T.C.
KARABÜK ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
MEKATRONİK MÜHENDİSLİĞİ

GEZGİN ROBOTLARDA EŞ ZAMANLI KONUM BELİRLEME VE HARİTALAMA

BİTİRME TEZİ

Hazırlayanlar
İsmail Demir 2011010225069

Tez Danışmanı
Yrd.Doç.Dr. AYTÜL BOZKURT

KARABÜK-2016

İÇİNDEKİLER

Sayfa

KABUL VE ONAY

İsmail DEMİR...tarafından hazırlanan " Mobil Robotlarda Engel Tanıma ve Kablosuz İletişim ile Kontrol" başlıklı bu tezin Lisans Bitirme Tezi olarak uygun olduğunu onaylarım. 09/06/2016

Tez Danışmanı

Yrd. Doç. Dr. Aytül BOZKURT

Bu çalışma, jürimiz tarafından oy birliği / oy çokluğu ile Mekatronik Mühendisliği Anabilim Dalında Lisans Bitirme Tezi olarak kabul edilmiştir. 09/06/2016

Tez Jürisi

Başkan: Yrd. Doç. Dr. İbrahim ÇAYIROĞLU .....

Üye : Yrd.Doç. Dr. Aytül BOZKURT .....

Üye : ..... Arş. Gör. Emre ÜKER ....

KBÜ Mühendislik Fakültesi, Mekatronik Mühendisliği Mezuniyet Komisyonu ve Bölüm Başkanlığı bu tezi Lisans Bitirme Tezi olarak onamıştır. 11/06/2016

Yrd.Doç.Dr. İbrahim ÇAYIROĞLU

Mekatronik Müh. Bölüm Bşk.

ÖNSÖZ

Robotlar, günlük hayatımızda sosyal, endüstriyel, askeri veya başka alanlarda kendilerini kanıtladılar ve büyük bir hızla buna devam ediyorlar. Bu noktada, bu hızlı ilerleyişin getirdiği bazı problemler göze çarpıyor. Robotların bazı kullanım alanlarında ihtiyaç halinde konumunun bilinmesi ve bulunduğu ortam hakkında bilgilerin anlık alınması gerekebiliyor. Eş zamanlı konum belirleme ve haritalama gezgin robotlar için, bilinmeyen bir çevre ve çevre içerisindeki mevcut bir yerin haritasını çıkarma veya verilen haritaya göre konum bilgisi belirleme ve güncel haritalamadır. Ve maalesef bu bir tek komut veya basit bir işlemle gerçekleştirilecek bir durum değildir. Bu çalışmada bu problemin kaynağın ve problemin çözümü için geliştirilen ve uygulanan bazı yöntemler incelenecektir.

İsmail DEMİR

BÖLÜM 1

1. GİRİŞ

Bilgi ve iletişim teknolojilerindeki ilerlemeler ile iç ortamlarda konumlama gezgin robotlar, akıllı evler, sağlık sektörü, alışveriş merkezleri vb. sayısız uygulamalarda ihtiyaç haline gelmeye başlamıştır.

Robotlar günümüzde yerlerini almaktalar ve almaya devam edeceklerinin de garantisini veriyor gibiler. Bu durum neticesinde, insanların bulunduğu ortamlar ile robotların etkileşimi kaçınılmaz bir hal alıyor. Robotun içinde bulunduğu ortamla etkileşime girmesinin altyapısında, ortamı tanıması ve kendinin bu ortamın neresinde olduğunu belirlemesi, konumlandırabilmesi yatıyor. Bunun için de öncelikle içinde bulunduğu ortamı tanıması yani haritasını çıkarabilmesi gerekmektedir.[1]

Bu çalışmada ele alacağımız Gezgin Robotlarda Eş Zamanlı Konum Belirleme Ve Haritalama, gezgin robotların iki problemi üzerinde durur. Bunlardan ilki konumun belirlenmesidir.

Robot bulunduğu ortamda "Ben Şu An Neredeyim?" sorusunu kendisine soruyormuş gibi bir cevap aramaya çalışır. Bir diğer problem ise robotun bulduğu ortamın haritalandırılmasıdır. Robot bu durumda kendine "Bulunduğum Yer Neye Benziyor?" diye soruyormuş gibi cevap aramaya çalışır. Ortamı haritalayabilmesi için robotun sensör ölçümleri alması gerekir. Konum belirleyebilmek için haritaya, haritalayabilmek için de konum bilgisine ihtiyaç vardır.

Dolayısıyla yukardaki iki sorun ve iki soruyu tek bir çatı altında topladığımızda robot kendisine "Şu Ana Kadar Topladığım Bilgiler İle Bulunduğum Ortamın Neresindeyim? " sorusunu sorup bu doğrultuda bir cevap arayacaktır.[2]

BÖLÜM 2

2. EŞ ZAMANLI KONUM BELİRLEME VE HARİTALAMA

Eş Zamanlı Konum Belirleme Ve Haritalama, Smith, Self ve Cheeseman'ın önceki çalışmalarına dayanarak, Hugh Durrant-Whyte ve John J. Leonard tarafından geliştirilmiştir. Eş Zamanlı Konum Belirleme Ve Haritalama bilinmeyen bir ortamda gezinen robotun aynı anda bulunduğu konumu haritada belirleyerek haritalayabilmesiyle ilgilidir.

Robot bulunduğu ortamda konumunu belirleyebilmesi için iki bilgi elde etmesi gereklidir. Bu ilgilerden ilki robotun kendisinden (harita bilgisine sahip olması gibi) elde ettiği bilgilerdir. İkinci bilgi ise ortam verileridir. Bu da robotun hareketi sırasında elde ettiği sürüş ve sensör verileridir. Robot bu verileri bir arada kullanarak konumunu belirleyebilir. Bu konum belirleme problemini Konum İzleme, Bütünsel Konum Belirleme ve Eş Zamanlı Konum Belirleme Ve Haritalama olarak üç madde olarak incelenir.

