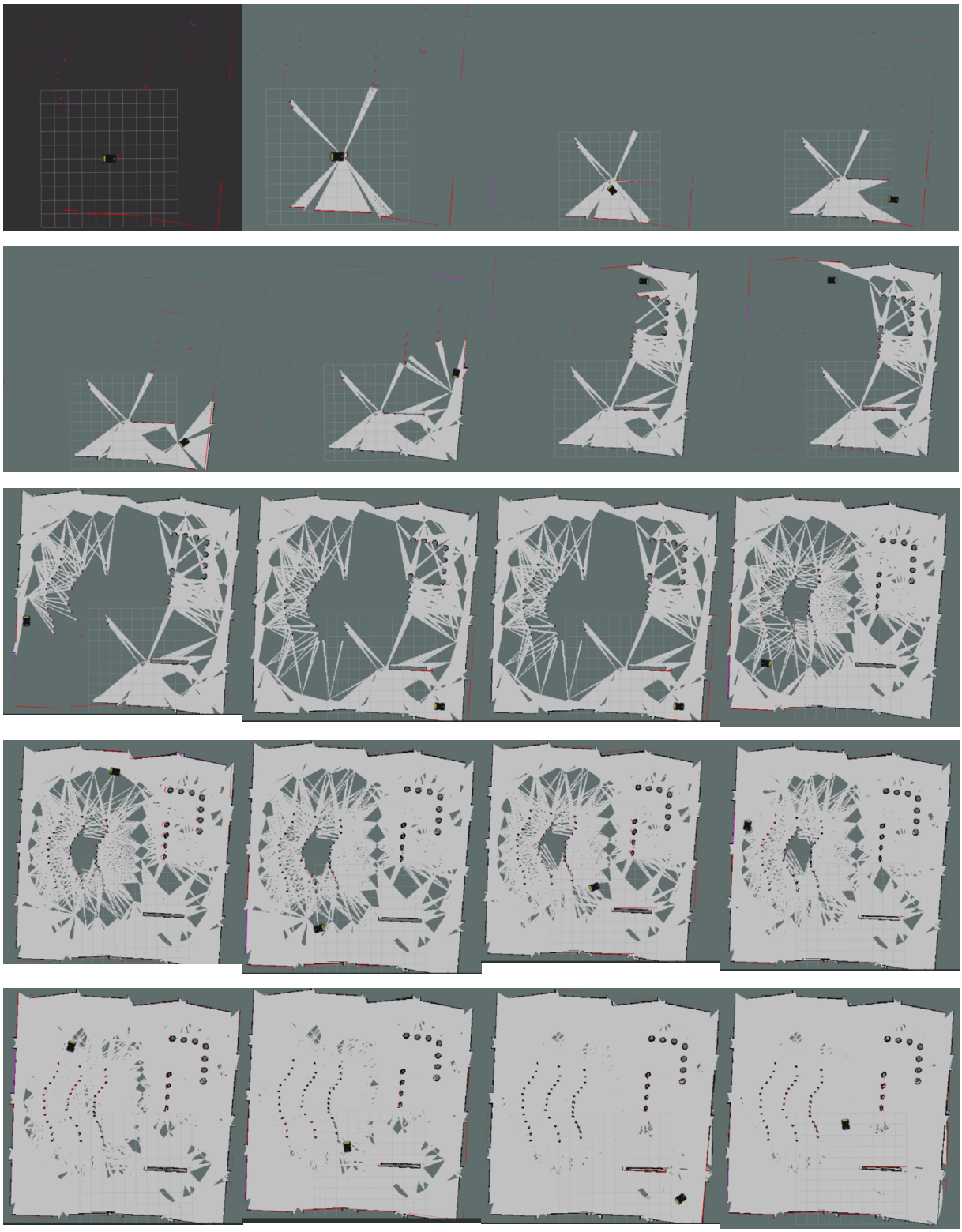
Üç farklı kongifürasyon için entropi değerlerinin dağılımları incelendiğinde normal dağılıma sahip oldukları görülmektedir (Şekil 4.7).



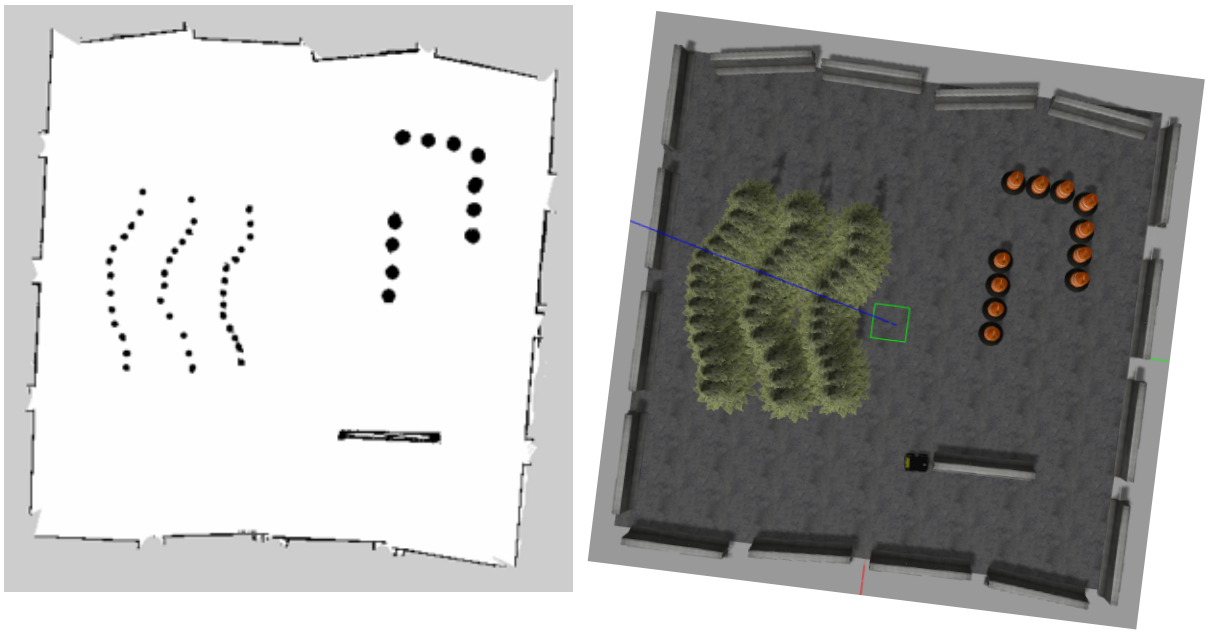
Şekil 4.7. Farklı parçacık sayısı değerlerine göre harita oluşturmada elde edilen entropi değerlerinin dağılımı

Haritalama yazılımı için uygun güncelleme ve parçacık sayısı değerleri seçildikten sonra haritalama işlemi tasarlanan otonom aracın simülasyon ortamında dolaştırılmasıyla gerçekleştirilmiştir. Harita çözünürlüğü kullanılan LIDAR sensör ile uyumlu olarak 2 cm olarak seçilmiş. Bir hücrede engel olduğuna karar verilmesi için dış ortam haritalamasında uygun değerler olan 0,65 eşik değeri ve hücrenin boş olduğuna karar verilmesi için ise 0,196 eşik değeri kullanılmıştır (Joseph 2018).

Şekil 4.8' de otonom araç simülasyon ortamında ilerledikçe oluşturulan harita görülmektedir. Haritalama için serbest sürüş yaparken sabit hızda gidilmesi ve dönüşlerin mümkün olduğunca yavaş yapılması entropiyi düşürecek ve daha kısa sürede harita oluşturulmasını sağlamıştır. Haritalanan ve üzerinde engel olmayan yerler beyaz, engeller siyah ve daha ölçüm yapılmamış belirsiz yerler ise gri ile gösterilmiştir.



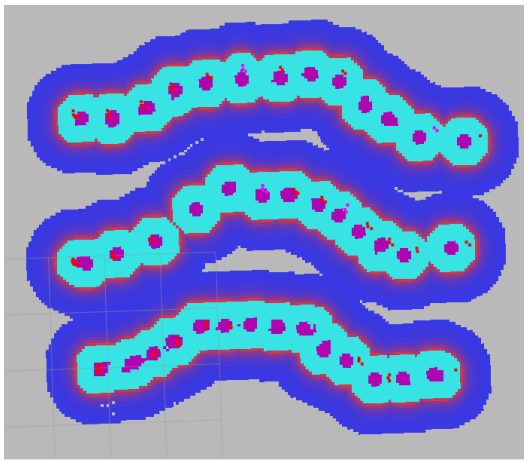
Şekil 4.8. Simülasyon ortamının haritalama aşamaları



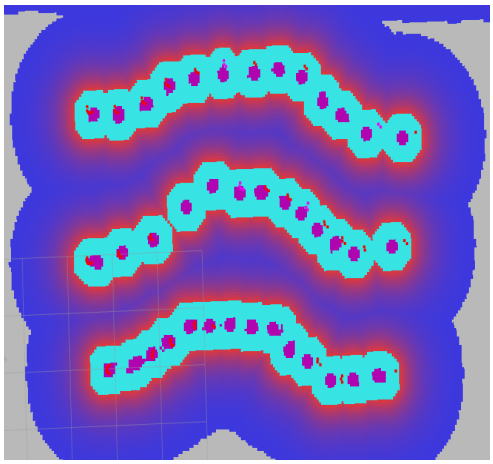
Şekil 4.9. Simülasyon ortamının yerleşimi ve yazılım tarafından çıkarılan haritası

Şekil 4.9' da üretilen haritanın son hali ve simülasyon ortamının gerçek görünümü görülmektedir. Üretilen harita üzerinde problemli noktalar (yanlış ölçüm veya ölçülmemiş alanlar) Microsoft Paint kullanılarak düzenlenmiş ve hatalı yerler düzeltilmiştir. Bunun diğer bir avantajı otonom aracın gitmesi istenmeyen yerlerin siyah bir çizgi ile çevrilerek rota hesaplaması dışarısında bırakılabilmesidir.

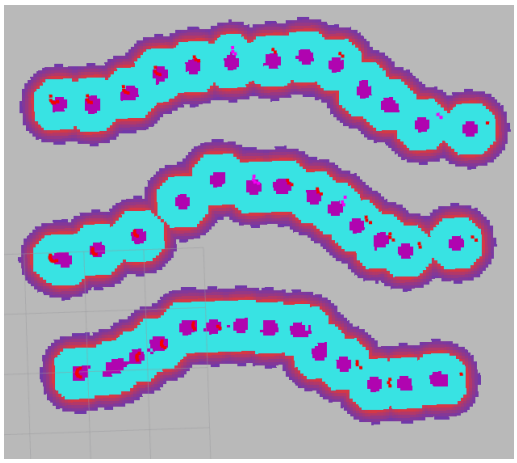
Oluşturulan haritada tolerans katmanı için otonom araç izdüşüm dış teğet çemberinin yarıçapına göre farklı katsayılar denenmiştir. Çok küçük katsayılar için tolerans katmanı düşük değer almakta bu da rota planlama sonucunda bir engelle çarpışmaya neden olabilmektedir. Çok büyük değerler için ise otonom araç aslında girebileceği koridorlara girememektedir. Simülasyon ortamında yapılan sürüş denemeleri sonucunda tolerans katmanı dış teğet çemberin 1.5 katı olacak şekilde seçilmiştir (Şekil 4.10).



Dış teğet çemberin yarıçapının 1.5 katı



Dış teğet çemberin yarıçapının 3 katı



Dış teğet çemberin yarıçapına eşit

Şekil 4.10. Farklı dış teğet çember değerlerine göre tolerans katmanının değişimi

Üretilen harita ROS tarafından sağlanan harita sunucusu ile lokalizasyon ve rota planlama modülleri tarafından kullanılmıştır.

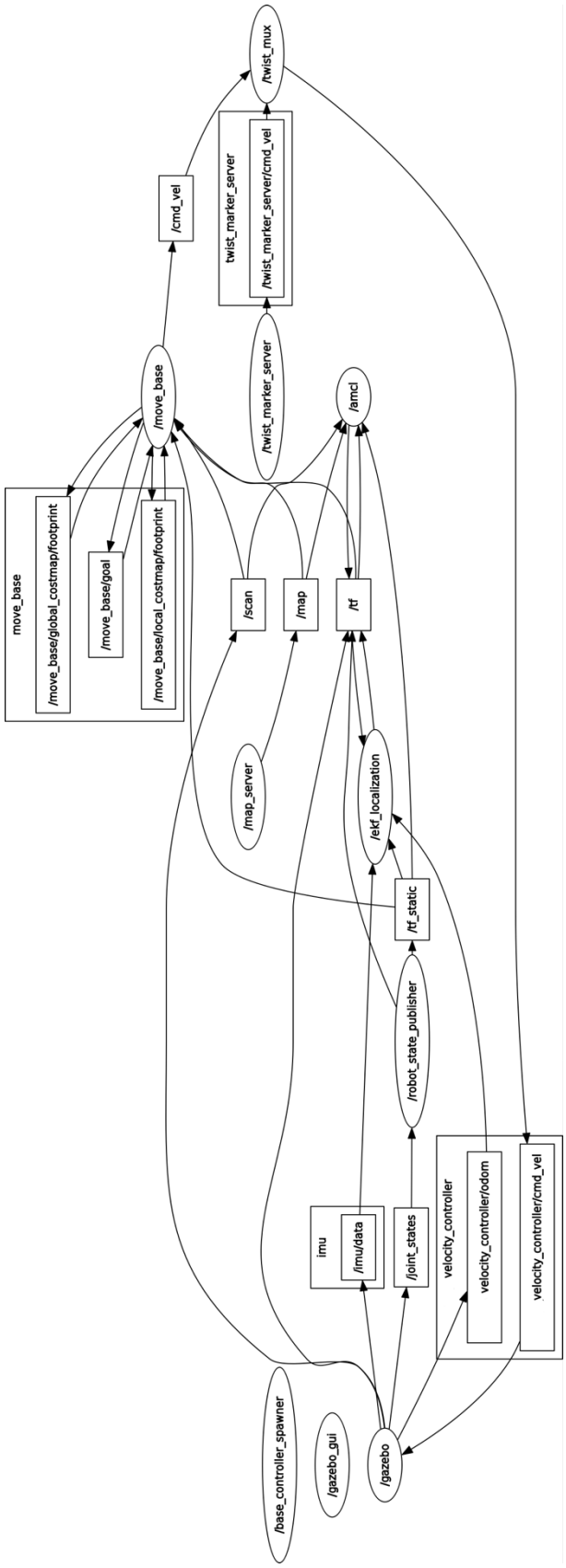
4.2. Lokalizasyon

Otonom navigasyon sağlayabilmek için engel haritası tek başına yeterli değildir. Navigasyonda ikinci adım olarak otonom aracın harita üzerinde hangi konumda olduğunu bilmesi gereklidir. Bunun için LIDAR, atalet ve odometri sensörlerinden aldığı verileri işleyen, bu ölçümleri harita ile karşılaştırarak otonom aracın harita üzerindeki konumunu tahmin eden bir lokalizasyon uygulaması geliştirilmiştir. Lokalizasyon uygulamasında adaptif monte karlo lokalizasyon algoritması kullanılmıştır.

Şekil 4.11'de lokalizasyon sistemini oluşturan yazılım kısımları görülmektedir. Otonom araç ve bu araca ait tüm sensör verileri simülatör (/gazebo) ortamında simüle edilmiştir. Harita sunucusu tarafından sağlanan harita (/map) LIDAR sensörden alınan veriler (/scan), odometri (atalet ve devir ölçer) sensörlerinden alınarak birleştirilen veriler (/ekf_localization) ve dönüş merkezine göre sensörlerden alınan verilere uygulanacak geometrik düzeltme (/tf) girdi olarak işlenmiş, buradan elde edilen sonuca göre otonom aracın harita üzerindeki pozisyonu belirlenmiştir. Otonom aracın alan üzerinde ilerlemesi rota planlayıcı tarafından yapılan kontrol ile (/cmd_vel) sağlanmış ve yeni konum tekrardan simüle edilmiş otonom araca iletilerek (/gazebo) bu döngü devam ettirilmiştir.

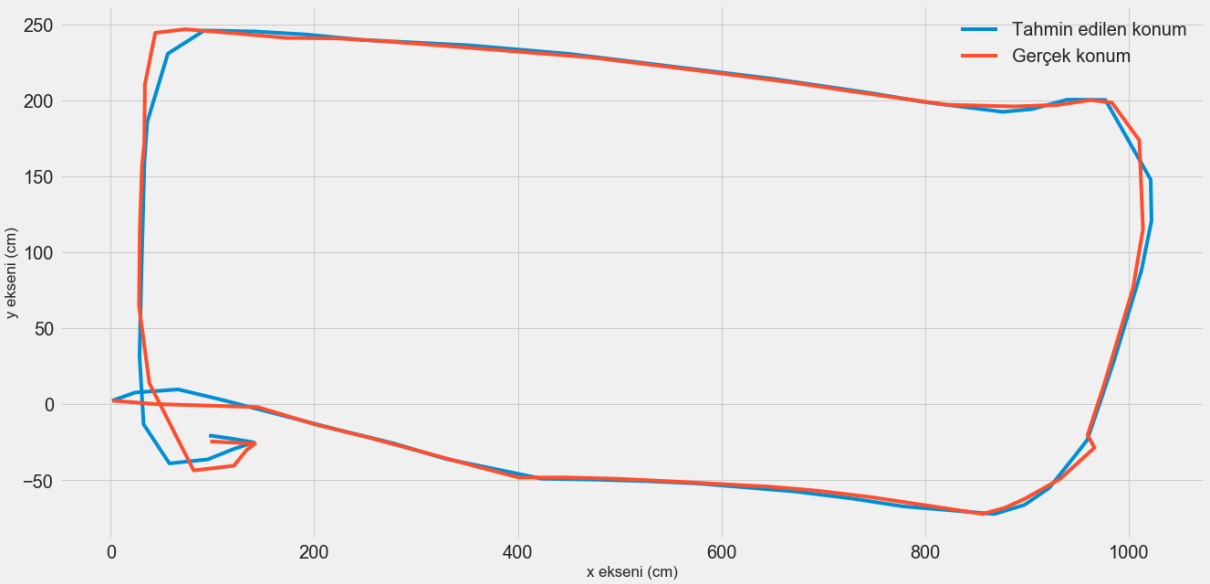
Lokalizasyon uygulamasının başarısı tahmin edilen konum ile simülasyon ortamında otonom aracın sahip olduğu gerçek konum arasındaki fark alınarak ölçülmüştür. Ölçülen bu konumlama hatası tahmin edilen konumun ne kadar harekette güncelleneceği ile lokalizasyon uygulamasında kullanılan parçacık sayısına bağlıdır. Farklı değerler için lokalizasyon başarısı karşılaştırılmıştır.

Dış ortamda yapılan adaptif monte karlo lokalizasyon uygulamalarında ortamda ayırt edici engel yapısına bağlı olarak en az 50-200 parçacık çiftinin kullanılması önerilmiştir (Quigley ve ark. 2016). Lokalizasyonda (5,20), (50,200), (500,2000) ve (5000,20000) minimum maksimum parçacık çiftleri kullanılmıştır. Parçacık sayısının arttırılması bilgisayar işlem gereksinimini arttırırken çok az seçilmesi ise doğru sonuçlar elde edilmesini önlemektedir. Farklı parçacık sayısı değerlerinin konumlama hatası üzerindeki etkisi araştırılırken konum güncelleme değeri 50 Hz çözünürlüğe sahip bir LIDAR sensör ve 1 m/s sabit ilerleme hızı kullanıldığı için 2 cm olarak kullanılmıştır.

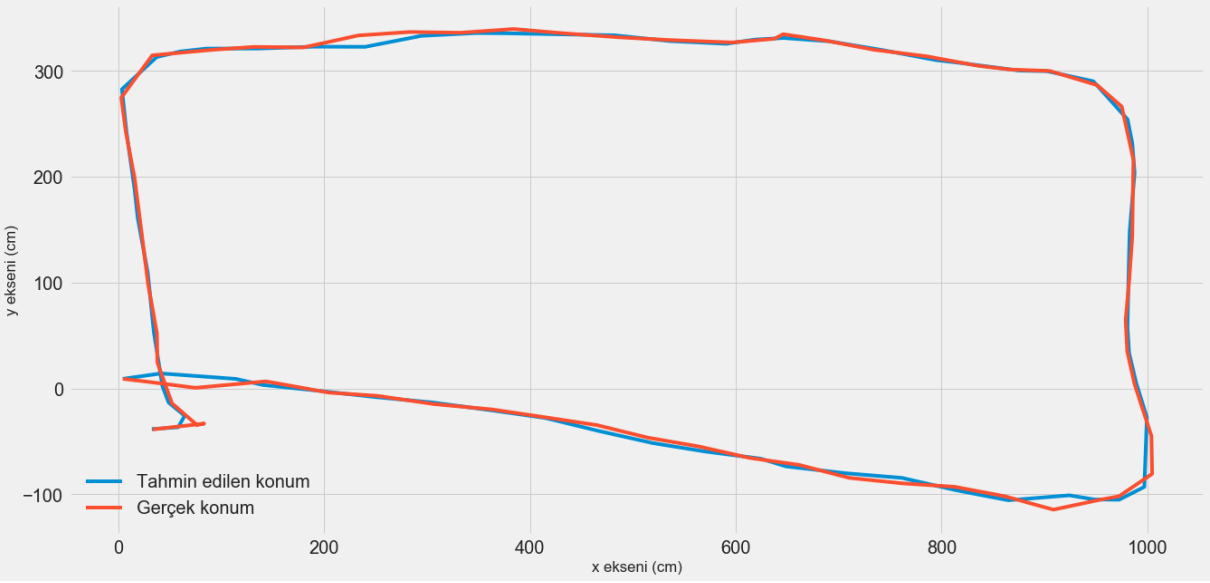


Şekil 4.11. Geliştirilen Adaptif Monte Karlo Lokalizasyon Uygulamasına ait Diyagram

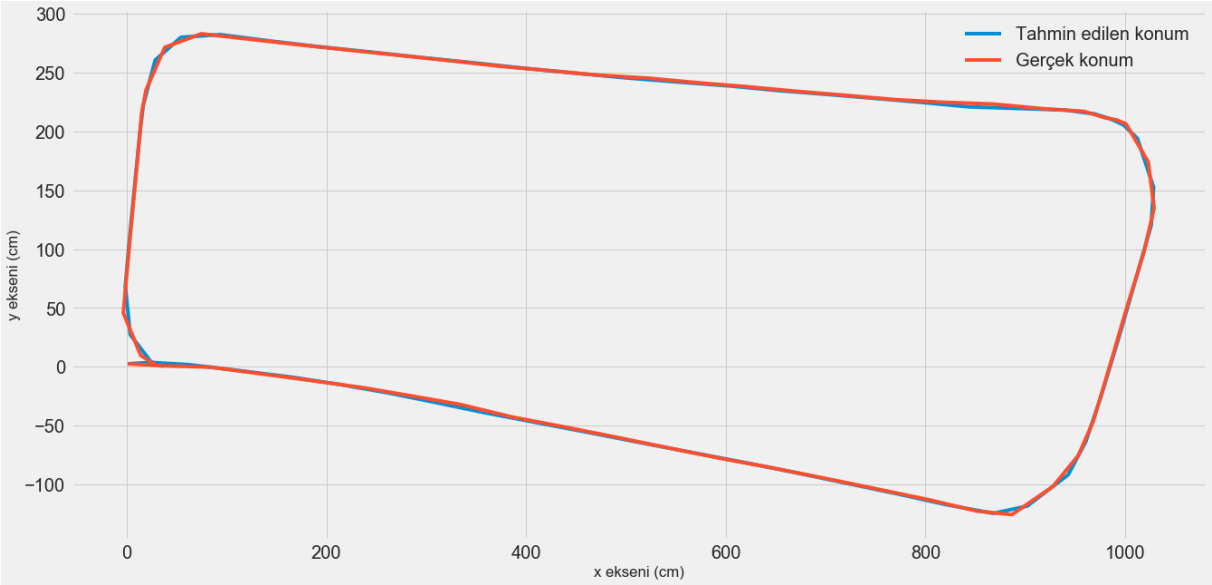
Otonom araç iki komşu sıra içerisindeki hareket ettirilerek farklı sayıdaki minimum-maksimum parçacık sayısı çiftleri, parçacık sayısının gerçek konum ve tahmin edilen konum arasında oluşturduğu fark ölçülmüştür.



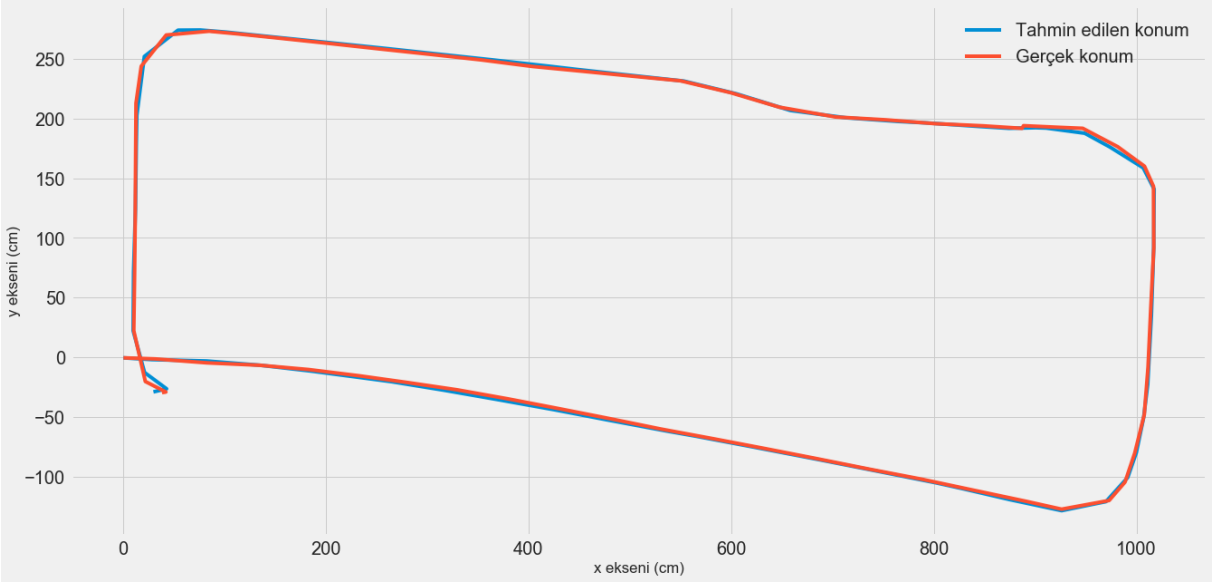
Şekil 4.12. Lokalizasyonda minimum 5, maksimum 20 parçacık kullanımında gerçek konum ve tahmin edilen konum



Şekil 4.13. Lokalizasyonda minimum 50, maksimum 200 parçacık kullanımında gerçek konum ve tahmin edilen konum



Şekil 4.14. Lokalizasyonda minimum 500, maksimum 2000 parçacık kullanımında gerçek konum ve tahmin edilen konum



Şekil 4.15. Lokalizasyonda minimum 5000, maksimum 20000 parçacık kullanımında gerçek konum ve tahmin edilen konum

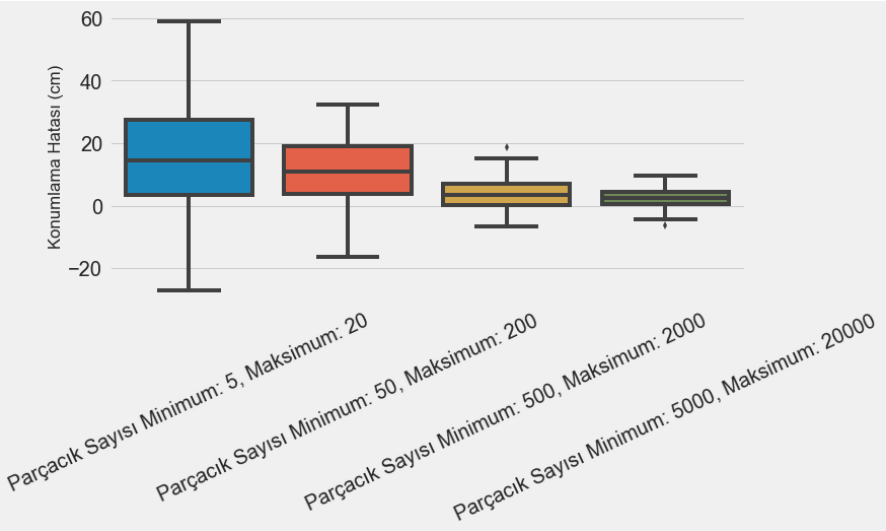
Çizelge 4.3. Lokalizasyonda farklı parçacık sayısı kullanımına bağlı olarak tahmin edilen konum ve gerçek konum arasındaki fark

	Konumlama Hatası (cm) Min:5 - Maks: 20 Parçacık	**Konumlama Hatası (cm) Min: 50 - Maks: 200 Parçacık**	**Konumlama Hatası (cm) Min: 500 - Maks: 2000 Parçacık**	**Konumlama Hatası (cm) Min: 5000 Maks: 20000 Parçacık**
Ortalama	15,3	10,2	3,5	2,3
Standard Sapma	17,7	10,5	4,9	3,0
Minimum Değer	0,0	0,0	0,0	0,0
Ortanca Değer	14,4	10,8	3,5	2,5
Maksimum Değer	59,1	32,5	18,8	9,7

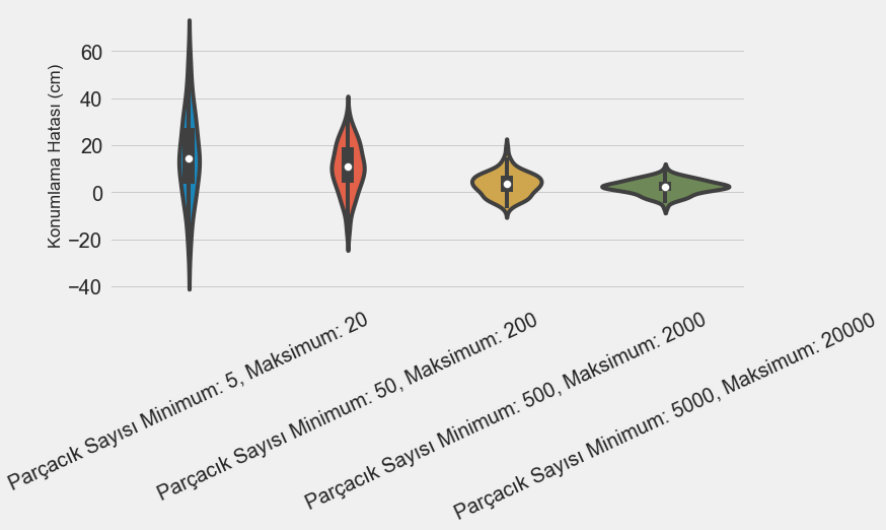
Geliştirilen lokalizasyon uygulamasında otonom araç ilerleyip ölçüm aldıkça ve bu ölçümleri harita ile karşılaştırdıkça tahmin edilen konum ile gerçek konum birbirine yaklaşmıştır. Kullanılan algoritmanın adaptif yapısı nedeniyle başlangıç durumunda daha fazla parçacık kullanılırken ot pozisyonu konusundaki tahminin kesinlik değeri arttıkça (parçacıklar birbirine yakınlaştıkça) kullanılan parçacık sayısı yazılım tarafından otomatik olarak azaltılarak bilgisayar işlem gücüne olan ihtiyaç düşürülmüştür.

Şekil 4.12, Şekil 4.13, Şekil 4.14, Şekil 4.15' te parçacık sayısı azaldıkça gerçek konum ile tahmin edilen konum arasındaki konum farkının arttığı gözlenmiştir.

Kullanılan dört farklı senaryo için gerçek ve tahmin edilen konum arasındaki en düşük ortalama fark 2,3 cm ile parçacık sayısının minimum: 5000 ve maksimum 20000 olduğu konfigürasyonda alınmıştır. En yüksek ortalama konumlama hatası ise 15,3 cm ile minimum parçacık sayısının 5, maksimum ise 20 olarak kullanıldığı senaryoda elde edilmiştir (Çizelge 4.3).



Şekil 4.16. Lokalizasyonda kullanılan farklı parçacık sayılarına göre tahmin edilen konum ve gerçek konum arasındaki fark dağılımını gösteren kutu grafik

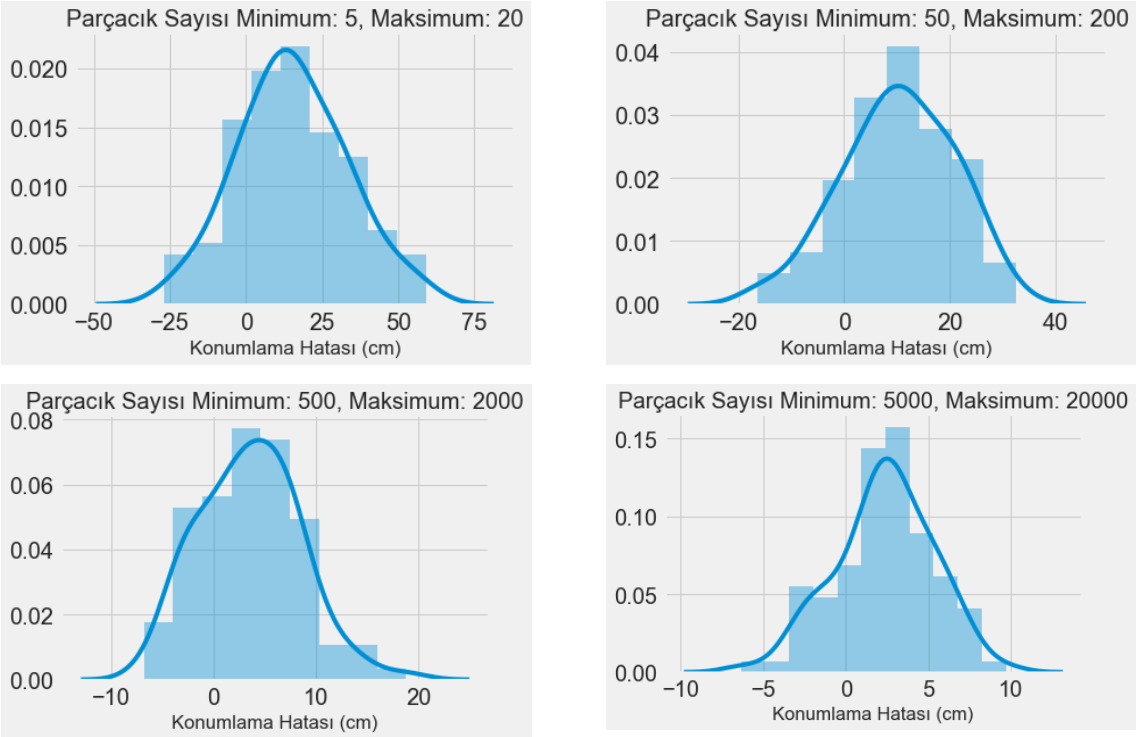


Şekil 4.17. Lokalizasyonda kullanılan farklı parçacık sayılarına göre tahmin edilen konum ve gerçek konum arasındaki fark dağılımını gösteren viyolin grafik

Yukarıdaki kutu ve viyolin grafik şekillerinde dört farklı konfigürasyon için gerçek konum ile tahmin edilen konum arasındaki fark değerlerinin dağılımı görülebilir (Şekil 4.16, Şekil 4.17). Kutu grafik, değerlerin dağılımını görmek için uygundur ancak her bir değerin maksimum, minimum ve ortanca nokta arasındaki örnek dağılımını görmek için viyolin grafik kullanılmıştır.

En yüksek varyansın minimum 5 maksimum ise 20 parçacık kullanılan konfigürasyonda olduğu kutu ve viyolin grafiklerde görülmektedir. Örnekleme dağılımının minimum 50-

maksimum 200 parçacık kullanılan konfigürasyonda ortanca değer olan 10,8 cm etrafında, minimum 500-maksimum 2000 parçacık kullanılan konfigürasyonda ortanca değer olan 3,5 cm etrafında, minimum 5000-maksimum 20000 parçacık kullanılan konfigürasyonda ortanca değer olan 2,5 cm etrafında ve minimum 5-maksimum 20 parçacık kullanılan konfigürasyonda ise varyansı daha yüksek bir şekilde maksimum ve minimum noktalar arasında dağıldığı görülmektedir.



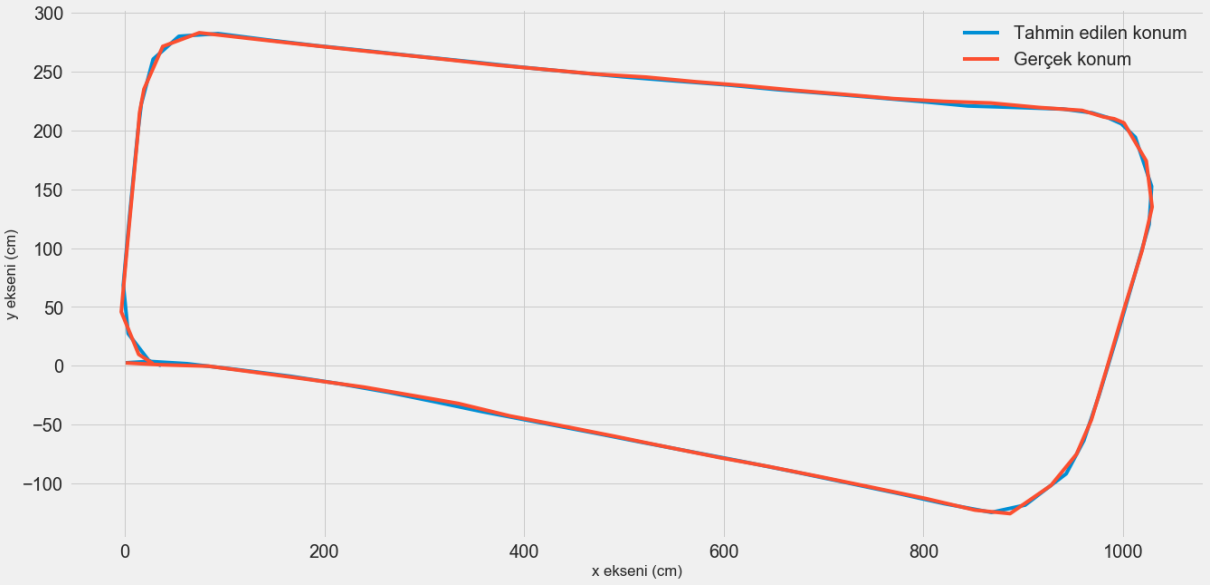
Şekil 4.18. Lokalizasyonda kullanılan farklı parçacık sayılarına göre tahmin edilen konum ve gerçek konum arasındaki fark dağılımı

Üç farklı kongifürasyon için gerçek ve tahmin edilen konum arasındaki fark değerlerinin dağılımları incelendiğinde normal dağılıma sahip oldukları görülmektedir (Şekil 4.18).