Robot haritayı çıkarabilmek için öncelikle konum bilgisine sahip olmalıdır. Bu bilgi bütünsel konumlandırma sisteminden harici olarak alınabileceği gibi, robotun tekerleklerindeki hareket kodlayıcılardan da alınabilir. Sonra robot bulunduğu ortamı tanımlamak için algılayıcılardan ulaştığı bilgilerle bulunduğu ortamın haritasını oluşturur. [3]

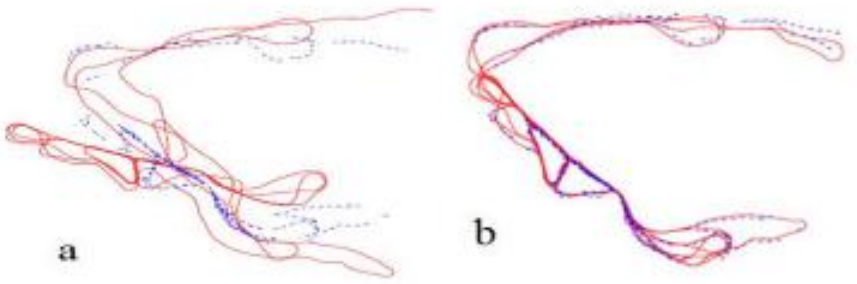
Haritalama sürecinin en belirgin özellikleri ve karşılaşılan problem aşağıda sıralanmıştır:

a) Harita çıkarma problemi, genellikle robot pozisyonu belirleme yani lokalizasyon problemi ile birlikte ele alınır. Çünkü etraftaki nesnelerin konumunu bilmek için başka bir deyişle harita çıkarabilmek için robot kendi konumunu ve buna bağlı olarak da algılayıcılarının konumunu bilmelidir; robotun kendi konumunu belirleyebilmesi için de gezindiği ortamda etrafında bulunan nesnelerin konumunu bilmelidir.[10]

b) Robot, haritasını çıkarmayı amaçladığı ortamda gezinirken dış dünyayı algılayabilmelidir. Bunun için de kamera; sonar, lazer veya kızılötesi teknolojilerini kullanan mesafe ölçerler, pusulalar veya GPS'ler kullanabilirler. Ancak algılayıcıların verisi tamamen güvenilir değildir, ölçümler gürültülü olabilir, ayrıca çoğu algılayıcının görme – algılama mesafesi kısıtlıdır. Örneğin ışık ve ses duvarlardan geçemez [11]

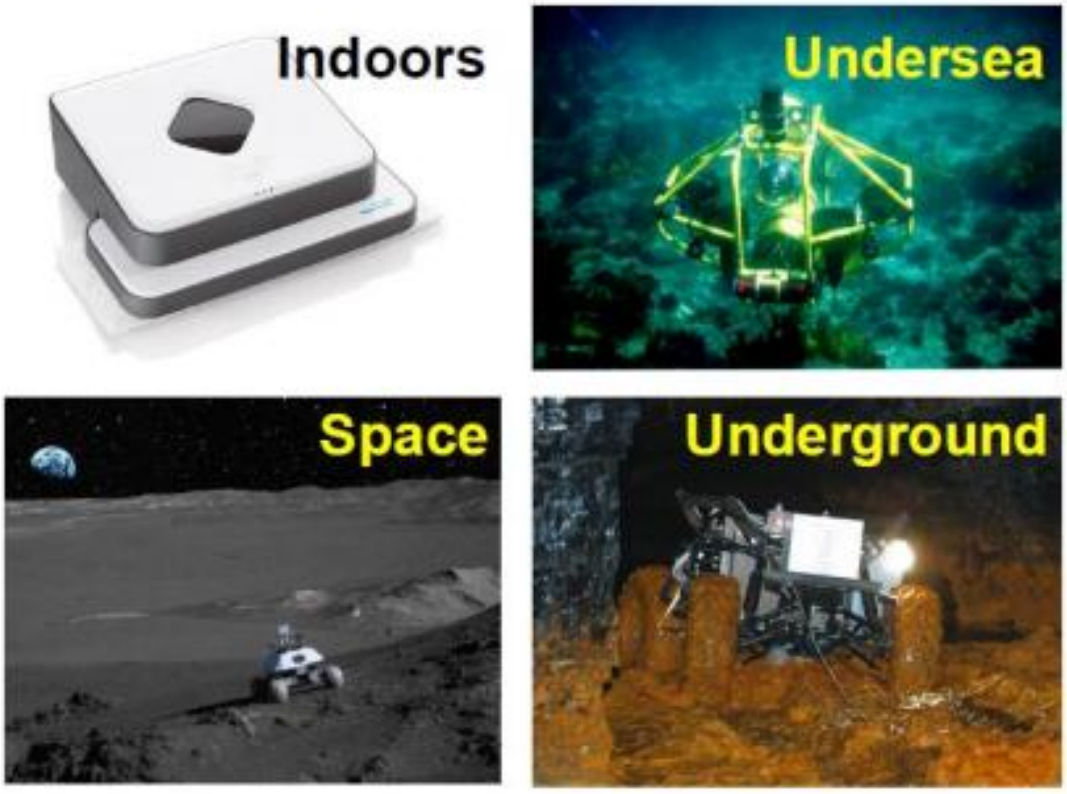
c) Haritası çıkarılacak ortamda gezinirken üretilen hareket (kontrol) komutları, farklı algılayıcı ölçümlerinin alındığı konumlarla ilgili bilgi içerdikleri için haritalamada önemlidir (Siegwart ve Nourbakhsh, 2004). Robot aldığı komuta göre hareket ederken birtakım problemler çıkabilir, dolayısıyla robotun yeni pozisyonunu belirlerken sadece kontrol işaretlerine göre verilerin kesin olduğuna inanmak doğru değildir[10]