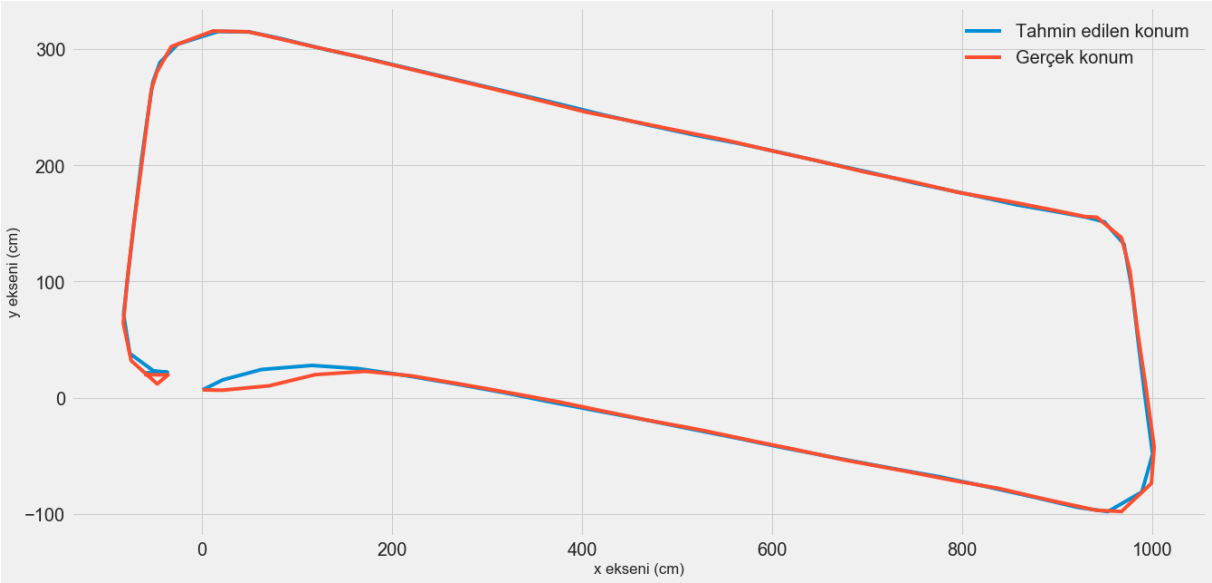
Geliştirilen lokalizasyon uygulamasında otonom araç ilerleyip ölçüm aldıkça ve bu ölçümleri harita ile karşılaştırdıkça tahmin edilen konum ile gerçek konum birbirine yaklaşmıştır. Kullanılan parçacık sayısının dışında konum doğruluğunu etkileyen diğer bir faktör hangi sıklıkla konum güncellemesi yapılacağıdır. Otonom aracın iki komşu sıra içerisindeki hareketi farklı sayıdaki konum güncelleme değerleri için gerçek konum ve tahmin edilen konum arasında oluşturduğu farkı ölçmek için kullanılmıştır.

Konum güncellenmesi için yapılması gereken yer değiştirme miktarı kullanılan sensör ve ilerleme hızına uygun olacak şekilde seçilmiştir. Güncelleme değerlerinin küçültülmesi bilgisayar işlem gereksinimini arttırırken çok büyük seçilmesi ise doğru sonuçlar elde edilmesini önlemektedir.

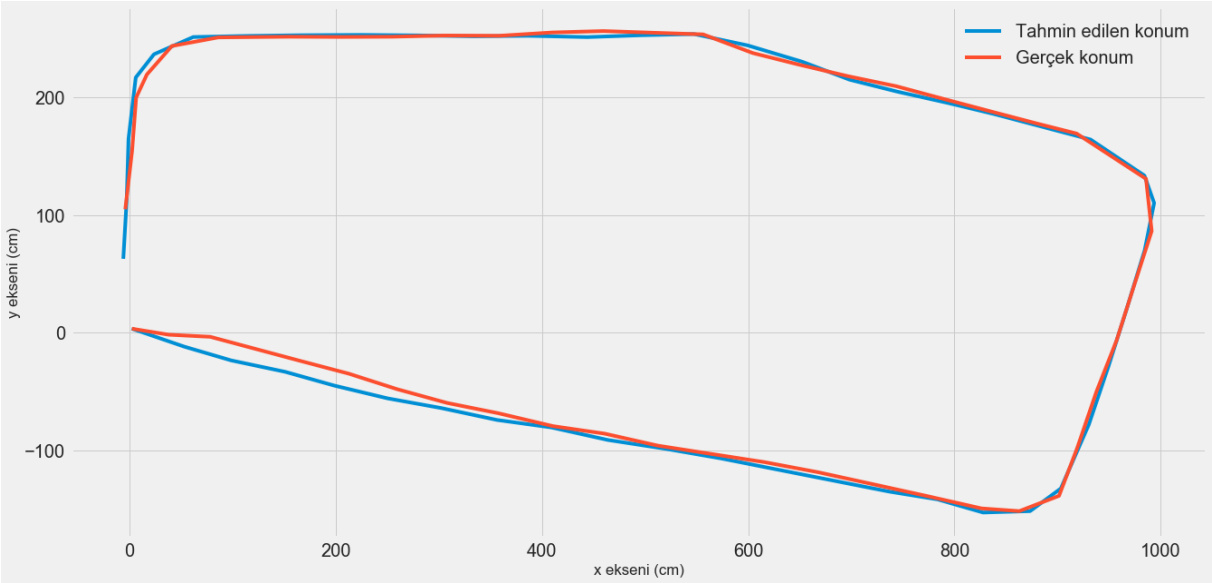
Şekil 4.19, Şekil 4.20 ve Şekil 4.21'de güncelleme oranı düştükçe gerçek ve tahmin edilen konum arasındaki farkın arttığı gözlenmiştir. İki komşu sıra içerisindeki hareket kullanılarak güncelleme değeri gerçek konum ve tahmin edilen konum arasındaki farkın karşılaştırılması yapılmıştır. Bu amaçla kullanılan LIDAR sensör 50 Hz ölçüm frekansına sahip olduğu ve ilerleme hızı 1 m/s sabit hız seçildiği için 2 cm, 4 cm ve 8 cm harekette konum (Çelen ve ark. 2008) güncellemesi yapılacak 3 farklı konfigürasyon tercih edilmiştir.



Şekil 4.19. Lokalizasyonda 2 cm yer değiştirmede yapılan konum güncellemesinde gerçek konum ve tahmin edilen konum



Şekil 4.20. Lokalizasyonda 4 cm yer değiştirmede yapılan konum güncellemesinde gerçek konum ve tahmin edilen konum

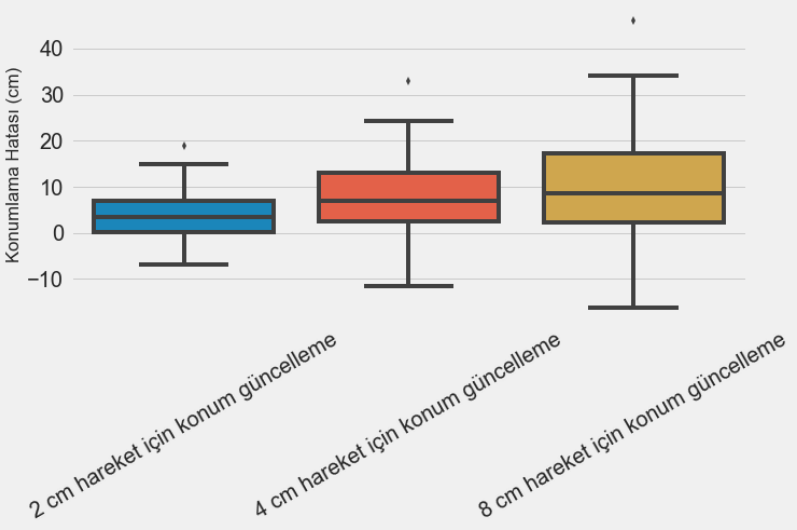


Şekil 4.21. Lokalizasyonda 8 cm yer değiştirmede yapılan konum güncellemesinde gerçek konum ve tahmin edilen konum

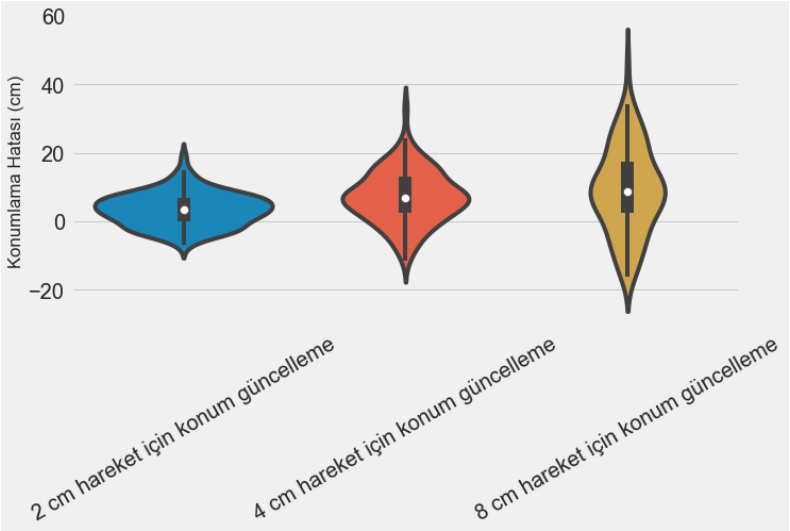
Çizelge 4.4. Lokalizasyonda farklı güncelleme değerleri kullanımına bağlı olarak tahmin edilen konum ve gerçek konum arasındaki fark

	Konumsal Hata (cm) 2 cm hareket için konum güncelleme	**Konumsal Hata (cm) 4 cm hareket için konum güncelleme**	**Konumsal Hata (cm) 8 cm hareket için konum güncelleme**
Ortalama	3,5	7,2	9,2
Standard Sapma	4,9	7,7	12,7
Minimum Değer	0,0	0,0	0,0
Ortanca Değer	3,5	7,0	9,0
Maksimum Değer	18,8	33,0	46,0

Kullanılan üç farklı konfigürasyon için gerçek ve tahmin edilen konum arasındaki en düşük ortalama fark 3,5 cm ile konum güncellemesinin 2 cm'lik yer değişiminde yapıldığı konfigürasyonda alınmıştır. En yüksek ortalama fark ise 9,2 cm ile konum güncellemesinin doğrusal hareket için 8 cm'lik yer değişiminde yapıldığı konfigürasyonda elde edilmiştir (Çizelge 4.4).



Şekil 4.22. Lokalizasyonda farklı güncelleme değerleri kullanımına bağlı olarak tahmin edilen konum ve gerçek konum arasındaki fark dağılımını gösteren kutu grafik

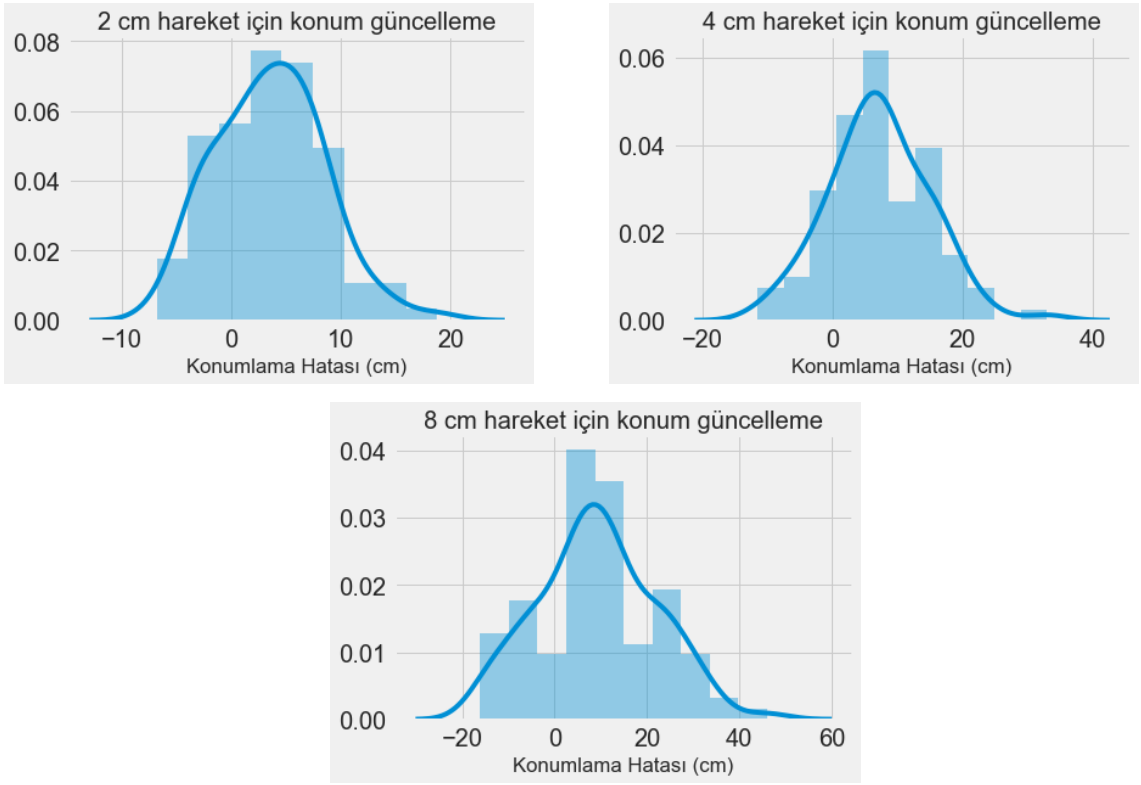


Şekil 4.23. Lokalizasyonda farklı güncelleme değerleri kullanımına bağlı olarak tahmin edilen konum ve gerçek konum arasındaki fark dağılımını gösteren viyolin grafik

Şekil 4.22 ve Şekil 4.23'te üç farklı konfigürasyon için gerçek konum ile tahmin edilen konum arasındaki fark değerlerinin dağılımı görülebilir. Kutu grafik, değerlerin dağılımını görmek için uygundur ancak her bir değerin maksimum, minimum ve ortanca nokta arasındaki örnek dağılımını görmek için viyolin grafik kullanılmıştır.

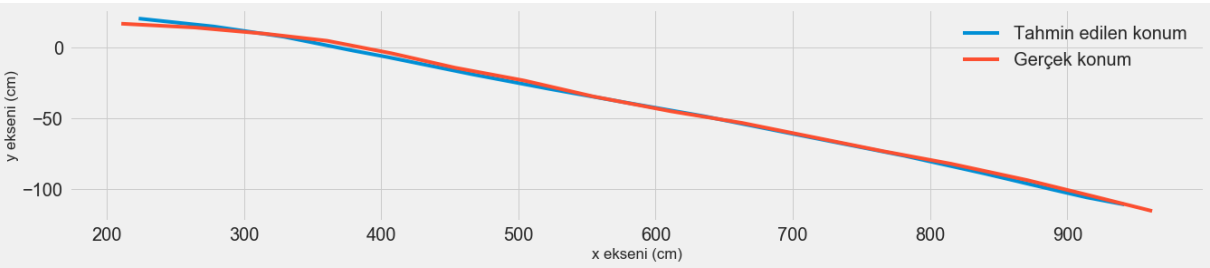
En yüksek varyansın konum güncellemesinin 8 cm'lik yer değiştirme için yapıldığı konfigürasyonda olduğu kutu ve viyolin grafiklerde görülmektedir. Örnek dağılımının konum 2 cm'lik yer değişiminde yapıldığı konfigürasyonda ortanca değer olan 3,5 cm etrafında, konum güncellemesinin 4 cm'lik yer değişiminde yapıldığı konfigürasyonda ortanca değer olan 7 cm etrafında ve konum güncellemesinin 8 cm'lik yer değişiminde yapıldığı konfigürasyonda ise varyansı daha yüksek bir şekilde maksimum ve minimum noktalar arasında dağıldığı görülmüştür.

Üç farklı kongifürasyon için gerçek ve tahmin edilen konum arasındaki fark değerlerinin dağılımları incelendiğinde normal dağılıma sahip oldukları görülmektedir (Şekil 4.24).

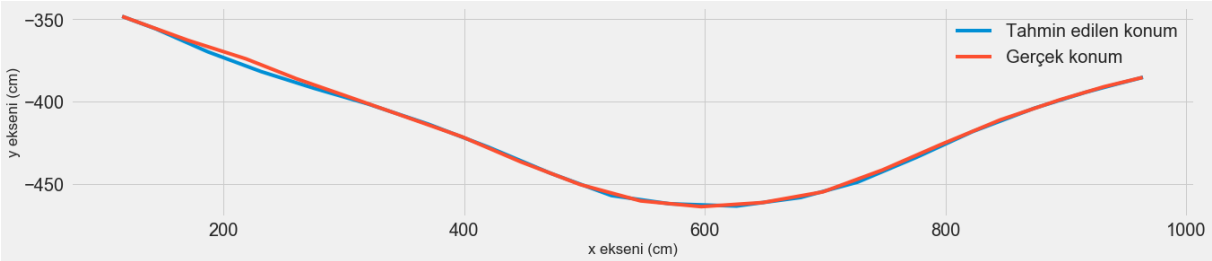


Şekil 4.24. Lokalizasyonda farklı güncelleme değerleri kullanımına bağlı olarak tahmin edilen konum ve gerçek konum arasındaki fark dağılımı

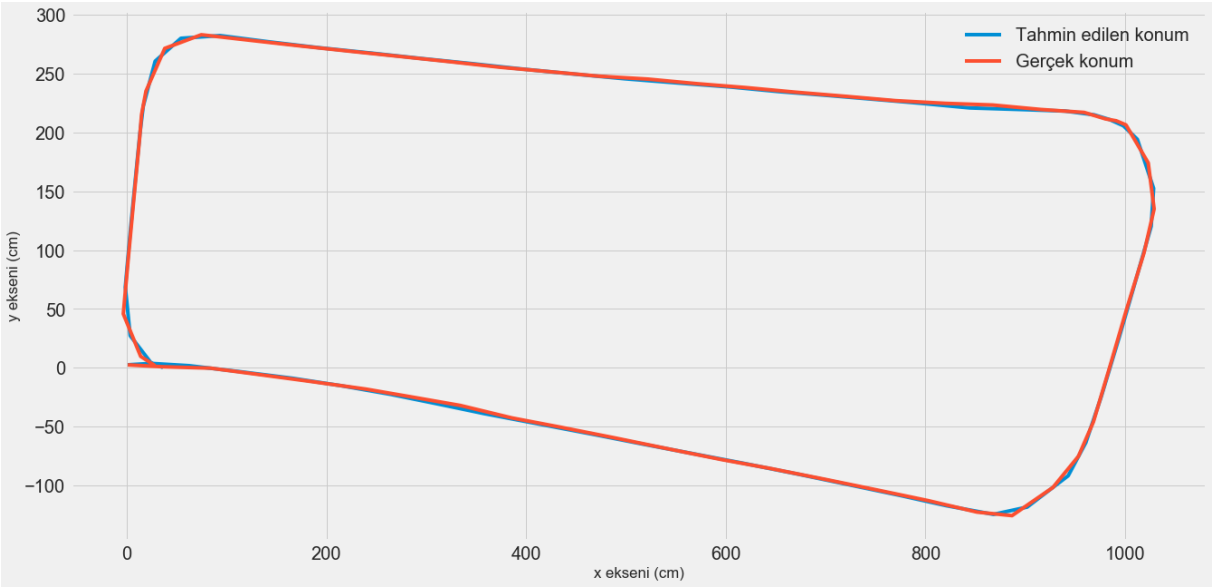
Uygun konum güncelleme ve parçacık sayısı değerleri bulunduktan sonra oluşturulan lokalizasyon uygulaması böyle bir otonom aracın karşılaşacağı düz bir sıra üzerinde hareket, dönemeçli bir sıra üzerinde hareket ve iki komşu sıra içerisinde yapılan hareket olmak üzere üç farklı senaryo için simüle edilmiştir. Otonom aracın simülasyon ortamı üzerindeki gerçek konumu ve otonom araç tarafından tahmin edilen konumu karşılaştırılarak lokalizasyon uygulamasının başarısı saptanmıştır. Lokalizasyonun belirli bir oranda sağlanması için fark ölçümleri alınmaya, otonom araç ortamda 50 cm ilerledikten sonra başlanmıştır.



Şekil 4.25. Düz sıra üzerinde tahmin edilen ve gerçek konum



Şekil 4.26. Dönemeçli sıra üzerinde tahmin edilen konum ve gerçek konum

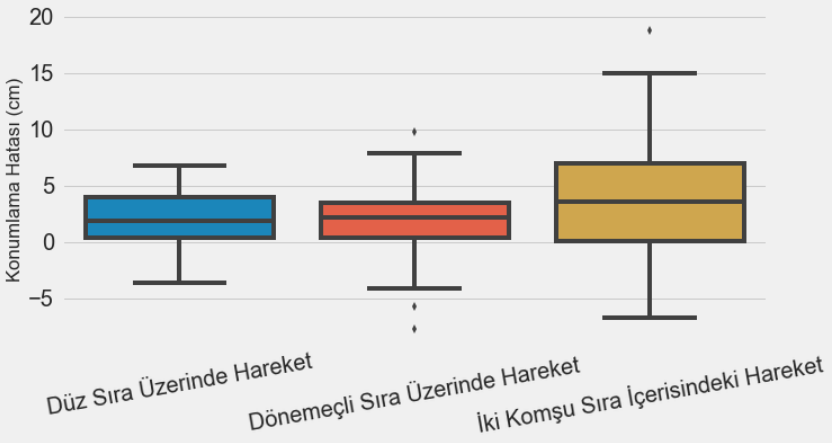


Şekil 4.27. İki komşu sıra içerisindeki harekette tahmin edilen konum ve gerçek konum

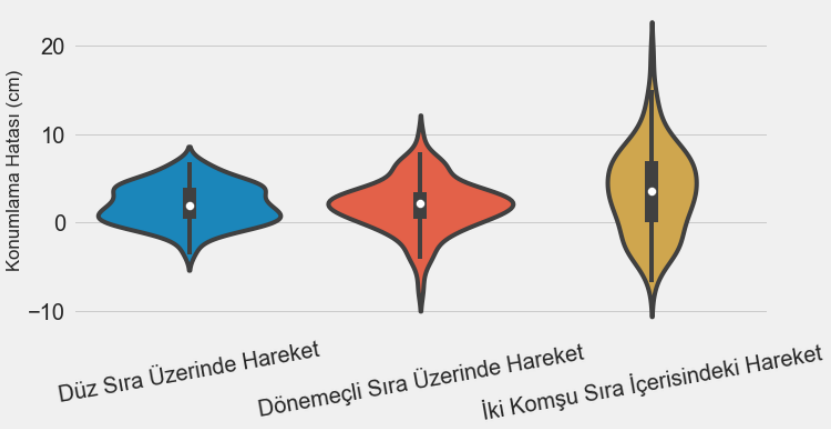
Çizelge 4.5. Farklı hareket konfigürasyonlarında tahmin edilen konum ve gerçek konum arasındaki fark değerleri

	Konumlama Hatası (cm) Düz Sıra Üzerinde Hareket	**Konumlama Hatası (cm) Dönemeçli Sıra Üzerinde Hareket**	**Konumlama Hatası (cm) İki Komşu Sıra İçerisindek Hareket**
Ortalama	2,1	2,2	3,5
Standard Sapma	2,3	2,9	4,9
Minimum Değer	0,0	0,0	0,0
Ortanca Değer	2,0	2,1	3,5
Maksimum Değer	6,8	9,8	18,8

Geliştirilen lokalizasyon uygulamasında otonom araç ilerleyip ölçüm aldıkça ve bu ölçümleri harita ile karşılaştırdıkça tahmin edilen konum ile gerçek konum birbirine yaklaşmıştır (Şekil 4.25, Şekil 4.26, Şekil 4.27). Ancak diferansiyel sürüşe sahip otonom araç, doğası gereği dönüş yaptıkça tekerleklerin farklı devirlerde dönmesi nedeniyle patinaja sebep olmakta bu da belirsizliği arttırmaktadır. Şekil 4.15'e bakıldığında dönüşler sırasında belirsizliğin arttığı ancak düz gidildikçe belirsizliğin azaldığı görülmüştür. Bu üç farklı senaryoda gerçek ve tahmin edilen konum arasındaki en düşük ortalama konumlama hatası 2.1 cm ile düz bir sıra üzerinde gidilen konfigürasyonda alınmıştır. En yüksek ortalama konumlama hatası ise ise 3,5 cm ile iki komşu sıra içerisinde hareket edilen senaryoda elde edilmiştir (Çizelge 4.5).



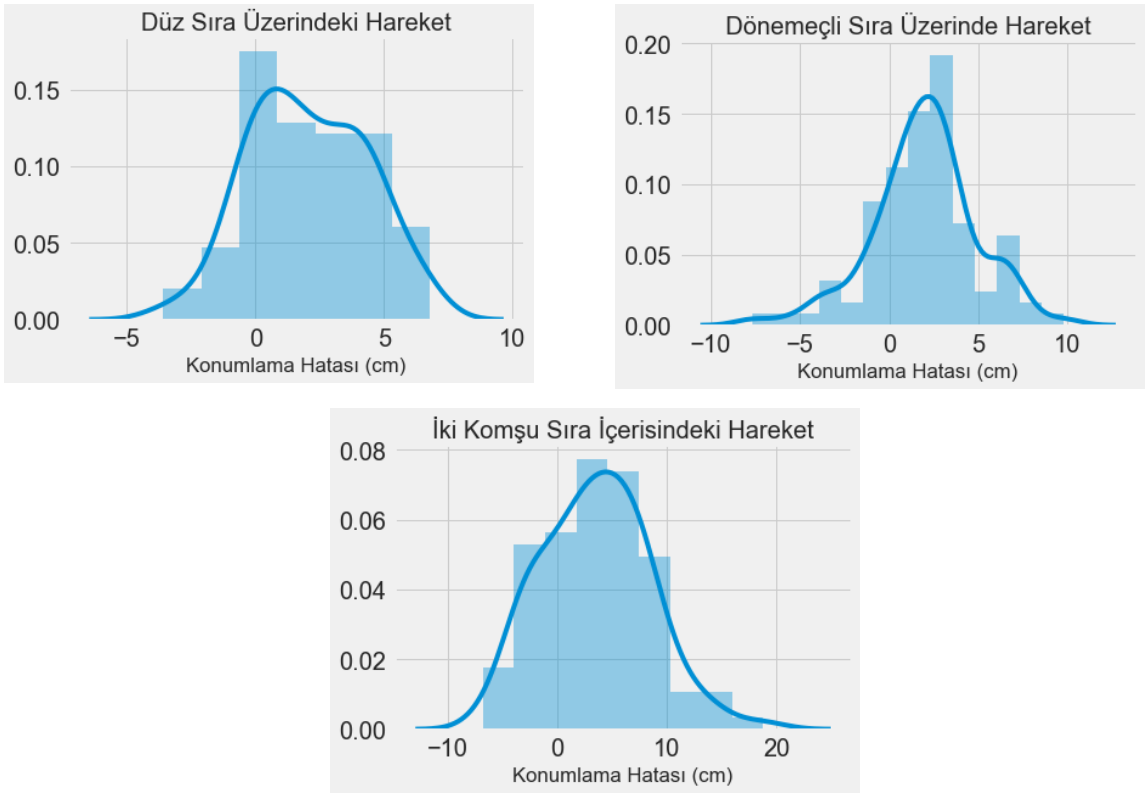
Şekil 4.28. Farklı hareket konfigürasyonlarında tahmin edilen konum ve gerçek konum arasındaki fark dağılımını gösteren kutu grafik



Şekil 4.29. Farklı hareket konfigürasyonlarında tahmin edilen konum ve gerçek konum arasındaki fark dağılımını gösteren viyolin grafik

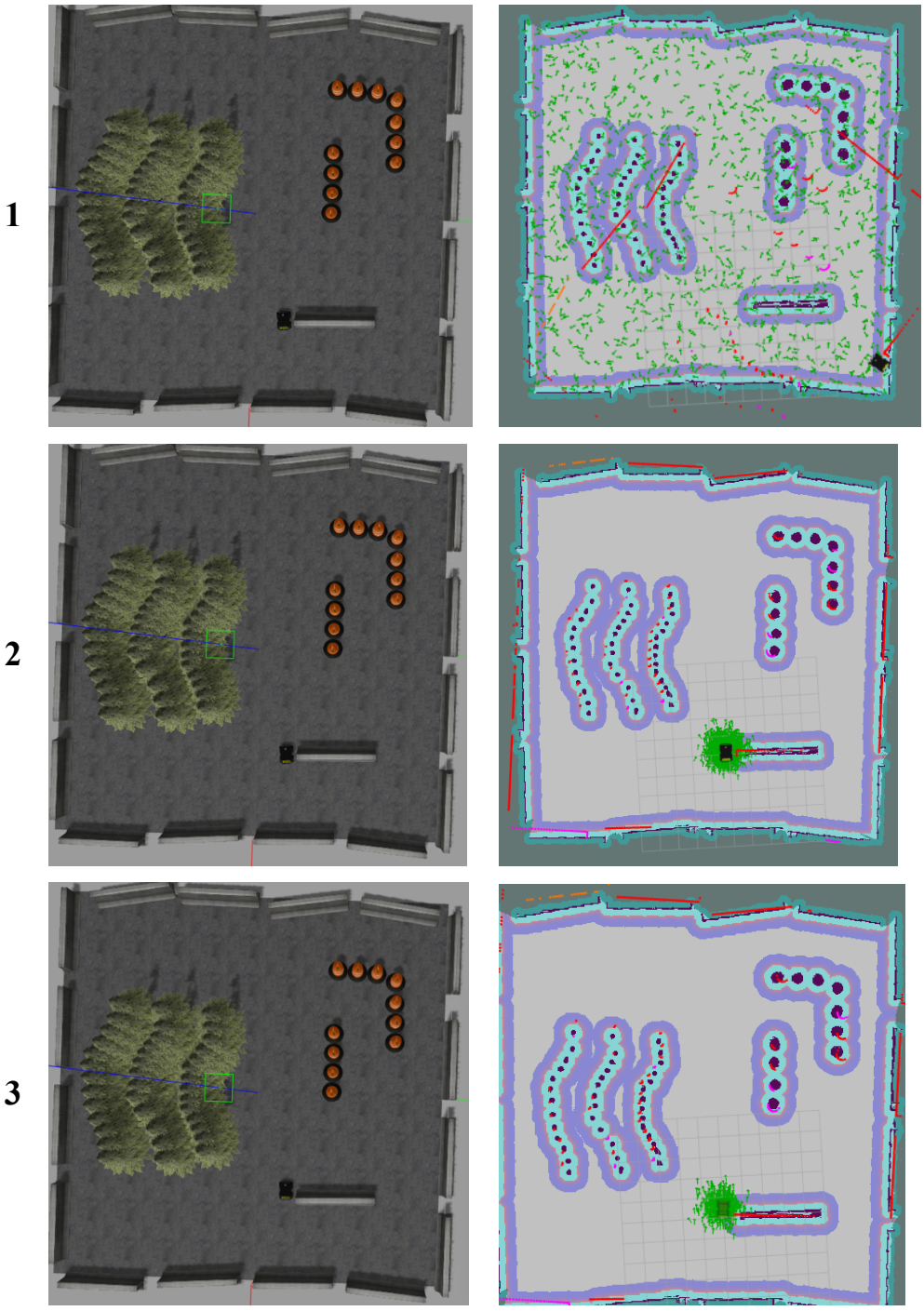
Yukarıdaki kutu ve viyolin grafik şekillerinde üç farklı konfigürasyon için gerçek konum ile tahmin edilen konum arasındaki fark değerlerinin dağılımı görülmektedir (Şekil 4.28, Şekil 4.29). Kutu grafik, değerlerin dağılımını görmek için uygundur ancak her bir değerin maksimum, minimum ve ortanca nokta arasındaki örneklem dağılımını görmek için viyolin grafik kullanılmıştır.

En yüksek varyansın iki komşu sıra içerisinde hareket edilen konfigürasyonda olduğu kutu ve viyolin grafiklerde görülmektedir. Örnekleme dağılımının düz sıra üzerinde hareket edilen konfigürasyonda ortanca değer olan 2 cm etrafında, dönemeçli sıra üzerinde hareket edilen konfigürasyonda ortanca değer olan 2,1 cm etrafında, iki komşu sıra içerisinde hareket edilen üçüncü konfigürasyonda ise varyansı daha yüksek bir şekilde maksimum ve minimum noktalar arasında dağıldığı görülmektedir.

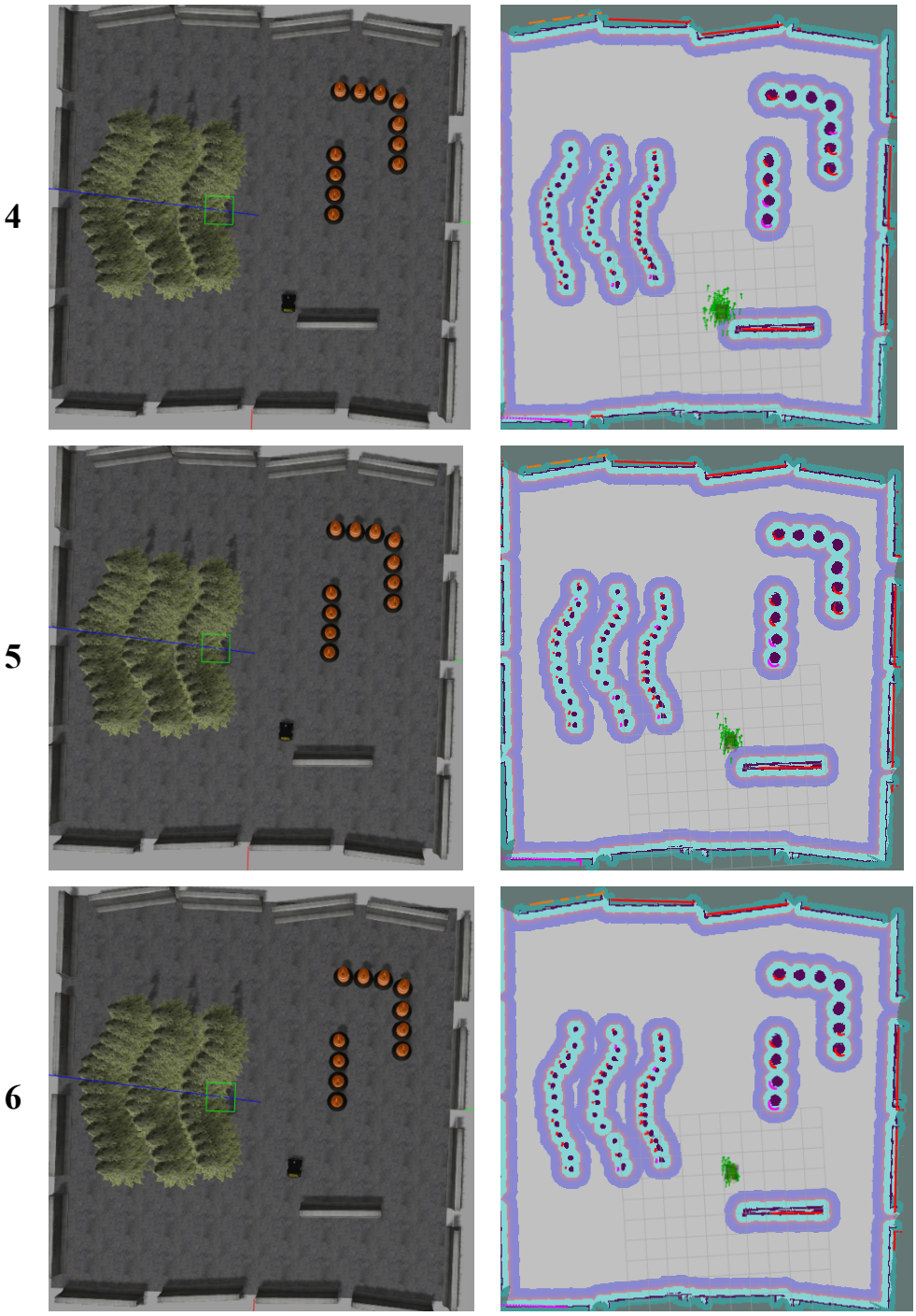


Şekil 4.30. Farklı hareket konfigürasyonlarında tahmin edilen konum ve gerçek konum arasındaki fark dağılımları

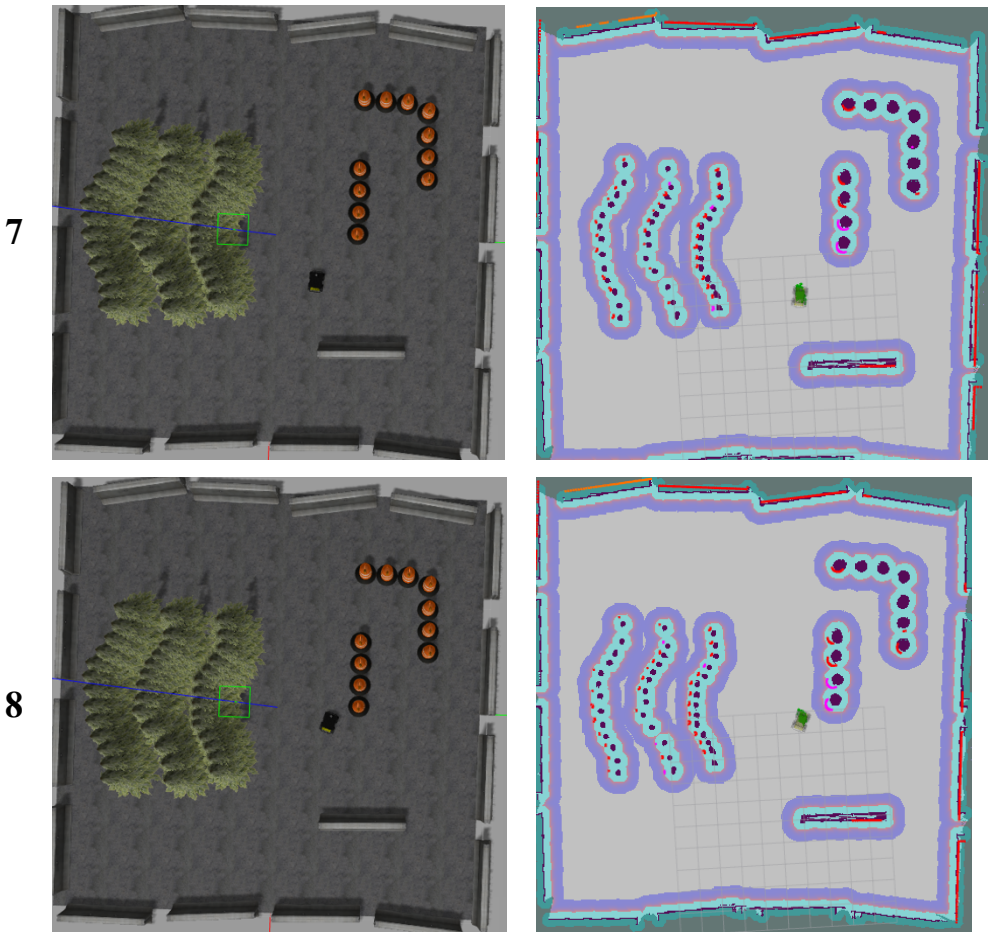
Üç farklı kongifürasyon için gerçek ve tahmin edilen konum arasındaki fark değerlerinin dağılımları incelendiğinde normal dağılıma sahip oldukları görülmüştür (Şekil 4.30).