Eş Zamanlı Konum Belirleme Ve Haritalama alanında son yıllarda çok sayıda çalışmalar olmuştur. Bilinen algoritmalar geliştirilerek optimum seviyede çalıştırılması amaçlanmakta ve ya eski bilinen algoritmaların yerine onların yetersiz kaldığı noktaları daha da aydınlatmak sebebiyle yeni algoritmalar geliştirilmektedir. Bunlara örnek olarak Kalyaev vd. (1997) tarafından yapılan çalışmada algılayıcı olarak pahalı lazer sistemi kullanılmış, bilinmeyen her hangi bir ortamda gezinen robotun algılanan bu veriler yardımıyla konumunu belirlemesi ve düzeltme yapması amaçlanmıştır. Sonuçların başarılı olmasından dolayı Rus savunma sanayisinde kullanılmıştır. Yun vd. (1998) tarafından, doğrusal özellikler kullanılarak robotun kendi konumunu belirlemesi çalışmasının yapılmasını görebiliriz. Karpov vd. (2001) yapmış olduğu çalışmada ise her hangi bir komut almadan robotun bir ortamda kendi konumunu ve ortamın haritasının çıkarılması için yeni bir çözüm geliştirilmeye çalışılmıştır. Algılayıcı verisinden En Küçük Kareler yöntemi ile doğrular uydurulmuş, doğruların köşe oluşturduğu yerler belirlenmiş ve bunun sonucunda robot kendi konumunu kesinleştirmiştir. Tardos vd. (2002) tarafından yapılan çalışmada ise algılayıcı olarak sonar algılayıcılar kullanılarak, elde edilen sonuçlara Hough Dönüşümü uygulanmıştır. Bunu sonucunda harita özellikleri doğrusal ve noktasal olarak elde edilmiştir. Thrun vd. (2005) Victoria Park' ta gezinen lazer sensorlu araba çalışmasında o güne kadar çok az kullanılmış bir yöntemle parktaki detaylar, özellikler belirlenmiştir. Odometri verisi kullanılmadan sadece doğrusal ve açısal hız kontrolünden yararlanarak araba 30 dakika parkta gezdirilmiştir. Kullanılan algoritma FastSLAM olmuştur. Parktaki detaylar bulunarak gerçek haritadan ne kadar sapma olduğunun sebepleri araştırılmış ve ortadan kaldırılmaya çalışılmıştır.



Şekil-1: Victoria Park ‘ta Gezgin robot ile Eş Zamanlı Konum Belirleme Ve Haritalama Örneği

Bailey vd. (2004) tarafından yapılmış çalışmada, FastSLAM algoritması kullanılarak haritalamanın başlıca sorunlarından biri olan ‘veri ilişkilendirme’ sorununa çözüm aranmıştır. Alınan sonuçları GKF yöntemi ile elde edilen sonuçlarla kıyasladıktan sonra işlem sayında ve işlemlerin karmaşıklık derecesindeki farklar araştırılmıştır. Bununla beraber FastSLAM algoritmalarının da kendi aralarındaki ortak ve farklı yönleri belirlenmiş kullanılan robot modeline ve ortama göre hangi algoritmanın kullanılmasının uygunluğu gözden geçirilmiştir. Beevers ve Huang (2006) tarafından yapılan çalışmada sınırlı sayıda algılayıcıya sahip bir robot ile Parçacık Filtresi kullanılarak Eş Zamanlı Konum Belirleme Ve Haritalama yapılmıştır. Diğer çalışmalarda olduğu gibi bu çalışmada da algı verileri kullanılarak elde edilen noktalardan doğrular çıkarılmıştır. Bundan önce noktalar kümelenmiş, her kümeye İteratif Uç Nokta uygulanarak gerektiği takdirde bu kümeler daha küçük kümelere ayrılmıştır. Bir eşik değer belirlendikten sonra bunun üstündeki sayıda noktası bulunan kümeler belirlenmiştir. Bu nokta kümelerinden doğrular çıkarılarak sadece doğruların uç noktaları özellik olarak alınmıştır.[20]



Şekil-2: Eş Zamanlı Konum Belirleme Ve Haritalama ‘nın
Bazı Kullanım Alanları

2.1. Konum Belirleme Modelleri

Mutlak Konum Belirleme

Mutlak konum belirleme yönteminin en yaygın biçimi, yörüngedeki uyduların bilinen konumları arasında mesafeleri kullanan bir uzay tabanlı mikrodalga konumlandırma sistemi olan Küresel Navigasyon Uydu Sistemi (GNSS, Global Navigation Satellite System)’dir. GNSS dünyanın herhangi bir yerinde, yer yüzeyine yakın bir kullanıcı için 24 saat üç boyutlu konum, hız ve zaman bilgilerini sağlayan bir sistemdir. Günümüzde mevcut iki sistem bulunmaktadır. Bunlar; GPS ve GLONASS tır. 2003 yılında Avrupa Komisyonu 2014 yılında tam olarak hizmete girecek olan Avrupa GNSS yi kurmaya karar vermiştir.

Bugün GPS sistemi en yaygın olarak kullanılan GNSS sistemidir. GPS; uzay bölümü, kontrol bölümü ve kullanıcılardan oluşmaktadır. GPS Uzay bölümü yörüngedeki en az 24 uydudan oluşur.

GPS uyduları 1,57542 (L1) GHz ve 1,22760 GHz (L2) frekanslarda iki L-band taşıyıcı frekans yayar. L1 sinyalinin, bir hassas (P) kodu ve bir de kaba (C/A) kodu içeren kodlanmış bir kod dizini vardır. L2 taşıyıcı fazı sadece askeri ve sivil yetkili kullanıcılar için şifreli P kodu içerir. Ticari GPS alıcıları L1 sinyali ve C/A kodu215 kullanmaktadır. P kod kullanıcıları yer merkezli konumlarını bir tek el uydu alıcısı ile yaklaşık 5 m hassasiyetle bulurlar.

Askerler tarafından şifrelenmiş olan P kod sadece yetkilendirilmiş kullanıcılar tarafından kullanılabilmektedir. Bunun sonucu olarak sivil kullanıcılar P-kodları kullanamazlar. Seçici Kullanılabilirlik (SA, Selective Availability), tek GPS alıcılarının konumsal doğruluğunu % 95 düzeyinde yatayda 100 m ve düşeyde 156 m olarak sınırlamaktaydı. 2 Mayıs 2000 tarihinde SA kapatılmış, % 95 seviyesinde yatay doğruluk yaklaşık 5-25 m ye ulaşmıştır. [15,16,17,18,19]

Diferansiyel GPS (DGPS)

Diferansiyel konum belirleme (DGPS, Differential GPS) ya sonradan işlenerek ya da gerçek zamanlı yapılabilir. Birincisi daha az pahalı ve basittir, ikincisi bir radyo bağlantısına gereksinim duyulduğu için daha karmaşık ve pahalıdır. Diferansiyel düzeltmeler ölçüm düzeltmeleri veya pozisyon düzeltmeleri şeklinde olabilir. Referans istasyonu olarak kullanılacak bir noktanın her iki yaklaşımda da koordinatları bilinmelidir ve mevcut olmalıdır. Referans noktasından uzaklaşıldıkça hata daha fazla olacak ve diferansiyel teknikler kullanılarak konum belirleme doğruluğu daha az olacaktır.

Post processing diferansiyel konum belirleme uygulamasında, sabit ve gezicide yapılan gözlemler konum belirleme hesapları için bir bilgisayarda birleştirilir. Gezici ve sabitin gözlem zamanları eşleştirilerek eşzamanlı gözlemler oluşturulur ve algoritmalar kullanılarak uygun diferansiyel düzeltmeler hesaplanır.

DGPS dinamik durumlarda birkaç metre hassasiyetle, hatta sabit iken daha iyi konumsal veri sağlayabilir. Bu gelişme bir veri kaynağı olarak GPS üzerinde önemli bir etkiye sahiptir: GPS, yalnızca tekne ve uçak için kaba navigasyon için kullanılmamalıdır. GPS çok hassas bir ölçekte konum belirleme kapasitesi ile evrensel bir ölçme sistemi yeteneğine sahiptir.

Gerçek zamanlı DGPS konumu bilinen bir kontrol noktasını baz istasyonu olarak kullanır ve onun bilinen konumu ile hesaplanan konumu arasında farkları sürekli hesaplar. Daha sonra radyo vericileri ve diğer uydular üzerinden bu düzeltme bilgilerini gönderir. Kullanıcının GPS alıcısı, uydular tarafından dağıtılan düzeltme bilgisini okuyabilen bir radyo alıcısı veya GPS alıcısına sahip olmaya ihtiyacı vardır ve 1 veya 2 saniye önce kaydedilen konum bilgilerini düzeltmelidir. Sistem 2 ile 5 m doğrulukla gerçek zamanlı konum bilgisi sağlayabilir. Gerçek zamanlı GPS sistemlerinin bir dezavantajı, referans noktasında bir ölçüm yapıldığında ölçü anındaki düzeltmeyi geziciye göndermek için geçen zaman gecikmesidir. Bu hata tabii ki post-processing modda olmayacaktır. GNSS sisteminin performans gereksinimlerini geliştirmek için kullanıcılara diferansiyel düzeltme iletimi sağlayacak çeşitli uydu tabanlı destek sistemleri geliştirilmiştir. Bunlar geniş alanları kapsamakta ve veri alıcı genellikle GPS alıcısının içerisine entegre edilmektedir. Bu tip güçlendirme sistemi Avrupa Yerdurağan Navigasyon Kaplama Servisini
(EGNOS, Geostationary Navigation Overlay Service) içerir. Bu Avrupa'nın uydu navigasyon sistemine ilk adımıdır. EGNOS askeri kontrollü GPS ve GLONASS sistemlerini tamamlayacaktır. EGNOS düzeltme verileri mevcut hizmetlerin yaklaşık 20 m olan doğruluğunu 5 m den daha aza düşürecektir. EGNOS'un kapsama alanı tüm Avrupa devletlerini içine almakta ancak bu kapsam Avrupalı olmayan bölgeleri alacak şekilde artttırılabilmektedir. 2009 başından itibaren EGNOS altyapısı Galileo sistemine entegre edilecektir. Kuzey Amerika benzer bir sistem olan WAAS (Wide Area Augmentation System, Geniş Alan Güçlendirme Sistemi) a sahip iken, Japonya' ise MSAS (Multifunctional Transport Satellite Space-based Augmentation System "Çok Fonksiyonlu Ulaşım Uydu Tabanlı Uzay Güçlendirme Sistemi) ne sahiptir.[15,16,17,18,19]

Gerçek Zamanlı Kinematik GPS (RTK GPS)

Taşıyıcı faz GPS konum belirleme santimetre ölçeğinde konumsal doğruluk sağlayabilen bir yöntemdir. RTK GPS özellikle ölçme uygulamalarında kullanılmaktadır. Bu alıcılar kod tabanlı el GPS alıcıları ile karşılaştırıldığında çok daha karmaşık ve pahalıdır. RTK GPS

tekniği hızla pratik ve bilimsel amaçlarla kullanılan santimetre doğrulukla ölçme ve navigasyon yapma aracı haline gelmiştir. [15,16,17,18,19]

Uydusallar (Pseudolites)

GPS uydusallar veya sahte (pseudo) uydular lokal alanlarda GPS'in aynı sinyallerini ileten yer tabanlı vericileridir. GPS uydusallar üç ana yoldan GPS'le konum belirlemeye yardımcı olabilirler [21]. Birinci olarak, GPS doğal uydu kapsamasının yetersiz olduğunda çeşitli ilave mesafe kaynakları sağlayarak GPS uydu kümesini arttırmak için kullanılabilir. İkincisi, uydusallar hassas konum belirleme için taşıyıcı faz diferansiyel GPS (CDGPS) kullanıldığında, taşıyıcı faz belirsizlik çözünürlüğünde yardımcı olarak kullanılabilir. Üçüncü, uydusallar tamamen GPS uydu kümesi yerine kullanılabilir. Bu genellikle, GPS'in kapalı alanlarda konum belirlemede kullanımı veya dünya dışı yerlerdeki (örneğin, Mars'ta bir araç konumlandırma) konum belirlemelerinde söz konusudur [21].

Bağıl Konum Belirleme

"*Parekete hesabı*" belirli bir zaman süresi boyunca bilinen rota ve hız bilgileri ile önceki konum bilgileri kullanılarak aracın mevcut yerinin belirlenmesi için matematiksel bir işlemdir. Göreli konum belirleme genel olarak "parekete hesabını ifade eder çünkü her zaman bilinen pozisyonlardan önceki pozisyonu ve bağıllığı ifade eder. Odometri kapalı alanlarda mobil robotlarda çok yaygın olarak kullanılan bağıl konum belirleme yöntemidir. Biriken hata özellikleri nedeniyle sadece kısa süreli navigasyon işlerinde kullanılır. O tekerlek dönme ve/veya zaman içinde dümenleme yönlendirme ile artan hareket bilgilerinin entegrasyonudur. Bu alanda birçok makale ve yürütülen kapsamlı araştırmaların mevcut olduğu görülmektedir [22].