Şekil 4.31. Lokalizasyon adımlarının görselleştirilmesi



Şekil 4.31. Lokalizasyon adımlarının görselleştirilmesi (devam)



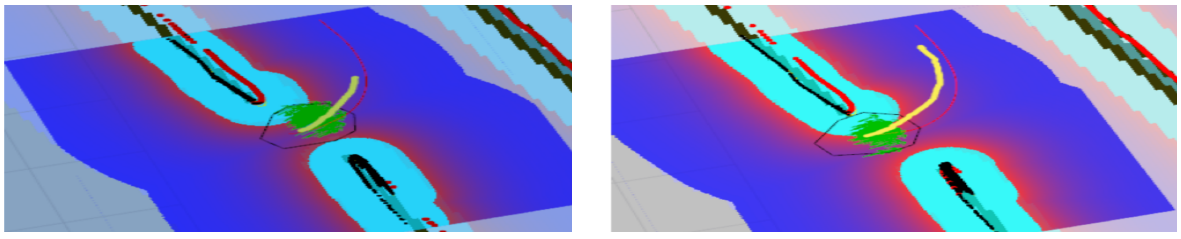
Şekil 4.31. Lokalizasyon adımlarının görselleştirilmesi (devam)

Şekil 4.31'de, sol taraftaki görüntüler otonom aracın simülasyon ortamındaki gerçek pozisyonunu (bizim gördüğümüz dünyayı) sağ taraftaki görüntüler ise otonom aracın yazılım tarafından tahmin edilen pozisyonunu (otonom aracın algıladığı dünyayı) göstermektedir. Sağ taraftaki görüntüde bulunan yeşil okların herbiri otonom araç tarafından olduğunu düşündüğü pozisyonları ifade eden parçacıklardır. Bir numaralı adımda otonom araç ölçüm almaya başlamış ancak hareket etmemiştir. Görüldüğü üzere otonom aracın pozisyonunu ifade eden parçacıklar (yeşil oklar) haritanın tamamına dağılmış durumdadır. Daha sonraki adımlarda ise otonom araç hareket edip ölçüm alarak haritayla karşılaştırdıkça parçacıkların otonom aracın etrafına toplandığı ve gerçek pozisyon ile tahmin edilen pozisyon arasındaki farkın azaldığı görülmüştür.

4.3. Rota Planlama

Otonom navigasyonda lokalizasyon aşamasından sonra gelen üçüncü adım ise rota planlamadır. Engel haritası ile içerisinde bulunulan ortamı tanıyan, lokalizasyon uygulaması ile bu ortam içerisindeki pozisyonunu bulan otonom araç için bir sonraki aşama bulunduğu konumdan istediği nokta/noktalara gitmesini sağlayacak rota planlama uygulamasıdır. Rota planlama uygulaması temel olarak otonom aracın bulunduğu konumu ve gitmek istediği konumu girdi olarak alarak bu iki nokta arasındaki hücre değerlerine göre en kısa rotayı hesaplamıştır. Rota planlama uygulaması global ve lokal planlayıcı olarak iki aşamalı tasarlanmıştır. Global planlayıcı adımında haritaya göre iki nokta arasındaki gidiş rotası hesaplanmaktadır. Lokal planlayıcı adımında ise global rota planı sensör ölçümleri sayesinde engellerden sakınarak takip edilmiştir.

Lokal rota planlamada kullanılan dinamik pencere yaklaşımı algoritmasında yol üzerindeki engellerden kaçınmak amacıyla rota simülasyon adımı bulunmaktadır. Bu adımda lokal planlayıcı otonom aracın kontrol uzayından aldığı hız örneklerini ve bu örnekler simülasyonda belirtilen simülasyon süresi boyunca uygulandıktan sonra elde edilen rotaları incelemiştir. Engeller ile kesişen rotaları üreten hızlar elemine edilmiş, üretilen rotalardaki hücre değerleri toplanmış ve en küçük değere sahip rota seçilmiştir. Denemelerde simülasyon süresi için verilen değer büyüdükçe kompütasyonel yükün arttığı gözlenmiştir. Ayrıca simülasyon süresi için verilen değer büyüdükçe daha uygun rotalar üretilebilmiştir. Simülasyon süresi için verilen değer 2 s' ye eşit veya küçük olduğu değerlerde özellikle otonom aracın dar bir koridordan manevra yapması gerektiği durumlarda sınırlı bir performans gösterdiği görülmüştür (Quigley ve ark. 2016). Denemelerde 4 s değerinin en uygun sonuçları verdiği belirlenmiştir (Şekil 4.32).



Simülasyon Süresi: 2 s Simülasyon Süresi: 4 s

Şekil 4.32. Lokal planlayıcı simülasyon süresi değişkeninin planlanan lokal rota üzerindeki etkisi

Rota planlama uygulaması otonom aracın ziyaret etmesi istenen noktaların harita üzerindeki konumları bir matris halinde verilmesi şeklinde otonom araç üzerinde uygulanmıştır. Belirtilen bu noktalar otonom araç tarafından verilen sırada ziyaret edilmiştir.

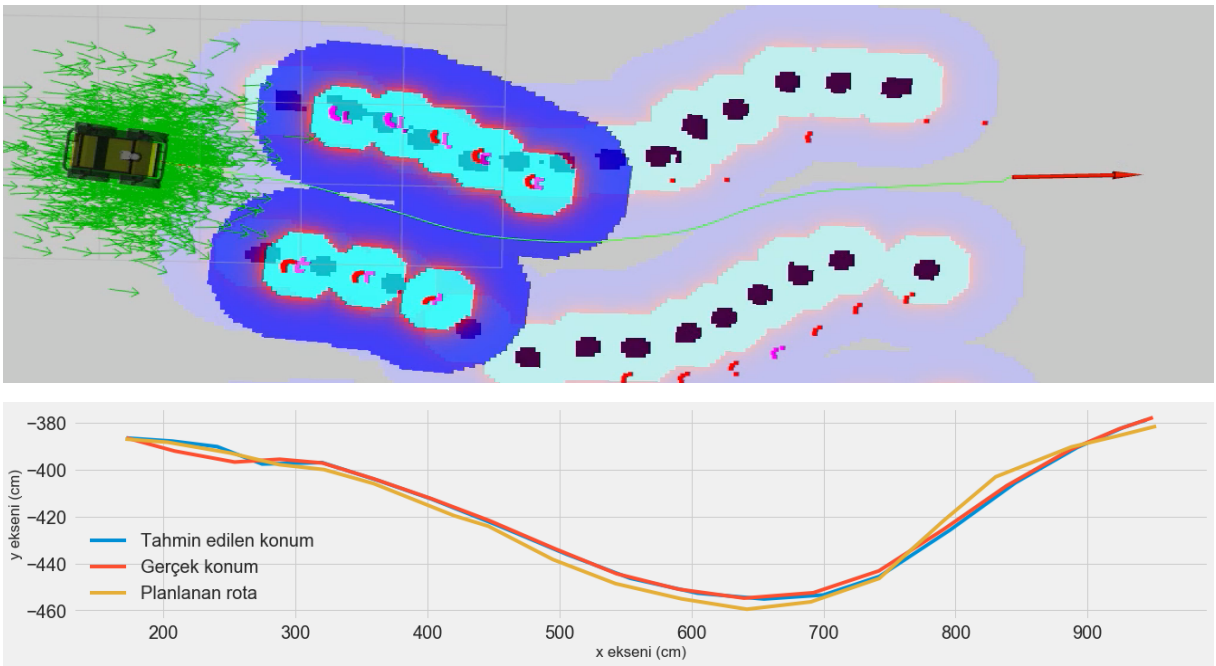
Çizelge 4.6. Otonom aracın istenilen noktalara rota planlamasını ve hareket etmesini sağlayan program örneği

```
#!/usr/bin/env python3

import rospy
import actionlib

from move_base_msgs.msg import MoveBaseAction, MoveBaseGoal

waypoints = [
        [(pos_x1, pos_y1, pose_z1), (or_x1, or_y1, or_z1, or_w1)],
        [(pos_x2, pos_y2, pose_z2), (or_x2, or_y2, or_z2, or_w2)],
        ... ] #1

def goal_pose(pose): #2
  goal_pose = MoveBaseGoal()
  goal_pose.target_pose.header.frame_id = "map"
  goal_pose.target_pose.pose.position.x = pose[0][0]
  goal_pose.target_pose.pose.position.y = pose[0][1]
  goal_pose.target_pose.pose.position.z = pose[0][2]
  goal_pose.target_pose.pose.orientation.x = pose[1][0]
  goal_pose.target_pose.pose.orientation.y = pose[1][1]
  goal_pose.target_pose.pose.orientation.z = pose[1][2]
  goal_pose.target_pose.pose.orientation.w = pose[1][3]

  return goal_pose

if __name__ == "__main__":
  rospy.init_node("patrol")

  client = actionlib.SimpleActionClient("move_base", MoveBaseAction) #3
  client.wait_for_server()

  while True:
     for pose in waypoints: #4
        goal = goal_pose(pose)
        client_send_goal(goal)
        client.wait_for_result()
```

- #1. adımda otonom araç tarafından ziyaret edilmesi istenilen noktaların listesi matris halinde programa verilmektedir.

- #2. adımda verilen bi koordinatların otonom araç tarafından algılanacak şekilde (x,y, θ) hedef haline dönüştürecek fonksiyon tanımlanmaktadır.
- #3. adımda hedef komutlarının motor sürücüye gönderilmesini sağlayacak sürücü çalıştırılmaktadır.
- #4. adımda ise verilen hedefler için sırasıyla rota planlaması yapan otonom araç hedeflere belirtilen sırada otonom olarak ulaşmaktadır (Çizelge 4.6).

Ağaç sıraları arasında planlanan bir rota Şekil 4.33' te gösterilmiştir. Planlanan rota, tahmin edilen konum ve gerçek konum görselleştirilmiş ve aralarındaki fark karşılaştırılarak rota planlama uygulamasının başarısı incelenmiştir.

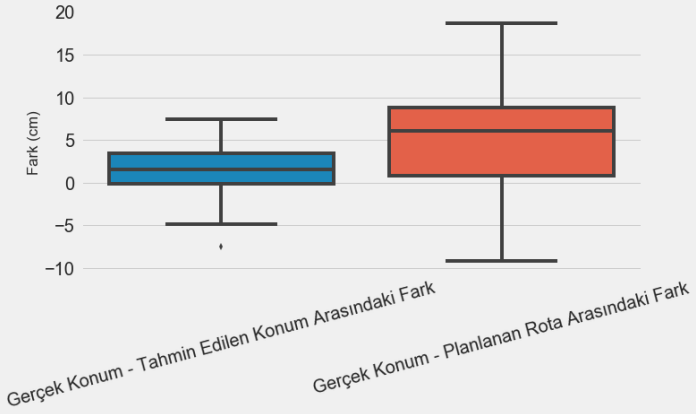


Şekil 4.33. Otonom araç tarafından planlanan rota, tahmin edilen konum ve gerçek konum gösterimi

Tasarlanan rota planlama uygulaması rota üzerinde tahmin edilen konum ve gerçek konum arasındaki fark ile, gerçek konum ile planlanan rota arasındaki fark üzerinden incelenmiştir. Gerçek ve tahmin edilen konum arasında ortalama 2,1 cm fark elde edilmiştir. Planlanan rotadan ise ortalama 5 cm sapma görülmüştür (Çizelge 4.7). Bu fark kontrol sinyallerinin uygulanıp yanıtın alınması arasında oluşan doğal farktan dolayı oluşmaktadır.

Çizelge 4.7. Planlanan rota ve gerçek konum ile gerçek ve tahmin edilen konum arasındaki fark

	Gerçek ve Tahmin Edilen Konum Arasındaki Fark (cm)	Gerçek Konum ve Planlana Rota Arasındaki Fark (cm)
Ortalama	2,1	5,0
Standard Sapma	2,9	5,7
Minimum Değer	0,0	0,0
Ortanca Değer	2,0	6,0
Maksimum Değer	7,4	18,6

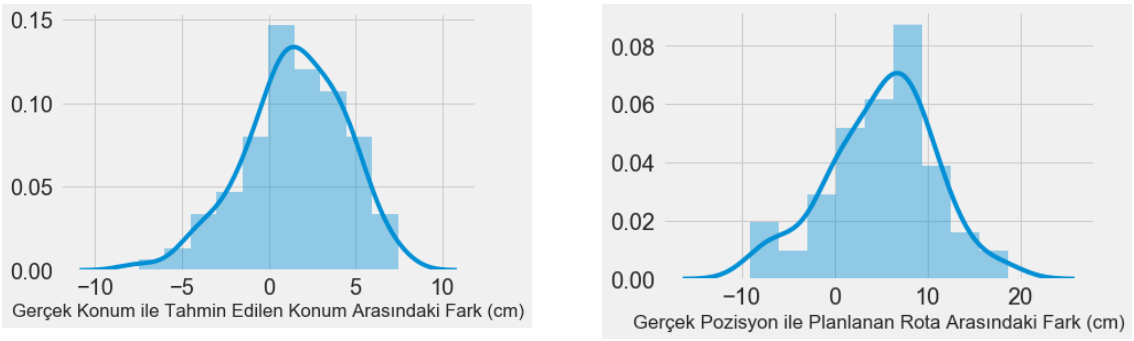


Şekil 4.34. Planlanan rota ve gerçek konum ile Gerçek ve Tahmin Edilen Konum arasındaki fark dağılımını gösteren kutu grafik

Şekil 4.35. Planlanan rota ve gerçek konum ile Gerçek ve Tahmin Edilen Konum arasındaki fark dağılımını gösteren viyolin grafik

Şekil 4.34 ve Şekil 4.35' teki kutu ve viyolin grafik şekillerinde gerçek konum ve tahmin edilen konum ile, gerçek konum ve planlanan rota arasındaki fark değerlerinin dağılımı görülmektedir. Kutu grafik, değerlerin dağılımını görmek için uygundur ancak her bir değerin maksimum, minimum ve ortanca nokta arasındaki örnek dağılımını görmek için viyolin grafik kullanılmıştır.

Örnek dağılımı gerçek konum ve tahmin edilen konum arasındaki ortanca değer olan 2 cm etrafında dağıldığı görülmüştür. Gerçek konum ve planlanan rota arasındaki fark değerleride ortanca değer olan 6 cm etrafında dağılmıştır ancak varyansı gerçek ve tahmin edilen konum arasındaki farktan daha yüksektir.



Şekil 4.36. Planlanan rota ve gerçek konum ile Gerçek ve Tahmin Edilen Konum arasındaki fark dağılım

İki farklı kongifürasyon için gerçek ve tahmin edilen ile gerçek ve planlanan rota konumu arasındaki fark değerlerinin dağılımları incelendiğinde normal dağılıma sahip oldukları görülmüştür (Şekil 4.36).

4.4. İlaçlamanın Gerçekleştirilmesi

İstenilen noktalarda ilaçlamanın başlatılıp diğer noktalarda kapatılması için otonom aracın arazi üzerindeki konumunu belirli bir tolerans içerisinde bilmesine bağlıdır. Tasarlanan sistem simülasyon üzerinde çeşitli hareket senaryoları ile test edilmiştir. Otonom araç simülasyon üzerinde yapılan lokalizasyon testlerinde görüldüğü üzere sıra içerisinde 1 m/s ilerleme hızında ilerlerken 2,1 cm ortalama hata ile sıra sonlarındaki dönüşler de hesaba katıldığında 3,5 cm ortalama hata ile konumunu belirleyebilmektedir. Daha önce geliştirilen kanopi hacmine göre değişken düzeyli ilaçlama sistemi ilaçlama kollarının sıra merkezine olan mesafesine göre hesaplama yapmaktadır. Tasarlanan navigasyon sisteminde uygun

çözünürlükte sensörler kullanıldığında RTK-GPS hassasiyetinde (Perez-Ruiz ve ark. 2015) konum ölçme sağlanabileceği için bu tür ilaçlama uygulaması için uygundur. Ortam haritası oluşturulduktan sonra ağaç sıralarını içerisine alan bir dörtgen belirlenmiş, otonom araç ortam üzerinde ilerlerken lokalizasyon uygulaması tarafından algılanan otonom araç konumunun bu dörtgen sınırları içerisinde olup olmadığı ((x_0, y_0) ve (x_1, y_1)) kontrol edilmiştir. Otonom aracın konumuna göre açma/kapama komutlarını üretecek bir program geliştirilmiştir.

4.5. Toparlanma Davranışları

Yukaridaki parametreler uygulamaya yönelik ne kadar optimize edilirse edilsin uygulama sırasında otonom aracın bir engele takılıp kalacağı ve global rota planınını icra edemeyeceği şartlar oluşabilir. Bu gibi durumlarda otonom aracın nasıl bir aksiyon sergileyeceğinin belirlenmesi önemlidir.

Toparlanma davranışı için iki farklı yöntem kullanılmıştır. Bunlardan birincisi lokal rota planının sıfırlanarak yeni bir global rota planı hesaplanmasıdır. Böylece lokal rota planı tekrardan hesaplanmış bir kaçış yolu bulunmaya çalışılmıştır. Birinci yöntem denendikten sonra uygun bir kaçış rotası bulunamadıysa ikinci yöntem olarak bulunduğu konumda 360° dönmesiyle kaçış aksiyonları üretmeye çalışacak şekilde programlanmıştır. Bu iki yöntem denendikten sonra otonom araç sıkıştığı konumdan kurtulacak bir plan üretemiyorsa otonom aracın bulunduğu pozisyonda durarak kalması sağlanmıştır.

5. SONUÇ ve ÖNERİLER

Bu araştırma neticesinde; tel terbiyeli meyve bahçelerinde ağaçların kanopi hacimlerine göre ilaçlama yapan hassas ilaçlama sistemini sürücü kullanmadan otonom bir şekilde taşıyabilecek, hassas konum tespiti sayesinde ilaçlamayı sadece istenilen noktalarda başlatabilecek ve bahçe içerisinde belirlenecek hedeflere otonom şekilde gidebilecek bir otonom araç tasarımı yapılmıştır.

Bu amaçla ortamdaki engellerin birbirlerine olan mesafelerini gösteren bir haritalama uygulaması, otonom araç tarafından üretilen bu harita üzerinde otonom aracın kendi konumunu bulabilmesini sağlayacak lokalizasyon uygulaması ve istenilen hedef/hedeflere gitmeyi sağlayacak aynı zamanda önüne çıkan engellerden kaçmayı gerçekleştirecek rota planlama uygulaması geliştirilmiştir.

Sonuçlar;

- Geliştirilen haritalama uygulaması LIDAR ve odometri sensörlerinden alınan veriler ile otomatik olarak harita oluşturmaktadır. Oluşturulan harita png veya jpg formatında kaydedilmektedir. Oluşturulan harita ile beraber harita boyutu, harita çözünürlüğü, engel ve boş alan eşik değeri verilerini içeren bir öznitelik dosyası ROS tarafından sağlanacak harita sunucuya girdi olarak verilerek tasarlanan navigasyon sisteminin ihtiyaç duyduğu harita sağlanmıştır.
- Görüntü dosyası halinde oluşturulan harita Microsoft Paint gibi bir editör ile istenilen şekilde düzenlenebilir. Harita üzerinde otonom aracın girmesini istemediğimiz ağaç sıraları veya bölgeler varsa buralar siyah bir çizgiyle çevrilerek rota planlama aşamasında gözardı edilebilir.
- Oluşturulan haritaların doğrulukları, otonom aracın pozisyonuna ait sonsal olasılık dağılımından entropisi hesaplanarak bulunabilir. Entropi değeri 0' a yaklaştıkça haritanın doğruluğu artacaktır.
- Harita oluşturma sırasında farklı güncelleme değerleri için entropi ölçümü yapılmış ve karşılaştırılmıştır. Güncelleme değeri büyüdükçe haritanın entropisi artmakta bu değer küçüldüğünse ise entropi düşmektedir. Güncelleme değerinin düşürülmesi ihtiyaç duyulan işlem gücünü arttırmaktadır. En düşük ortalama entropi 3,03 olarak, 50 cm yer değiştirme olduğunda güncelleme yapılan konfigürasyonda elde edilmiştir. En yüksek

ortalama entropi 5,81 olarak, 200 cm yer değiştirme olduğunda güncelleme yapılan konfigürasyonda elde edilmiştir.

- Harita oluşturma algoritmasında kullanılan parçacık sayısının arttırılması haritanın doğruluğunu arttırmaktadır ancak parçacık sayısı arttıkça ihtiyaç duyulan bilgisayar işlem gücü de üstel olarak büyümektedir.
- Üç farklı parçacık sayısı değeri için harita oluşturmada elde edilen entropi değerleri karşılaştırılmış en yüksek ortalama entropi 6,61 ile 3 parçacık kullanılan konfigürasyonda, en düşük ortalama entropi değeri ise 2,06 olarak 300 parçacık kullanılan uygulamada elde edilmiştir.
- Harita oluşturmada 30 parçacık kullanımı ile 300 parçacık kullanımı arasında entropi açısından %47 iyileşme sağlanırken ihtiyaç duyulan işlem gücü 10 kat artmaktadır. Bunun sebebi ağaç sıraları gibi benzer ve tekrarlı yapılar bulunan uygulamalarda, parçacık sayısı ortamı ifade edecek bir sayıya ulaştıktan sonra, arttırılmasının aynı doğrusal faydayı sağlayamamasıdır.
- Harita oluşturmada entropi ve işlem gücü arasındaki ilişki gözönüne alınarak 50 cm yer değiştirmede güncelleme yapılmış ve 30 parçacık kullanılmıştır.
- Rota planlama aşamasında önemli olan tolerans katmanı otonom aracın izdüşüm büyüklüğüne göre ayarlamaktadır. Tolerans katmanı otonom aracın izdüşümüne dış teğet çemberin yarı çapı değiştirilerek ayarlanmaktadır. Kullanılan sıra aralığı için (1.5m) otonom aracın ağaçlara çarpmadan ilerlemesi ve dönebilmesi amacıyla dış teğet çemberin 1.5 katı tolerans katmanı genişliği kullanılmıştır.
- LIDAR ve odometri sensörlerinden alınan veriler harita ile karşılaştırılarak otonom aracın harita üzerindeki konumunu bulan lokalizasyon uygulaması geliştirilmiştir. Başlangıç konumunun doğru olarak verilmesi otonom aracın tahmin edilen konumu ile gerçek konumunun birbirine daha kısa sürede yakınsamasını sağlamaktadır.
- Farklı hareket konfigürasyonlarında geliştirilen lokalizasyon uygulamasının simülasyon üzerinde ortalama 2,1 cm ile 3,5 cm arasında konumlama hatası ile konum tahmini yapabildiği görülmüştür.
- Lokalizasyon uygulaması geliştirilirken parçacık filtresi temelli adaptif monte karlo lokalizasyon yöntemi kullanılmıştır. Bu yöntemde parçacık sayısı konum kesinliğine göre belirlenen sınırlar içerisinde otomatik olarak ayarlanabilmektedir. Farklı parçacık sayıları için simülasyon üzerinde konumlama hatası incelendiğinde 5-20 parçacık kullanan konfigürasyonda 15,3 cm ortalama hata, 50-200 parçacık içeren

konfigürasyonda 10,2 cm ortalama hata, 500-2000 parçacık kullanılan uygulamada 3,5 cm ortalama hata ve 5000-20000 parçacık kullanılan uygulamada ise 2,3 cm ortalama hata gözlenmiştir. Kullanılan parçacık sayısı arttıkça ortalama hata düşmekte ancak gerekli olan bilgisayar işlem gücü üstel olarak artmaktadır. Geliştirilen lokalizasyon uygulamasında 500-2000 parçacık kullanılan konfigürasyon tercih edilmiştir.

- Lokalizasyon başarısını etkileyen diğer bir faktör otonom araç ne kadar yer değiştirme yaptığında konum güncellemesi yapılacağıdır. Güncelleme değeri düştüğünde (sensör ölçüm limitleri içerisinde) konum doğruluğu artarken ihtiyaç duyulan bilgisayar işlem gücü doğrusal olarak artmaktadır. En az ortalama konum hatası 3,5 cm ile 2 cm yer değişikliği olduğunda konum güncellemesi yapılan konfigürasyonda elde edilmiştir. En büyük ortalama konumlama hatası 9,2 cm ile 8 cm yer değiştirme olduğunda konum güncellemesi yapılan konfigürasyonda elde edilmiştir.
- Tasarlanan rota planlama uygulaması birbirine tandem olarak çalışan global ve lokal rota planlayıcı olmak üzere iki kısımdan oluşmaktadır. Global planlayıcıda engeller, otonom aracın konumu ve hedefe göre hesaplanan genel plan lokal planlayıcıya gönderilir. Lokal planlayıcı sensörlerden alınan verileri inceleyerek sensörlerin görüş alanına girmiş engellerden otonom aracın kaçınmasını sağlar. Lokal planlayıcıda dinamik pencere yaklaşımı kullanılmıştır. Bu yaklaşımda otonom aracın hız uzayı örneklenerek belirtilen simülasyon süresi içerisinde kullanılmak üzere lokal rota seçenekleri oluşturulur. Simülasyon süresinin uzun tutulması bilgisayar işlem gücü ihtiyacını arttırırken oluşturulan rotaların kesinliğini arttırır. Simülasyon süresinin 4 s olarak seçilmesinin otonom aracın dar koridorlardan geçerken daha esnek olmasını sağladığı görülmüştür.
- Uygulama sırasında otonom aracın engelden kaçamayacağı durumlar oluşabilir. Bu durumlarda otonom aracın icra edebilmesi için iki aşamalı bir toparlanma ve kaçış planı yazılımda uygulanmıştır. Birinci adım olarak lokal planlayıcı sıfırlanarak yeni bir global plan oluşturulmaya çalışmakta bu başarısız olursa otonom araç etrafında dönerek yeni rotalar bulmaya çalışmaktadır. Eğer otonom aracın etrafında dönüşü çarpışma riski nedeniyle mümkün olmazsa otonom araç rota planlamayı durdurarak olduğu yerde kalmakta ve hata sinyali vermektedir.
- Yaprak alanı yoğunluğuna göre değişken oranlı ilaçlama yapan sistem hesaplama işlemini ilaçlama kollarının sıra merkezine konumuna göre yaptığı için otonom aracın sıra üzerindeki konumunun hassas olarak bilinmesi değişken düzeyli ilaçlamada

kullanılacak otonom araç için önemlidir. Tasarlanan sistem simülasyon ortamında yapılan lokalizasyon testlerinde görüldüğü üzere sıra içerisinde 2,1 cm ortalama hata ile ilerlerken sıra sonlarındaki dönüşlerde hesaba katıldığında 3,5 cm ortalama hata ile konumunu belirleyebilmektedir. RTK-GPS seviyesinde bir konum doğruluğu sağlayabildiği için hassas ilaçlama uygulamaları için uygundur.

- Geliştirilen tarımsal otonom araç boyut ve ağırlığının klasik traktörlere göre daha az olması sebebiyle hem manevra avantajı sağlayacak hemde daha az toprak sıkışıklığına sebebiyet verecektir.
- Tasarımı yapılan otonom araç hem sürücüsüz hem de sürücülü kullanılabilecektir.

Öneriler;

- Geliştirilen sistem LIDAR sensör ile 2 boyutlu ölçüm almasına rağmen kolaylıkla 3 boyutlu veri alınacak şekilde adapte edilebilir. Bu sayede ağaçların 3 boyutlu modellenmesi sağlanabilir.
- Sensörlerin yerleşimi navigasyonun başarısını arttırmaktadır ve ölçülen verinin işlenmesini basitleştirmektedir. Otonom aracın ön tarafına, gidiş yönüne bakacak şekilde yerleştirilen LIDAR sensör navigasyonu kolaylaştırmaktadır. Bahçe özelliğine bağlı olarak LIDAR sensörün yere yakın olması erken dönem bitki ve alçak engellerinde algılanmasını sağlayacaktır.
- Geliştirilen navigasyon uygulaması diferansiyel sürüşe sahip otonom araç platformları için uygundur. Doğrusal ve açısal hız değerleri uygulama tarafından çıktı olarak verilmektedir ve herhangi bir elektrik motor sürücü ile uygun elektriksel bağlantı sağlandığı takdirde çalışabilir.
- Otonom aracın haritada belirtilen çözünürlük değerlerine sahip bir LIDAR sensör kullanılması gereklidir. LIDAR ve odometri sensör çözünürlüğü düştükçe navigasyon başarısı düşecektir.
- Otonom aracın, yazılım tasarımından dolayı dörtgen veya dairesel bir şekle sahip olması gereklidir. Otonom araç üzerine takılan odometri (devir ölçer ve atalet) ve LIDAR sensörlerinin dönüş merkezine olan uzaklıkları tam olarak belirtilmelidir. Bu değerler ROS tarafından otomatik olarak dönüş merkezine göre bir geometrik dönüşüme tabi tutulmaktadır.
- Tarımsal mekanizasyon sistemleri artan gıda ihtiyacına karşılık verecek şekilde otomasyonu kullanmak zorundadırlar. Geliştirilen otomasyon uygulamaları

ölçeklenebilir olmalıdır. Geleceğin tarımsal üretiminde yabancı ot kontrolü, ilaçlama hasat ve bitki bakımı gibi işlemler çok sayıda robot ile otonom şekilde yapılmak zorundadır. Bu otonom araçlar lokal verilere göre bağımsız karar alabilecek kapasitede olmalıdırlar. Bu yeni konsept içerisinde çiftçi sadece acil durumlarda müdahele edecek diğer durumlarda uzaktan bir danışman olarak sadece sistemin verilerini izleyecektir. Olasılıksal yöntemler tel terbiyeli meyve bahçeleri gibi yarı düzenli ortamlarda otonom sistemler tasarlamak, yönetmek ve haberleşmek için kullanılacak tek yöntemdir.

- Oluşturulan haritada karşılaşılan hatalar gelişen teknoloji ile daha yüksek çözünürlüğe sahip sensörler veya daha fazla sayıda sensör kullanılarak giderilebilir. Bunun yanısıra daha fazla parçacık kullanımı, daha yüksek güncelleme hızı ve haritalama parametrelerinin uygulama ortamına göre optimize edilmesi haritalama başarısını arttıracaktır. Haritalama aşamasında otonom aracın yavaş kullanımı, dönüşlerin yavaş yapılması ve aynı yerden birden fazla kez geçilmesi üretilen haritanın kalitesini arttıracaktır.
- Başarılı bir navigasyon için otonom aracın harita üzerindeki lokalizasyonunun doğru bir şekilde yapılması önemlidir. Başlangıç durumunda lokalizasyonun doğru yapılması için LIDAR sensörden alınan verilerin harita ile eşleşmesi rviz ortamında kontrol edilerek, başlangıç lokalizasyonu iyileştirilebilir. Bunun yanısıra otonom sürüş başlamadan önce aracın ortamda bir süre dolaştırılması lokalizasyonun doğru bir şekilde yapılmasına yardımcı olacaktır.
- Rota planlamada, ortamın özelliklerine bağlı olarak engel haritasındaki tolerans katmanı ayarlanabilir. Böylece engellere çarpmadan daha güvenli bir sürüş sağlanabilir. Otonom aracın fiziksel özelliklerinin doğru olarak tanımlanması, sensörlerin yerleşiminin yazılıma doğru olarak girilmesi rota planlamanın başarısını arttıracaktır. Ortamın zorluğuna göre ileri simülasyon zamanının arttırılması rota planlama ve takibindeki başarıyı arttırırken ihtiyaç duyulan bilgisayar işlem gücünü de arttıracaktır. Bu nedenle uygulama şartlarına uygun olarak bu değişkenin seçilmesi gerekmektedir.
- Otonom araç prototipi geliştirilirken rota planlamasında elde edilecek dönüş açıları otonom aracın kullanıldığı zemine bağlı olarak, donanımın elektriksel özellikleri dikkate alınarak incelenmeli ve gerektiğinde dönüş açıları belli değerler ile sınırlandırılmalıdır.

- Böyle bir otonom aracın şu an için LIDAR sensör maliyetinden dolayı fiyat dezavantajı olmakla birlikte bu teknolojinin ülkemizde geliştirilmesi ve bilgi birikiminin sağlanması önemlidir.
- Tarımda otonom araç kullanımı traktör sürücüsü gibi klasik iş gücünü azaltma durumu oluştursa bile bu sistemlerin programlanması, elektronik donanımların tasarımı ve bakımı, ortam haritasının oluşturulması gibi konularda yeni iş olanakları sağlayacaktır.
- Büyük boyutlu konvansiyonel traktörlerin otonom yapılması güvenlik açısından bazı riskler barındırabilmektedir. Otonom konvansiyonel boy, tek bir traktör yerine birden fazla küçük ve orta boyutlu otonom tarım aracı kullanımı daha güvenli bir kullanım sağlayacaktır. Birden fazla otonom aracın birbiriyle paralel şekilde çalışması ile, herhangi bir aracın yaşadığı sorun nedeniyle devre dışı kalması durumunda diğer araçlar bunu telafi edebilecektir. Otonom araçlar üzerinde boyut olarak daha küçük ekipmanlar kullanılacağı için daha hassas işlemler yapılabilir. Bunun yanı sıra birden fazla robot otonom olarak hareket edebilmekte ve tek bir operatör tarafından kontrol edilebilmektedir. Ancak konvansiyonel traktörlerin her biri için bir operatöre ihtiyaç bulunmaktadır.

6. KAYNAKLAR

Ahamed T, Takigawa T, Koike M, Honma T, Yoda A, Hasegawa H, Zhang Q (2004). Characterization Of Laser Range Finder For In-Field Navigation Of Autonomous Tractor. Automation Technology for Off-Road Equipment Proceedings of the Conference, 120-130, Japan. doi:10.13031/2013.17825.

Andersone I (2017). Probabilistic Mapping with Ultrasonic Distance Sensors. *Procedia Computer Science, 104*: 362-368. doi: 10.1016/j.procs.2017.01.146.

Anguelov D, Koller D, Parker E, Thrun S (2004). Detecting and Modeling Doors with Mobile Robots. IEEE International Conference on Robotics and Automation Proceedings ICRA '04, 3777-3784, USA. doi:10.1109/robot.2004.1308857.

Anonim (2018). Akınsoft Tarım Robotu. http://www.akinrobotics.com/akinsoft-tarim-robotu.php (erişim tarihi: 05.11.2018).

Anonim (2018a). rviz. http://wiki.ros.org/rviz (erişim tarihi: 05.11.2018).

Anonim (2018b). rqt_plot. http://wiki.ros.org/rqt_plot (erişim tarihi: 05.11.2018).

Anonim (2018c). rqt_graph. http://wiki.ros.org/rqt_graph (erişim tarihi: 05.11.2018).

Anonim (2018d). 2D-LIDAR sensörleri LMS1xx. https://www.sick.com/tr/tr/oelcuem-ve-alglama-coezuemleri/2d-lidar-sensoerleri/lms1xx/lms111-10100/p (erişim tarihi: 05.11.2018).

Anonim (2018e). Quadrature Encoder. http://www.mbeddedc.com/2015/11/quadrature-encoder-in-c-quadrature.html (erişim tarihi: 05.11.2018).

Anonim (2018f). UM6 Orientation Sensor. http://www.chrobotics.com/shop/orientation-sensor-um6 (erişim tarihi. 05.11.2018).

Arkin RC (1998). Behavior-based robotics. MIT Press, 506 p, USA.

Arras K, Vestli S (1998). Hybrid, High-Precision Localisation For the Mail Distributing Mobile Robot System MOPS. Proceedings of 1998 IEEE International Conference on Robotics and Automation, 3129-3134, Belgium. doi:10.1109/robot.1998.680906

Astolfi P, Gabrielli A, Bascetta L, Matteucci M (2018). Vineyard Autonomous Navigation in the Echord GRAPE Experiment. *IFAC-PapersOnLine, 51*(11): 704-709. doi: 10.1016/j.ifacol.2018.08.401

Astrand B (2005). Vision Based Perception for Mechatronic Weed Control. Ph.D. Thesis, Chalmers University of Technology, Sweden.

Aznar F, Pujol FA, Pujol M, Rizo R, Pujol M (2014). Learning Probabilistic Features for Robotic Navigation Using Laser Sensors. PLoS ONE, 9(11): 1-21. doi:10.1371/journal.pone.0112507.

Barawid OC, Mizushima A, Ishii K, Noguchi N (2007). Development of an Autonomous Navigation System using a Two-dimensional Laser Scanner in an Orchard Application. Biosystems Engineering, *96*(2: 139-149. doi: 10.1016/j.biosystemseng.2006.10.012.

Belforte G, Deboli R, Gay P, Piccarolo P, Aimonino DR (2006). Robot Design and Testing for Greenhouse Applications. Biosystems Engineering, *95*(3): 309-321. doi: 10.1016/j.biosystemseng.2006.07.004.

Bell T (2000). Automatic Tractor Guidance Using Carrier-Phase Differential GPS. Computers and Electronics in Agriculture, *25*(1-2): 53-66. doi:10.1016/s0168-1699(99)00055-1.

Bergerman M, Singh S, Hamner B (2012). Results with Autonomous Vehicles Operating in Specialty Crops.IEEE International Conference on Robotics and Automation, 1829-1835, USA. doi:10.1109/icra.2012.6225150

Billingsley J, Schoenfisch M (1997). The Successful Development of a Vision Guidance System for Agriculture. Computers and Electronics in Agriculture, *16*(2): 147-163. doi:10.1016/s0168-1699(96)00034-8.

Biswas R, Limketkai B, Sanner S, Thrun S (2002). Towards Object Mapping in Non-Stationary Environments with Mobile Robots. IEEE/RSJ International Conference on Intelligent Robots and System, 1014-1019, Switzerland. doi:10.1109/irds.2002.1041523

Bonadies S, Gadsden SA (2018). An Overview of Autonomous Crop Row Navigation Strategies for Unmanned Ground Vehicles. Engineering in Agriculture, Environment and Food,https://www.sciencedirect.com/science/article/abs/pii/S188183661730188X?via%3Dihub (erişim tarihi: 05.11.2018).

Borenstein J, Koren Y (1991). The Vector Field Histogram-Fast Obstacle Avoidance for Mobile Robots. IEEE Transactions on Robotics and Automation, 7(3): 278-288. doi:10.1109/70.88137.

Brooks RA, Lozano-Perez T (1985). A Subdivision Algorithm in Configuration Space for Findpath With Rotation. IEEE Transactions on Systems, Man, and Cybernetics, SMC-15(2): 224-233. doi:10.1109/tsmc.1985.6313352.

Bubeck A (2013). Modular Snake ROS. ROSCon - Stuttgart / Germany, https://roscon.ros.org/2013/wp-content/uploads/2013/06/MSROSCon2013.pdf (erişim tarihi: 05.11.2018).

Burlina P, DeMenthon D, Davis LS (1991). Navigation with Uncertainty. University of Maryland, Center for Automation Research, Computer Vision Laboratory.