Açık alan uygulamalarında konum bilgisinde ortaya çıkan genellikle kabul edilmez hatalara katkı yapan birçok problem meydana gelir. Dış mekan uygulamalarında hatalar çoğunlukla tekerlek kayması ve 3 boyutlu konum belirleme nedeniyle ortaya çıkar. *Ataletsel ölçüm birimleri* (IMU, Inertial Measuring Units) ivme ve dönmeyi belirlemek için jiroskoplar ve ivmeölçerler kullanır. Bu ölçüm verilerine dayanarak yön ve mevcut konum, ölçmenin başladığı konumu bilinen bir önceki noktaya göre hesaplanabilir. Dış mekan uygulamaları için IMU'lar genellikle aracın dünyanın manyetik alanına göre yönlendirmesini ölçen elektronik pusulalar ile güçlendirilir. IMU'lar entegre olduklarında yüksek konum belirleme hassasiyetine ulaşırlar ancak hala pahalıdırlar.

Referanslı Konum Belirleme Yer işaretleri sensör tarafından algılanabilecek belirgin özelliklerdir. İşaretler geometrik şekilli olabilir ve desenler gibi ek bilgiler içerebilir. Yer işaretleri bir aracın kendini bağıl olarak lokalize edebilmesi için sabit ve bilinen bir konuma sahiptir. Bir aracın navigasyon için yer işaretlerini kullanmasından önce, simgesel özelliklerinin bilinmesi ve kontrolör (denetleyici) hafızasına depolanması gerekir. Hesaplanan araç konumunun doğruluğu, yer işaretlerinin güvenilir işaretler olmasına ve nasıl tanınacağına ve lokalizasyon için nasıl kullanılacağına bağlıdır. Yapay ve doğal işaretler olmak üzere iki türlü yer işareti vardır. Terimler aşağıdaki gibi tanımlanmaktadır [23].

•*Doğal yer işaretleri* ortamda zaten varolan ve mevcut başka işlevlerine ilave olarak araç navigasyonu işlevleri de olan işaretlerdir [24]

•*Yapay yer işaretleri* tek amacı konum belirlemek için özel olarak tasarlanmış obje veya işaretlerdir. Açık doğal yer işareti navigasyonunda, algılama ve sensör girdileri arasından karakteristik özelliklerin eşleşmesi önemli bir sorundur. Bu görev için en uygun sensör türü bilgisayarla görmedir. Bugün ürün sıraları, tarımda navigasyon amaçları için tipik bilgisayar görme tabanlı doğal yer işaretleridir. Bugün diğer tanınan bir ortak özellik, lazer tarayıcı tabanlı bir yakınlık sensörü tarafından algılanan ürün kenarlarıdır. Bu özelliklere dayanan otomatik işlemler, sıra arası çapalama ve biçerdöverin dümenlemesidir. Yapay yer işareti konum belirleme, çalışma alanının etrafına yerleştirilmiş üç veya daha fazla dedektörlü basit bir yapılandırma kullanır. Araca monte edilmiş lazer yatay süpürüldüğünde ve ışık algılandığında konum belirleme sistemine bildirilir. Nirengi kullanarak aracın konumu tespit edilebilir. Bu sistemin bir dezavantajı araç ve dedektörler arasında iletişim bağlantısı gerektirmesidir. Araca monteli lazeri esas alan sistemler, engebeli arazide kullanıldığında hedefleri kaçırma dezavantajı vardır. Alternatif olarak lazerler araziye yerleştirilir ve araç üzerine dedektör bağlanır [25].

Harita tabanlı (ya da harita eşleşmeli) konum belirleme, bilinmeyen ortamın haritasını oluşturmak için aracın kendi sensörlerini kullandığı bir tekniktir. Bu lokal harita daha sonra önceden hafızada varolan harita ile veya sisteme bağlı olan Coğrafi Bilgi Sistemi (GIS, Geografic Information System)'nden alınan harita ile karşılaştırılır. Robot, ölçülen ve önceden depolanan çevresel özellikleri eşleştirerek ortamdaki gerçek konumunu ve yönünü hesaplayabilir[26].

Sensör Birleştirme (Sensor Fusion) Tüm işletme şartları altında gerekli bilgileri sağlamak için yalnız tek bir pozisyon sensörü yeterli olmaz. Tek bir mükemmel yöntemin olmaması nedeniyle göreli veya mutlak konum belirleme kullanarak en az her gruptan gelen bir yöntemi birleştirmek tavsiye edilir. Farklı sensör türlerine dayanan bu işleme *sensörbirleştirme* denir.
Mevcut navigasyon algılayıcı entegrasyon tekniklerinin çoğu, çoklu sensör entegrasyonu için en iyi çözümlerden birini temsil eden, Kalman filtreleme yöntemine dayanır. Bir Kalman filtresi stokastik (rastgele), rekürsif, ağırlıklı ve en küçük kareler hesaplama algoritmalarını kullanan, doğrusal ve model tabanlı bir kestirimdir [27].

Olasılıksal durum kestirim yöntemi gezgin robotlar için en çok kullanılan en bilindik yöntemdir. Literatürde yer alan çözüm yöntemleri aşağıda incelenecektir.

2.1.1. Konum İzleme

Konum belirlemede ele alınan ilk problem konumun izlenme sürecinin nasıl olabileceği veya konum izlemedir. Bu durumda ilk olarak robotun içinde bulunduğu ortamın başlangıç haritası konumu ve konum bilgileri bilinerek problemin çözümüne bu bilgilerle yaklaşılır. Robotun başlangıç konumu referans alınarak, robot harekete geçirilir ve robotun hareketleri izlenir ve başlangıca göre robotun yaptığı hareketler tespit edilip robotun son konumuna ulaşılır. Doğal yer gösterici konumlarının ortamdaki yerlerinin bilindiği varsayılır. Bu yer göstericiler rotamdaki köşelerin ve kenarların bulundukları yerleri göstermektedir. Burada problemin çözülmesini sağlayan durum, izleme veya bölgesel yordamlar olarak adlandırılır.