Burgard W, Derr A, Fox D, Cremers A (1998). Integrating Global Position Estimation and Position Tracking for Mobile Robots: The Dynamic Markov Localization Approach. IEEE/RSJ International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications, 730-735, Canada. doi:10.1109/iros.1998.727279.

Campos Y, Sossa H, Pajares G (2016). Spatio-Temporal Analysis for Obstacle Detection in Agricultural Videos.Applied Soft Computing, *45*: 86-97. doi: 10.1016/j.asoc.2016.03.016.

Canny J (1988). The complexity of robot motion planning. MIT Press, 216 p, USA.

Cheein FA, Steiner G, Paina GP, Carelli R (2011). Optimized EIF-SLAM Algorithm for Precision Agriculture Mapping Based on Stems Detection. Computers and Electronics in Agriculture*, 78*(2): 195-207. doi: 10.1016/j.compag.2011.07.007.

Cheein FA, Carelli R (2013). Agricultural Robotics: Unmanned Robotic Service Units in Agricultural Tasks. IEEE Industrial Electronics Magazine, *7*(3): 48-58. doi:10.1109/mie.2013.2252957.

Cox IJ, Wilfong G T, Lorenzo-Pérez T (1990). Autonomous Robot Vehicles. Springer, 458 p, USA.

Cox IJ (1991). Blanche-An Experiment in Guidance and Navigation of An Autonomous Robot Vehicle. IEEE Transactions on Robotics and Automation, 7(2): 193-204. doi:10.1109/70.75902.

Crowley J (1989). World Modeling and Position Estimation for A Mobile Robot Using Ultrasonic Ranging. International Conference on Robotics and Automation, 674-680, USA. doi:10.1109/robot.1989.100062.

Çelen I, Arin S, Durgut M (2008). The Effect of the Air Blast Sprayer Speed on the Chemical Distribution in Vineyard. Pakistan Journal of Biological Sciences, 11(11): 1472-1476. doi:10.3923/pjbs.2008.1472.1476.

Çelen İH., Önler E, Kiliç E (2015). A Design of Autonomous Agricultural Robot to Navigate Between Rows, International Conference of Electrical, Automation and Mechanical Engineering, 349-352, Thailand.

Dellaert F, Fox D, Burgard W, Thrun S (1999). Monte Carlo Localization for Mobile Robots. IEEE International Conference on Robotics and Automation, 1322-1328, USA. doi:10.1109/robot.1999.772544

Dickmanns ED, Graefe V (1988). Dynamic Monocular Machine Vision. Machine Vision and Applications, 1(4): 223-240. doi:10.1007/bf01212361.

Dissanayake G, Durrant-Whyte H, Bailey T. (2000). A Computationally Efficient Solution to The Simultaneous Localisation And Map Building (SLAM) Problem. IEEE International Conference on Robotics and Automation, 1009-1014, USA. doi:10.1109/robot.2000.844732

Dolgov D, Thrun S, Montemerlo M, Diebel J (2010). Path Planning for Autonomous Vehicles in Unknown Semi-Structured Environments. The International Journal of Robotics Research, *29*(5): 485-501. doi:10.1177/0278364909359210.

Donald BR (1984). Motion Planning with Six Degrees of Freedom. Computer Science and Artificial Intelligence Lab-MIT, USA. doi:10.21236/ada181538

Donald BR (1987). A Search Algorithm for Motion Planning with Six Degrees of Freedom. Artificial Intelligence, *31*(3): 295-353. doi:10.1016/0004-3702(87)90069-5.

Dorf RC, Bishop RH (2017). Modern Control Systems. Pearson Education, 1032 p, USA.

Doruchowski G, Holownicki R (2000). Environmentally Friendly Spray Techniques for Tree Crops. Crop Protection, *19*(8-10): 617-622. doi:10.1016/s0261-2194(00)00081-8.

Doucet A, Freitas N, Gordon N (2001). An Introduction to Sequential Monte Carlo Methods. Sequential Monte Carlo Methods in Practice, 3-14. doi:10.1007/978-1-4757-3437-9_1.

Driankov D, Safiotti A (2001) Fuzzy Logic Techniques for Autonomous Vehicle Navigation. Studies in Fuzziness and Soft Computing, 385 p, USA. doi:10.1007/978-3-7908-1835-2.

Eliazar A, Parr R (2004). Dp-Slam 2.0. IEEE International Conference on Robotics and Automation Proceedings, 1314-1320, USA. doi:10.1109/robot.2004.1308006

Ericson SK, Astrand BS (2018). Analysis of Two Visual Odometry Systems for Use in An Agricultural Field Environment. Biosystems Engineering, 166: 116-125. doi: 10.1016/j.biosystemseng.2017.11.009.

Fan K, Lui P (1994). Solving Find Path Problem in Mapped Environments Using Modified A* Algorithm. IEEE Transactions on Systems, Man, and Cybernetics, *24*(9): 1390-1396. doi:10.1109/21.310516.

Fernandez B, Herrera PJ, Cerrada JA (2018). Robust Digital Control for Autonomous Skid-Steered Agricultural Robots. Computers and Electronics in Agriculture, *153*: 94-101. doi: 10.1016/j.compag.2018.07.038.

Ferreira JF, Dias J (2014). Probabilistic Learning. Springer Tracts in Advanced Robotics Probabilistic Approaches to Robotic Perception, 147-167. doi:10.1007/978-3-319-02006-8_6.

Fox D, Burgard W, Thrun S (1997). The Dynamic Window Approach to Collision Avoidance. IEEE Robotics & Automation Magazine, 4(1): 23-33. doi:10.1109/100.580977.

Fujii Y, Hayashi M (1989). Boundary detecting method and apparatus for automatic working vehicle. US Patent 4,868,752, http://patents.com/us-4868752.html (erişim tarihi: 05.11.2018).

Gamallo C, Regueiro C, Quintía P, Mucientes M (2010). Omnivision-based KLD-Monte Carlo Localization. Robotics and Autonomous Systems, *58*(3): 295-305. doi: 10.1016/j.robot.2009.08.007.

Gan-Mor S, Ronen B, Josef S, Bilanki Y (1997).Guidance of Automatic Vehicle for Greenhouse Transportation. Acta Horticulturae, 443: 99-104. doi:10.17660/actahortic.1997.443.12

Gan H, Lee W (2018). Development of a Navigation System for a Smart Farm. IFAC-PapersOnLine, *51*(17): 1-4. doi: 10.1016/j.ifacol.2018.08.051

Grisetti G, Stachniss C, Burgard W (2007). Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters. IEEE Transactions on Robotics, *23*(1): 34-46. doi:10.1109/tro.2006.889486.

Gružauskas V, Baskutis S, Navickas V (2018). Minimizing the Trade-Off Between Sustainability and Cost-Effective Performance by Using Autonomous Vehicles. Journal of Cleaner Production*, 184*: 709-717. doi: 10.1016/j.jclepro.2018.02.302.

Guan RP, Ristic B, Wang L, Palmer JL (2019). KLD Sampling with Gmapping Proposal for Monte Carlo Localization of Mobile Robots. Information Fusion*, 49*: 79-88. doi: 10.1016/j.inffus.2018.09.003.

Guan M, Wen C, Wei Z, Ng C, Zou Y (2018). A Dynamic Window Approach with Collision Suppression Cone for Avoidance of Moving Obstacles. 2018 IEEE 16th International Conference on Industrial Informatics (INDIN), 337-342, Portugal. doi:10.1109/indin.2018.8472029

Guan RP, Ristic B, Wang L, Evans R (2018a). Monte Carlo Localisation of a Mobile Robot Using a Doppler–Azimuth Radar. Automatica, *97*: 161-166. doi: 10.1016/j.automatica.2018.08.012.

Guibas LJ, Knuth D E, Sharir M (1992). Randomized Incremental Construction of Delaunay and Voronoi Diagrams. Algorithmica, 7(1-6): 381-413. doi:10.1007/bf01758770.

Guo H, Cao D, Chen H, Sun Z, Hu Y (2019). Model Predictive Path Following Control for Autonomous Cars Considering a Measurable Disturbance: Implementation, Testing, and Verification. Mechanical Systems and Signal Processing, *118*: 41-60. doi: 10.1016/j.ymssp.2018.08.028.

Gutmann J, Konolige K (1999). Incremental Mapping of Large Cyclic Environments. IEEE International Symposium on Computational Intelligence in Robotics and Automation. CIRA'99 Proceedings, 318-325, USA. doi:10.1109/cira.1999.810068

Hague T, Tillett N (1996). Navigation and Control of an Autonomous Horticultural Robot. Mechatronics, *6*(2): 165-180. doi:10.1016/0957-4158(95)00070-4.

Hahnel D, Burgard W, Fox D, Thrun S (2003). An Efficient FastSLAM Algorithm for Generating Maps of Large-Scale Cyclic Environments from Raw Laser Range Measurements. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003*)* Proceedings, 206-211, USA. doi:10.1109/iros.2003.1250629

Hamner B, Singh S, Bergerman M (2010). Improving Orchard Efficiency with Autonomous Utility Vehicles. American Scoiety of Agricultural and Biological Engineering Annual Convention Proceedings. 1009415, Pittsburgh - USA. doi:10.13031/2013.29902

Hamner B, Bergerman M, Singh S (2011). Autonomous Orchard Vehicles for Specialty Crops Production.https://www.ri.cmu.edu/pub_files/2011/7/20110808.ASABE.Autonomous_orchard_vehicles.pdf (erişim tarihi: 05.11.2018)

Han S, Zhang Q, Ni B, Reid J (2004). A Guidance Directrix Approach to Vision-Based Vehicle Guidance Systems. Computers and Electronics in Agriculture, *43*(3): 179-195. doi: 10.1016/j.compag.2004.01.007.

Heidman B, Abidine A, Upadhyaya S, Hills D, Robert P (2002). Application of RTK GPS Based Auto-Guidance System in Agricultural Production. Proceedings of the 6th International Conference on Precision Agriculture and Other Precision Resources Management, 1205-1214, Minneapolis, USA

Henkel C, Bubeck A, Xu W (2016). Energy Efficient Dynamic Window Approach for Local Path Planning in Mobile Service Robotics. IFAC-PapersOnLine, 49(15): 32-37. doi:10.1016/j.ifacol.2016.07.610.

Hertzberg J, Kirchner F (1996). Landmark-Based Autonomous Navigation in Sewerage Pipes. Proceedings of the First Euromicro Workshop on Advanced Mobile Robots (EUROBOT '96), 68-73, Germany. doi:10.1109/eurbot.1996.551883

Hiremath S, Evert FK, Braak CT, Stein A, Heijden GV (2014). Image-Based Particle Filtering for Navigation in A Semi-Structured Agricultural Environment. Biosystems Engineering, *121*: 85-95. doi: 10.1016/j.biosystemseng.2014.02.010

Hsu J, Koenig N (2012). The Gazebo Simulator as a Development Tool in ROS. https://roscon.ros.org/wp-uploads/2012/06/Gazebo%20ROSCon%20presentation.pdf (05.11.2018)

Joseph L (2018). Learning Robotics Using Python: Design, Simulate, Program, And Prototype an Autonomous Mobile Robot Using ROS, Opencv, PCL, and Python, 2nd Edition. Packt Publishing, 280 p, UK. ISBN: 978-1-78862-331-5

Kambhampati S, Davis L. (1986). Multiresolution Path Planning for Mobile Robots. IEEE Journal on Robotics and Automation, *2*(3): 135-145. doi:10.1109/jra.1986.1087051

Kaelbling LP, Rosenschein SJ (1990). Action and Planning in Embedded Agents. Robotics and Autonomous Systems, 6(1-2): 35-48. doi:10.1016/s0921-8890(05)80027-2

Kaelbling LP, Littman ML, Cassandra AR (1998). Planning and Acting in Partially Observable Stochastic Domains. Artificial Intelligence, 101(1-2): 99-134. doi:10.1016/s0004-3702(98)00023-x

Kalman RE (1960). A New Approach to Linear Filtering and Prediction Problems. Journal of Basic Engineering, 82(1): 35. doi:10.1115/1.3662552

Kavraki L, Latombe J (1994). Randomized Preprocessing of Configuration for Fast Path Planning. Proceedings of the 1994 IEEE International Conference on Robotics and Automation, 2138-2145, USA. doi:10.1109/robot.1994.350966

Kavraki L, Svestka P, Latombe J, Overmars M (1996). Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces. IEEE Transactions on Robotics and Automation, *12*(4): 566-580. doi:10.1109/70.508439

Keicher R, Seufert H (2000). Automatic Guidance for Agricultural Vehicles in Europe. Computers and Electronics in Agriculture, *25*(1-2): 169-194. doi:10.1016/s0168-1699(99)00062-9

Khatib O (1986). Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. Autonomous Robot Vehicles, 396-404. doi:10.1007/978-1-4613-8997-2_29

Koditschek D (1987). Exact Robot Navigation by Means of Potential Functions: Some Topological Considerations. Proceedings. 1987 IEEE International Conference on Robotics and Automation, 1-6, USA. doi:10.1109/robot.1987.1088038

Konolige K, Chou K (1999). Markov Localization Using Correlation. IJCAI'99 Proceedings, 2: 1080-1087, Sweeden.

Kuipers B, Modayil J, Beeson P, Macmahon M, Savelli F (2004). Local Metrical and Global Topological Maps In The Hybrid Spatial Semantic Hierarchy. IEEE International Conference on Robotics and Automation ICRA '04 Proceedings, 4845-4851, USA. doi:10.1109/robot.2004.1302485

Lamini C, Benhlima S, Elbekri A (2018). Genetic Algorithm Based Approach for Autonomous Mobile Robot Path Planning. Procedia Computer Science, *127*: 180-189. doi: 10.1016/j.procs.2018.01.113

Larsen W, Nielsen G, Tyler D (1994). Precision Navigation with GPS. Computers and Electronics in Agriculture, 11(1):85–95.

Latombe J (1991). Robot Motion Planning. Springer, 247, US.

Lebeltel O, Bessière P, Diard J, Mazer E (2004). Bayesian Robot Programming. Autonomous Robots, 16(1): 49-79. doi:10.1023/b:auro.0000008671.38949.43

Lee KH, Ehsani R (2009). A Laser Scanner Based Measurement System for Quantification of Citrus Tree Geometric Characteristics. Applied Engineering in Agriculture, *25*(5): 777-788. doi:10.13031/2013.28846

Leonard J, Durrant-Whyte H (1991). Mobile Robot Localization by Tracking Geometric Beacons. IEEE Transactions on Robotics and Automation, 7(3), 376-382. doi:10.1109/70.88147

Liu C, Zhao X, Du Y, Cao C, Zhu Z, Mao E (2018). Research on Static Path Planning Method of Small Obstacles for Automatic Navigation Of Agricultural Machinery. IFAC-PapersOnLine, *51*(17): 673-677. doi: 10.1016/j.ifacol.2018.08.119

Llorens J, Gil E, Llop J, Escolà A (2011). Ultrasonic and LIDAR Sensors for Electronic Canopy Characterization in Vineyards: Advances to Improve Pesticide Application Methods. Sensors, *11*(2): 2177-2194. doi:10.3390/s110202177

Lozano-Pérez T, Wesley MA (1979). An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles. Communications of the ACM, 22(10): 560-570. doi:10.1145/359156.359164

Lozano-Pérez T (1990). Spatial Planning: A Configuration Space Approach. Autonomous Robot Vehicles, 259-271. doi:10.1007/978-1-4613-8997-2_20

Lu F, Milios E (1997).Globally Consistent Range Scan Alignment for Environment Mapping. Autonomous Robots, *4*(4): 333-349. doi:10.1023/a:1008854305733

Malavazi FB, Guyonneau R, Fasquel J, Lagrange S, Mercier F (2018). LiDAR-only based navigation algorithm for an autonomous agricultural robot. Computers and Electronics in Agriculture, *154*: 71-79. doi:10.1016/j.compag.2018.08.034

Marchant J (1996). Tracking of row structure in three crops using image analysis. Computers and Electronics in Agriculture, *15*(2): 161-179. doi:10.1016/0168-1699(96)00014-2

Marchant JA, Brivot R (1995). Real-Time Tracking of Plant Rows Using a Hough Transform. Real-Time Imaging, *1*(5): 363-371. doi:10.1006/rtim.1995.1036

Meichen L, Jun C, Xiang Z, Lu W, Yongpeng T (2018). Dynamic Obstacle Detection Based on Multi-Sensor Information Fusion. IFAC-PapersOnLine, *51*(17): 861-865. doi:10.1016/j.ifacol.2018.08.086

Meng R, Dennison PE, Zhao F, Shendryk I, Rickert A, Hanavan RP, Serbin SP (2018). Mapping Canopy Defoliation by Herbivorous Insects at The Individual Tree Level Using Bi-Temporal Airborne Imaging Spectroscopy and Lidar Measurements. Remote Sensing of Environment, 215: 170-183. doi: 10.1016/j.rse.2018.06.008

Merwe RVD, Freitas ND, Doucet A, Wan E (2000). The Unscented Particle Filter. Technical Report CUED/F-INFENG/TR380, Cambridge University Engineering Department.

Montemerlo M, Thrun S, Whittaker W (2002). Conditional Particle Filters for Simultaneous Mobile Robot Localization and People-Tracking. Proceedings IEEE International Conference on Robotics and Automation, 695-701, USA. doi:10.1109/robot.2002.1013439

Montemerlo M, Thrun S, Koller D, Wegbreit B (2003). Fast SLAM 2.0: An Improved Particle Filtering Algorithm for Simultaneous Localization and Mapping That Provably Converges. In Proc. of the Int. Conf. on Artificial Intelligence (IJCAI), 1151–1156, Acapulco, Mexico.

Moravec HP (1989). Sensor Fusion in Certainty Grids for Mobile Robots. Sensor Devices and Systems for Robotics, 253-276. doi:10.1007/978-3-642-74567-6_19

Moravec HP (1996). Robot Spatial Perception by Stereoscopic Vision And 3D Evidence Grids. Carnegie Mellon University, the Robotics Institute.

Murphy K, Russell S (2001). Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks. Sequential Monte Carlo Methods in Practice, 499-515. doi:10.1007/978-1-4757-3437-9_24

Murphy K (1999). Bayesian Map Learning in Dynamic Environments. In Proc. of the Conf. on Neural Information Processing Systems (NIPS), 1015–1021, Denver, CO, USA

Mutz F, Veronese LP, Oliveira-Santos T, Aguiar ED, Cheein FA, Souza AF (2016). Large-Scale Mapping in Complex Field Scenarios Using an Autonomous Car. Expert Systems with Applications, *46*: 439-462. doi:10.1016/j.eswa.2015.10.045

Niu H, Lu Y, Savvaris A, Tsourdos A (2018). An Energy-Efficient Path Planning Algorithm for Unmanned Surface Vehicles. Ocean Engineering, *161*: 308-321. doi:10.1016/j.oceaneng.2018.01.025

Nolte BH, Fausey NR (2001). Soil Compaction and Drainage. http://ohioline.osu.edu/b301/index.html (erişim tarihi: 05.11.2018).

Ortin D, Neira J, Montiel J (2004). Relocation Using Laser and Vision. IEEE International Conference on Robotics and Automation Proceedings, 1505-1510, USA. doi:10.1109/robot.2004.1308037

Önler E, Çelen İH, Kiliç E, Durgut MR (2015). Ultrasonik Sensör Yardimiyla Belirlenen Yaprak Yoğunluğuna Göre Püskürtme Miktarini Ayarlayan Sistemin Damla Karakteristikleri, 29. Ulusal Tarimsal Mekanizasyon Ve Enerji Kongresi, 312-316, Diyarbakır.