Genel olarak konumun izlenmesinde ve kestirim probleminin çözünde GKS ‘den yararlanılır. Bu yöntemle birçok uygulamalardaki sorunlarda başarı sağlanmıştır. GKS nin tahminlerinin yetersiz kaldığı durumlarda ise ki özellikle yüksek dereceli lineer olmayan sistemlerin modelleri ile çalışıldığında, beklenen sonuçların aksine başarısız sonuçlar elde edilmiştir. Alternatifinin az olması hasebiyle birçok problem ve uygulamalarda tercih edilmektedir.

Gezgin robotlarda konum izleme problemlerinde hareket ve ölçüm için lineer olmayan durum uzay modelleri çıkarılır.

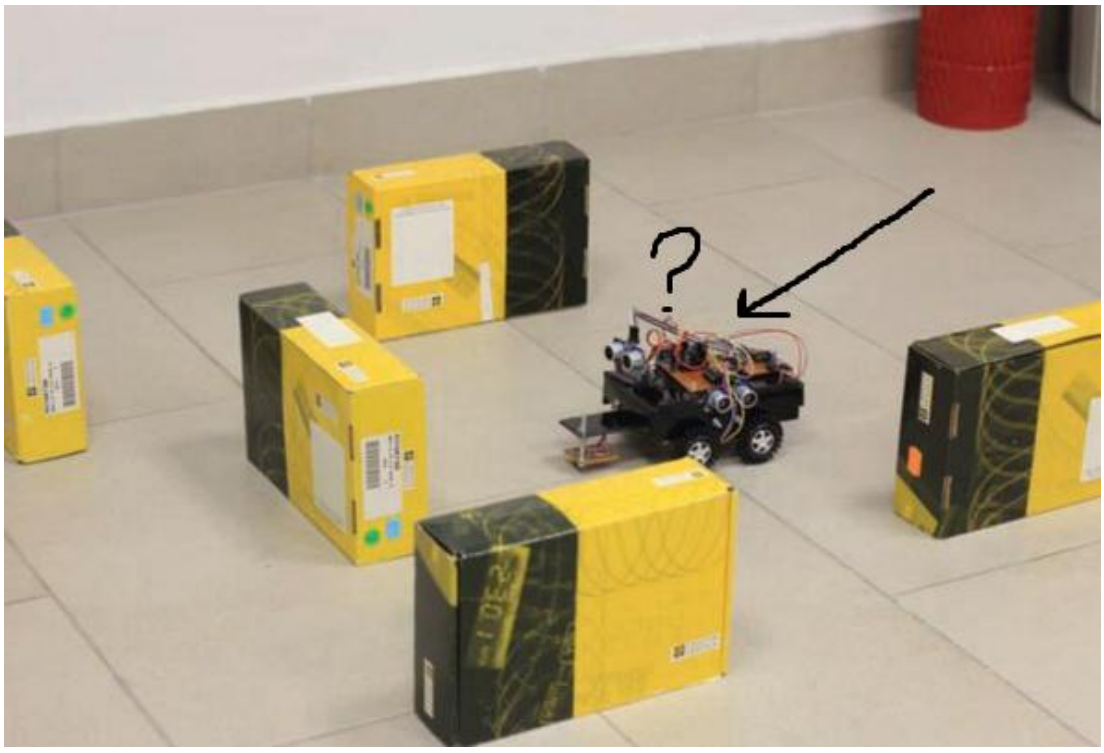
$$x_{k+1}=f(x_k,u_k)+w_k \tag{2.1}$$

$$z_k=h(x_k)+v_k \tag{2.2}$$

Modeldeki durum vektörü durum değişkenleri ile ifade edilir.

$$x_k=[x_r \quad y_r \quad \theta_r]^T \tag{2.3}$$

(x, y,) robotun bulunduğu ortam içerisindeki konumunu, Q ise yönelim açısını göstermektedir. Teker kodlayıcılarının algıladığı bilgiler denedim girdisi ile hareket modelindedir. Sistemin gürültüsü ve ölçüm gürültüsü birbirleri ile etkileşimde değillerdir. Sıfır ortalamalı Gauss beyaz gürültü olarak alınır.



Şekil-3: Konum izleme

2.1.2. Bütünsel Konum Belirleme

Konum belirleme problemine göre daha karmaşık ve zordur. Uyanma yada bütünsel konum belirleme problemi olarak literatürde yer alır. Robot bulunduğu ortam için kullanabileceği referans bir haritaya sahiptir ama bu harita üzerindeki konumunun bilgisine sahip değildir. Robot harekete başlamak için veya anlık olarak harita üzerindeki başlangıç durumunu bilmiyordur.

Bu nedenle ilk olarak robotun çözmesi gereken problem veya bulması gereken "Ben mevcut haritada hangi konumdayım" sorusunun cevabıdır. Bunu yapabilmesi içinde biraz uğraşması gerekir. Dolayısıyla bulunduğu konum hakkında oldukça farklı bilgiler elde edebilir.

Bu problemin çözümü için kullanılan yöntemler bütünsel yöntemler olarak adlandırılır. Bütünsel konum belirlemede Monte-Carlo parçacık süzgeci ile konum belirleme yöntemi geliştirilen diğer bütünsel konum belirleme yöntemlerine oranla daha etkin ve daha doğru sonuçlar vererek, daha başarılı olmuştur.

Şekil-4: Bütünsel konum Belirleme

2.1.3. Eş Zamanlı Konum Belirleme ve Haritalama

Bu problemde robot bulunduğu ortam hakkında herhangi bir ön bilgiye (ortamın haritası veya ortam haritası içindeki konumu) sahip değildir.

Bundan önce bahsetmeye çalıştığımız diğer yöntemlerdeki harita üzerindeki yer gösterici konumları bu problemde önceden bilinmez. Dolayısıyla robot ortam içerisindeki konumunu belirlemeye çalışırken ortam içerisindeki konumunu da bulmaya, belirlemeye çalışacaktır.

Eş zamanlı konum belirleme ve haritalama problemi için genel olarak, konum izleme yönteminde tanımlanan lineer olmayan durum uzay modelleri aynen kullanılarak GKS ile sonuca ulaşılmaya çalışılır.

$$x_{k+1}=f(x_k,u_k)+w_k \tag{2.4}$$

$$z_k=h(x_k)+v_k \tag{2.5}$$

Ancak konum izleme yöntemine göre eş zamanlı konum belirleme ve haritalamadaki en büyük fark vektörünün boyutundaki değişikliktir. Konum belireme yönteminde referans olarak kullanılacak haritanın önceden bilindiği ve doğru olduğu kabul edilir. Bu yüzden durum vektörü sadece robotun değişkenlerini içerir.

Eş zamanlı konum belirleme ve haritalama yönteminde, haritalama ve konumlama eş zamanlı olarak yapıldığı için durum vektörü hem robotun konum bilgisini hem de ortamdaki yer göstericilerin konum bilgilerini kapsar. Tanımlanan vektör ve matris boyutları dinamiktir çünkü her yani yer gösterici tespit edildiğinde durum vektörü içerisine tanımlanacak, durum vektörü ve durum hara matrisleri genişleyecektir.

$$x_k = \begin{bmatrix} x_r & y_r & \theta_r & m_1 & m_2 & . & . & . & m_N \end{bmatrix}^T \tag{2.6}$$

Genişlemiş durum kovaryans matrisi de;

$$P_{k,k} = \begin{bmatrix} P_{rr} & P_{r1} & \cdots & P_{rN} \\ P_{1r} & P_{11} & \cdots & P_{1N} \\ \vdots & \vdots & \ddots & \vdots \\ P_{Nr} & P_{N1} & \cdots & P_{NN} \end{bmatrix} \quad (2.7)$$

Şeklinde olacaktır.

Matristeki parametreler sırasıyla robotu-robota, robotun-yer göstericilere ve yer göstericilerin-yer göstericilere göre karşılıklı ilişkilerini ifade eder. Büyük boyuttaki dinamik matrislerden dolayı algoritmaların hesaplanma karışıklığı burada çözülmesi önemli olan bir diğer problemdir.

Eş zamanlı konum belirleme ve haritalama problemi için literatürde bulunan ve kullanılan bir diğer çözüm yöntemi, Rao-Blackwellized parçacık süzgeci ile tanımlanır[4]. FastSLAM algoritması, Rao-Blackwellized parçacık süzgecinin en çok kullanılan uygulamasıdır. FastSLAM algoritması konum izleme kestirimi yapmak için birden fazla olarak, birkaç parçacık süzgecini birlikte kullanır. Bu algoritmada haritalama problemi ise haritadaki her karakteristiğin bir birinden bağımsız olarak ele alınmasıyla çözülür. FastSLAM algoritması harita karakteristik konumlarının kestirimi için birbirinden bağımsız GKS'ler kullanır. Her bir karakteristik için ağırlıklandırılmış bağımsız bir GKS kullanılması daha düşük boyutlarda matrisler ile harita kestirimi yapılmasına olanak sağlamaktadır. Bu sayede yer gösterici sayısı N'e bağlı olarak, standart GKS'deki hesaplama karmaşıklığı O(N)3 'den, FastSLAM algoritmasıileO(logN)'edüşürülmüştür.

FastSLAM yönteminin diğer bir avantajı ise doğrusal olmayan robot sistem modelleri ile kullanılabilir olmasıdır. GKS yönteminde doğrusal olmayan sistem modelleri doğrusallaştırılarak kullanılmaktadır. Özellikle yüksek dereceli doğrusal olmayan sistem modellerinin doğrusallaştırılmasıyla, GKS doğru kestirim yapamayabilir. Bu nedenle yüksek dereceli doğrusal olmayan sistem modelleri ile çalışıldığında parçacık süzgeci ve FastSLAM algoritması ilk tercih edilen yöntemdir[4].

Yukarıdaki saydığımız avantajları nedeniyle FastSLAM gerçek zamanlı uygulamalarda daha çok tercihe dilmiştir.[4]

2.2. Haritalama

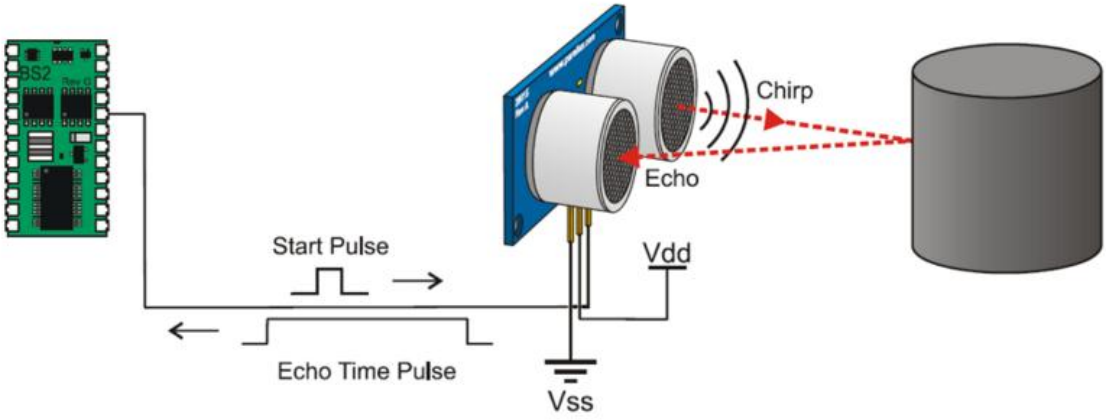
Ortamın haritasının çıkarılması için gezgin robot uygulamalarında kamera, lazer veya sonar sensörler kullanılmaktadır.

Kamerayla haritalama sonar ve lazer e göre daha ileri düzeylerde yapılabilmektedir. Sayısal görüntü işleme algoritmalarının çok fazla işlem karışıklığı meydana getirmesinden dolayı gerçek zamanlı uygulamalarda hızlı ve maliyetli sistemlerde kullanılır. Ortamı algılayabilmek için gereken ışığın yetersiz olması ve işlem karmaşıklığı en büyük dezavantajlarıdır.