Park S, Pfenning F, Thrun S (2005). A Probabilistic Language Based Upon Sampling Functions. ACM SIGPLAN Notices, 40(1): 171-182. doi:10.1145/1047659.1040320

Pierzchała M, Giguère P, Astrup R (2018). Mapping Forests Using an Unmanned Ground Vehicle With 3D Lidar And Graph-SLAM. Computers and Electronics in Agriculture, *145*: 217-225. doi:10.1016/j.compag.2017.12.034

Prado AJ, Cheein FA, Blazic S, Torres-Torriti M (2018). Probabilistic Self-Tuning Approaches for Enhancing Performance of Autonomous Vehicles In Changing Terrains. Journal of Terramechanics, *78*: 39-51. doi:10.1016/j.jterra.2018.04.001

Pérez-Ruiz M, Gonzalez-De-Santos P, Ribeiro A, Fernandez-Quintanilla C, Peruzzi A, Vieri M, Agüera J (2015). Highlights and Preliminary Results for Autonomous Crop Protection. Computers and Electronics in Agriculture, *110*: 150-161. doi:10.1016/j.compag.2014.11.010

Quigley M (2012). ROS: Past, Present, and Future, https://www.youtube.com/watch?v=-E1O98qrfBY (erişim tarihi: 05.11.2018).

Quigley M, Gerkey B, Smart WD (2016). Programming Robots With ROS: A Practical Introduction to the Robot Operating System. O'Reilly Books, 415, USA.

Radcliffe J, Cox J, Bulanon DM (2018). Machine Vision for Orchard Navigation. Computers in Industry, 98: 165-171. doi:10.1016/j.compind.2018.03.008

Reid J, Searcy S (1986). Detecting Crop Rows Using the Hough Transform. American Society of Agricultural Engineers. Microfiche collection.

Reid J, Zhang Q, Noguchi N, Dickson M (2000). Agricultural Automatic Guidance Research in North America. Computers and Electronics in Agriculture, 25(1-2): 155-167. doi:10.1016/s0168-1699(99)00061-7

Reid J (2011). The Impact of Mechanization on Agriculture. National Academy of Engineering's The Bridge, Issue on Agriculture and Information Technology, 41(3): 22-29.

Reif JH (1979). Complexity of The Mover's Problem and Generalizations. 20th Annual Symposium on Foundations of Computer Science, 421-427, USA doi:10.1109/sfcs.1979.10

Reina G, Milella A, Rouveure R, Nielsen M, Worst R, Blas MR (2015). Ambient Awareness for Agricultural Robotic Vehicles. Biosystems Engineering, *146*: 114-132. doi:10.1016/j.biosystemseng.2015.12.010

Rekleitis IM (2004) A Particle Filter Tutorial for Mobile Robot Localization. Centre for Intelligent Machines, McGill University, Tech. Rep. TR-CIM-04-02

Richey C (1959). Automatic Pilot for Farm Tractors. Agricultural Engineering, 40(2):78–79.

Rüegg J, Viret O, Raisigl U (1999). Adaptation of Spray Dosage in Stone-Fruit Orchards on The Basis of Tree Row Volume. EPPO Bulletin, 29(1-2): 103-110. doi:10.1111/j.1365-2338.1999.tb00803.x

Sadeghi-Niaraki A, Varshosaz M, Kim K, Jung JJ (2011). Real World Representation of a Road Network For Route Planning In GIS. Expert Systems with Applications, 38(10): 11999-12008. doi:10.1016/j.eswa.2010.12.123

Saffiotti A (1997). The Uses of Fuzzy Logic in Autonomous Robot Navigation. Soft Computing - A Fusion of Foundations, Methodologies and Applications, 1(4): 180-197. doi:10.1007/s005000050020

Salichs M, Armingol J, Moreno L, Escalera AD (1999). Localization System for Mobile Robots in Indoor Environments. Integrated Computer-Aided Engineering, 6(4): 303-318. doi:10.3233/ica-1999-6404

Schiele B, Crowley J (1994). A Comparison of Position Estimation Techniques Using Occupancy Grids. Proceedings of the 1994 IEEE International Conference on Robotics and Automation, 1628-1634, USA. doi:10.1109/robot.1994.351357

Schulz D, Burgard W, Fox D, Cremers A (2001). Tracking Multiple Moving Objects with A Mobile Robot. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1: 371-377, USA. doi:10.1109/cvpr.2001.990499

Schulz D, Burgard W, Fox D, Cremers A (2001a). Tracking multiple moving targets with a mobile robot using particle filters and statistical data association. Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation, 1665-1670, South Korea. doi:10.1109/robot.2001.932850

Schwartz JT (1987). Planning, Geometry, and Complexity of Robot Motion. Ablex, 352 p, USA.

Shaffer G, Gonzalez J, Stentz A (1993). Comparison of Two Range-Based Pose Estimators for A Mobile Robot. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.32.365&rep=rep1&type=pdf (erişim tarihi: 05.11.2018) . doi:10.1117/12.143791

Shih J, Lin T (2013). Reconstruction of 3D Natural Field Scenes Based on SLAM Robot. IFAC Proceedings Volumes, 46(4): 104-109. doi:10.3182/20130327-3-jp-3017.00026

Simmons R, Koenig S (1995). Probabilistic Robot Navigation in Partially Observable Environments. IJCAI'95 Proceedings, 2: 1080-1087, Canada.

Singh S, Burks TF, Lee WS (2005). Autonomous Robotic Vehicle Development for Greenhouse Spraying. Transactions of the ASAE, 48(6): 2355-2361. doi:10.13031/2013.20074

Smith T (2007). Probabilistic Planning for Robotic Exploration. Ph.D. Thesis, Carnegie Mellon University, The Robotics Institute, Pittsburgh, USA.

Smith RC, Cheeseman P (1986). On the Representation and Estimation of Spatial Uncertainty. The International Journal of Robotics Research, 5(4): 56-68. doi:10.1177/027836498600500404

Southall B, Hague T, Marchant J, Buxton B (2002). An Autonomous Crop Treatment Robot: Part I. A Kalman Filter Model for Localization and Crop/Weed Classification. The International Journal of Robotics Research, 21(1): 61-74. doi:10.1177/027836402320556485

Stoll A, Kutzbach HD (2000). Guidance of a Forage Harvester with GPS. Precision Agriculture, 2(3): 281-291. doi:10.1023/a:1011842907397

Subramanian V, Burks TF, Arroyo A (2006). Development of Machine Vision and Laser Radar Based Autonomous Vehicle Guidance Systems for Citrus Grove Navigation. Computers and Electronics in Agriculture, 53(2): 130-143. doi:10.1016/j.compag.2006.06.001

Takahashi O, Schilling R (1989). Motion Planning in a Plane Using Generalized Voronoi Diagrams. IEEE Transactions on Robotics and Automation, 5(2): 143-150. doi:10.1109/70.88035

Thrun S (1993). Exploration and Model Building in Mobile Robot Domains. IEEE International Conference on Neural Networks, 175-180, USA. doi:10.1109/icnn.1993.298552

Thrun S (1998). Learning Metric-Topological Maps for Indoor Mobile Robot Navigation. Artificial Intelligence, 99(1): 21-71. doi:10.1016/s0004-3702(97)00078-7

Thrun S (2001). A Probabilistic On-Line Mapping Algorithm for Teams of Mobile Robots. The International Journal of Robotics Research, 20(5): 335-363. doi:10.1177/02783640122067435

Thrun S, Fox D, Burgard W, Dellaert F (2001). Robust Monte Carlo Localization for Mobile Robots. Artificial Intelligence, 128(1-2): 99-141. doi:10.1016/s0004-3702(01)00069-8

Thrun, S. (2005). Probabilistic Robotics. Mit Press, 647, UK.

Tomovic A (2014). Path Planning Algorithms for The Robot Operating System. http://www.micsymposium.org/mics2014/ProceedingsMICS_2014/mics2014_submission_2.pdf (erişim tarihi: 05.11.2018)

Underwood JP, Hung C, Whelan B, Sukkarieh S (2016). Mapping Almond Orchard Canopy Volume, Flowers, Fruit and Yield Using Lidar and Vision Sensors. Computers and Electronics in Agriculture, 130: 83-96. doi:10.1016/j.compag.2016.09.014

Warren C (1993). Fast Path Planning Using Modified A* Method. Proceedings IEEE International Conference on Robotics and Automation, 662-667, USA. doi:10.1109/robot.1993.291883

Weiss G, Wetzler C, Puttkamer EV (1994). Keeping Track of Position and Orientation of Moving Indoor Systems By Correlation Of Range-Finder Scans. Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94), 595-601, Germany. doi:10.1109/iros.1994.407420

Weiss U, Biber P (2011). Plant Detection and Mapping for Agricultural Robots Using a 3D LIDAR sensor. Robotics and Autonomous Systems, 59(5): 265-273. doi:10.1016/j.robot.2011.02.011

Westling F, Underwood J, Örn S (2018). Light Interception Modelling Using Unstructured LiDAR Data in Avocado Orchards. Computers and Electronics in Agriculture, *153*: 177-187. doi:10.1016/j.compag.2018.08.020

Wilson J (2000). Guidance of Agricultural Vehicles — a historical perspective. Computers and Electronics in Agriculture, *25*(1-2): 3-9. doi:10.1016/s0168-1699(99)00052-6

Vroegindeweij BA, Blaauw SK, Ijsselmuiden JM, Henten EJ (2018). Evaluation of the Performance of PoultryBot, an Autonomous Mobile Robotic Platform for Poultry Houses. Biosystems Engineering, 174: 295-315. doi:10.1016/j.biosystemseng.2018.07.015

Yang S, Mei S, Zhang Y (2018). Detection of Maize Navigation Centerline Based on Machine Vision. IFAC-PapersOnLine, 51(17): 570-575. doi:10.1016/j.ifacol.2018.08.140

Zaman QU, Salyani M (2004). Effects of Foliage Density and Ground Speed On Ultrasonic Measurement Of Citrus Tree Volume. Applied Engineering in Agriculture, 20(2): 173-178. doi:10.13031/2013.15887

Zaman QU, Schumann AW, Hostler HK (2006). Estimation of Citrus Fruit Yield Using Ultrasonically-Sensed Tree Size. Applied Engineering in Agriculture, 22(1): 39-44. doi:10.13031/2013.20186

Zhang Q, Chen MS, Li B (2017). A visual navigation algorithm for paddy field weeding robot based on image understanding. Computers and Electronics in Agriculture, 143: 66-78. doi:10.1016/j.compag.2017.09.008

Zhang S, Wang Y, Zhu Z, Li Z, Du Y, Mao E (2018). Tractor path tracking control based on binocular vision. Information Processing in Agriculture, 5(4): 422-432. doi:10.1016/j.inpa.2018.07.003

Zheng W, Cheng L, Huang J, Dai Y, Shang C, Peng R, He Z (2016). Research on Path Planning of Family Nursing Robot Based on Robot Operating System. 2016 International Conference on Advanced Robotics and Mechatronics (ICARM), 47-52, China. doi:10.1109/icarm.2016.7606893

EK-1. Temel ROS Kullanımı

Sık Kullanılan Komut Satırı Araçları

roscore

ROS tabanlı sistemlerde çalışma için gerekli olan düğüm ve programlar. ROS düğümlerinin birbirleriyle haberleşebilmesi için roscore' un çalışıyor olması gerekir.

Kullanım:

$ roscore

roslaunch

ROS düğümlerinin lokal bilgisayarda veya uzaktan başlatmak, parametre sunucusundaki parametreleri ayarlamak için kullanılır.

Kullanım:

Örneğin turtlebot_teleop paketindeki keyboard_teleop.launch dosyası aşağıdaki gibi çalıştırılır.

$ roslaunch turtlebot_teleop keyboard_teleop.launch

rosrun

rosrun ardışıl paketlerin cd (veya roscd) komutu kullanılmadan çalıştırılmasını sağlar.

Kullanım:

$ rosrun package executable

rosnode

ROS düğümleri hakkında yayın, kayıt ve bağlantı bilgilerini almak için kullanılır.

Kullanım:

- $ rosnode list (çalışan tüm düğümleri listeler)
- $ rosnode info /düğüm_ismi (Belirtilen düğümle ilgili bilgileri sıralar)
- $ rosnode kill /düğüm_ismi (Belirtilen düğümü sonlandırır)

rosmsg/rossrv

rosmsg/rossrv komutları mesaj ve servislerin veri yapılarını gösterir

Kullanım:

rosmsg show /mesaj_ismi

rossrv show /servis_ismi

rostopic

Yayıncı, kayıtçı, yayın hızı ve mesajlar gibi ROS topikleri hakkında bilgileri ekrana yansıtmak için kullanılır.

Kullanım:

$ rostopic echo (Mesajı ekrana yazdırır)

$ rostopic info (Topikle ilgili bilgileri ekrana yazdırır)

$ rostopic list (Aktif topiklerle ilgili tüm bilgileri ekrana yazdırır)

$ rostopic pub (Belirtilen topiğe bir veri yayınlar)

rosparam

Parametre sunucusundaki YAML-encoded dosyalarını kullanarak ROS parametrelerini almak ve ayarlamak için kullanılır.

Kullanım:

$ rosparam set (Parametreyi ayarlar)

$ rosparam get (Parametreyi alır)

$ rosparam delete (Parametreyi siler)

$ rosparam list (Parametre isimlerini listeler)

rosservice

ROS servislerini sorgulamak ve listelemek için kullanılır.

Kullanım:

$ rosservice list (Aktif servisleri listeler)

$ rosservice type (Servis tipini ekrana yazdırır)

$ rosservice info (Servisle ilgili bilgi alır)

$ rosservice call (Verilen argümanlarla servisi çağırır)

Dosya Sistemi Komut Satırı Araçları

rospack

Paketleri incelemek için kullanılır

Kullanım:

$ rospack find [paket]

roscd

Dizini bir küme veya paket için değiştirmek amacıyla kullanılır

Kullanım:

$ roscd [paket [/altdizin]

roswtf

ROS sisteminde veya başlatma dosyasında bulunan hata ve uyarıları göstermek için kullanılır

Kullanım:

$ roswtf

catkin_create_package

Yeni bir paket oluşturmak için kullanılır

Kullanım:

$ catkin_create_pkg [paket ismi]

Veri Kaydı için Komut Satırı Araçları

rosbag

ROS topiklerini kaydetmek ve yeniden oynatmak için kullanılır.

rosbag record komutu dahil edilen tüm topiklerin içeriklerini kaydeder.

Kullanım:

$ rosbag record -a (tüm topikleri kaydeder)

$ rosbag record topic1 topic2 (sadece bu iki topiği kaydeder)

rosbag play komutu bir veya birden fazla bag dosyasını alarak oynatır.

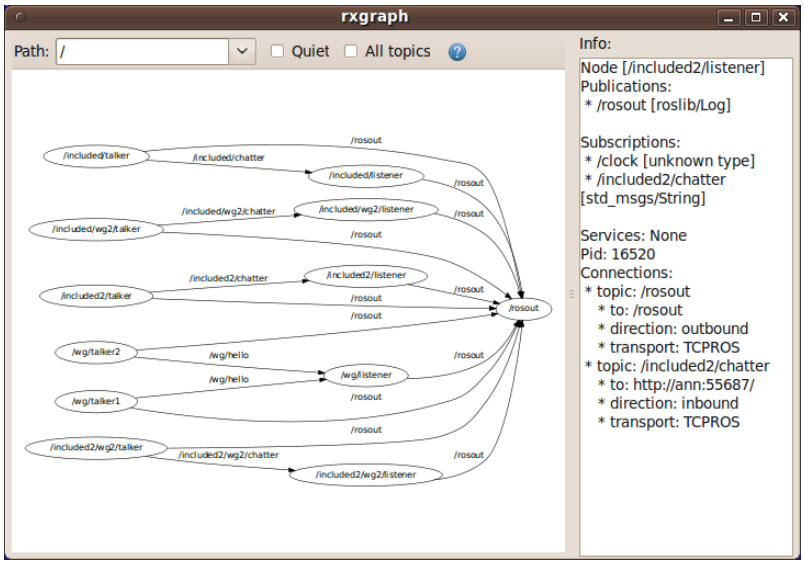
Kullanım:

rosbag play test.bag (test.bag dosyasını oynatır)

Grafik Araçları

rx_graph

Çalışan ROS düğümlerin ve bunları birbirine bağlayan topiklerin grafiğini görmek için kullanılır.



Kullanım:

$ rqt_graph

view_frames

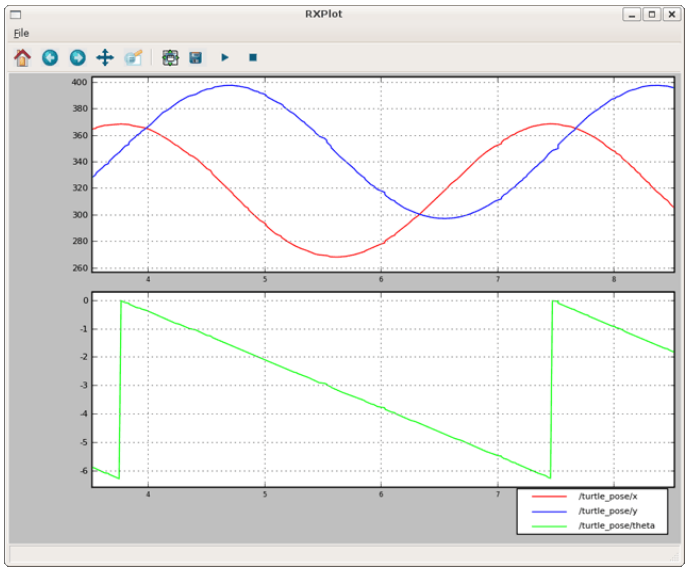
3 boyutlu koordinat dönüşümlerini görselleştirmek için kullanılır

Kullanım:

$ rosrun tf view_frames

rx_plot

Bir ROS topiğine ait bir veya birden fazla veri noktasının grafiğini çizmek için kullanılır



Kullanım:

$ rqt_plot

TEŞEKKÜR

Danışman hocam Prof.Dr. İlker Hüseyin ÇELEN' e, çalışmam süresince kıymetli düşünce, öneri, zaman ve emeğini bana harcadığı için teşekkürü bir borç bilirim. Tez çalışmamda fikir ve emeklerini benden esirgemeyen, desteklerini her zaman hissettiğim Tekirdağ N.K.Ü. Ziraat Fakültesi, Biyosistem Mühendisliği'ndeki değerli hocalarıma, sevgilimi aileme ayrı ayrı teşekkür ederim.

ÖZGEÇMİŞ

İstanbul İline bağlı olan Büyükçekmece ilçesinde 26/02/1984 tarihinde doğdu. İlk ve orta öğrenimi İstanbul'da tamamladı. Sakarya Üniversitesi Mühendislik Fakültesi Elektrik-Elektronik Mühendisliği bölümünde lisans eğitimine 2002 yılında başladı. Lisans eğitimini 2006 yılında tamamlayarak Elektrik-Elektronik Mühendisi olarak mezun oldu. 2006-2009 yılları arasında Türksan Yüksek Teknolojiler Ltd. Şti. - İstanbul' da Teknik Servis mühendisi olarak görev yaptı. 2009 yılı içerisinde Namık Kemal Üniversitesi Fen Bilimleri Enstitüsü Tarım Makinaları Anabilim Dalı'nda yüksek lisans eğitimine başladı ve 2013 yılında eğitimini tamamladı. 2013 yılında Namık Kemal Üniversitesi Fen Bilimleri Enstitüsü Tarımsal Makina Sistemleri Anabilim Dalı'nda doktora eğitimine başladı. Namık Kemal Üniversitesi Ziraat Fakültesi Biyosistem Mühendisliği bölümüne 2010 yılında Araştırma Görevlisi olarak atanmayı hak kazandı. Halen aynı bölümde görevine devam etmektedir**.**