Lazer sensörlerin çalışma prensibi yüksek dalga boylarındaki ışıklara duyarlılıkla açıklanabilir. Sonar sensörlere göre lazer sensörler nesne algılama ve mesafe belirlemede çok daha etkili sonuçlar verirler. Işığın saydam yüzeylerden geçmesiyle sağlıklı sonuçlar elde edilememesi ve maliyetinin yüksek oluşu en büyük dezavantajlarındandır.

Sonar sensörler birinci dünya savaşından beri kullanılmaya devam etmektedir. Sonar sensörlerin çalışma prensibi sesin ortamda yayılması ile açıklanabilir. Bir kaynaktan gönderilen ses sinyali hedef bir noktaya çarpıp gerş yansır ve tekrardan bu yansıyan dalga algılayıcılar ile yakalanır. Dönen ses sinyalinin şiddeti ve hızındaki değişiklikler gibi bazı parametrelerden bilgiler elde edilir. Bu sistemde birçok ses sinyali gönderilir ve yankıların alındığı süreler ölçülür, Uçuş zamanı olarak diye adlandırılır. Bu uçuş zamanının mesafelerin ölçümünde kullanılmasında sesin hızının değişmediği veya çevredeki ısı değişikliğine bağlı olarak az bir düzeydeki değişikliği imal edilir. Sonar sensörlerin maliyetleri düşüktür, lazer sensörlere nazaran ekonomiktir. Nesnelerin yüzeyindeki yansımalar en büyük dezavantajıdır. Yansımanın şekli gelen sinyalin açısı, çarptığı yüzeyin şekli gibi durumlarla ilgilidir.

Gerçek hayatta sonar sensörler, sesin yayılması özelliğini kullandıklarından, ölçüm sonuçlarını üçgen şeklinde bir alandan elde edeceklerdir. Bundan dolayı sonar bilgileri merkezi olarak alma yöntemi tutarlı olmayacaktır. Bundan dolayı bazı ihmallerle sonuçlar yeniden modellenir. Maliyetinin diğerlerine nazaran dala düşük olmasından dolayı, birçok dezavantajına rağmen robot uygulamalarında olukça sık kullanılır.

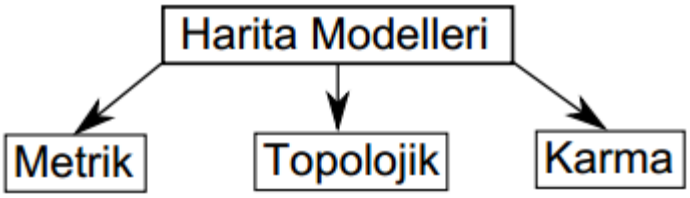


Şekil-5: Sonar Sensör Sistemi

2.2.1. Haritalama Modelleri

Harita modelleri, robotun uzamsal tanımlamayı nasıl yapacağı ile ilintilidir. Uzamsal gösterim, hem robotun bulunduğu konumdaki ortamı, hem de o anki algısal alanının dışında olup, önceden ziyaret ettiği veya bir başka kaynaktan alınmış olan

farklı ortam bilgisini içerir [29]. Geliştirilen modeller, dünya merkezli veya robot merkezli olabilmektedir [30]. Dünya merkezli haritalar küresel koordinat sistemi içinde oluşturulmaktadır. Robot merkezli haritalar ise ölçüm uzayı içinde tanımlanmaktadır. Geliştirilen modeller, Şekilde gösterildiği üzere üç grupta sınıflandırılabilir: i) Metrik modeller, ii) Topolojik modeller ve iii) Karma modeller. Ancak, topolojik modellerde çeşitli seviyelerde geometrik bilgi içerebildiğinden, metrik modeller ile topolojik modeller arasındaki fark her zaman tam olarak açık olmayabilir.



Şekil-6: Harita modelleri

Metrik Modeller

Metrik modellerde, Kartezyen koordinatlarında bir referans yön sistemi temel alınır ve tüm bilgiler bu sistem içine konumlandırılır. Bu yaklaşımlar, aynı zamanda allosentrik veya dünya merkezli olarak da tanımlanırlar [31].En çok kullanılan metrik harita, doluluk kafesleridir [32]. Burada, ortam kafeslere bölünerek gösterilir [30]. Kafes tabanlı haritalama yöntemleri, ortamın modellemesini sağlayarak değişik algılayıcılardan gelen verilerin tümleştirilmesinde kolaylık sağlayıp, yüksek çözünürlükle iyi bir sonuç vermektedirler. Ancak, geniş ortamlarda, sabit kafes boyutu nedeniyle, çok fazla sayıda kafes oluşabileceği için, bunların oluşturulması ve planlamada kullanılması hesapsal açıdan oldukça maliyetli olabilir [33]. Dolayısı ile, büyük ölçekli haritalarda döngüleri kapama, haritaları saklama, mevcut yeri önceki yerlerle karşılaştırma ve robotu ilk yeri hakkında bilgilendirmeden genel konumlandırma konularında problemler yaşanmaktadır[34].

Yüksek hesapsal yükü azaltmak için, alt harita tabanlı modeller öne sürülmüştür. Alt haritalar, küçük yerel haritalardır ve bir araya getirilerek büyük harita sistemleri oluşturabilmektedirler. Genellikle çevreyi eşit alt haritalara bölerek çevrenin topolojisini göz ardı ederler [35],[36]. Bu hususlara ek olarak, doluluk kafeslerinin yapısı, kameradan alınan imgeler gibi zengin bilgilerin kaydedilmesine olanak vermemektedir. Buna yönelik olarak, haritaların bilgi içerikleri, çeşitli özniteliklerin yerleri bulunarak ve haritada gösterilerek zenginleştirilmektedir [37],[38]. Burada önemli bir konu, haritaya hangi nirengi noktalarının koyulacağının belirlenmesidir. Yapay işaretler doğal olanlara göre daha kararlıdır. Ancak, yapay işaretler için ortama müdahale gerekir, bu yüzden de daha çok doğal işaretler tercih edilir. Doğal nirengi noktası olarak köşeleri, kapıları ve farklı binaları içeren pek çok farklı öznitelik kullanılmıştır. Ne var ki gerçek zamanlı kullanımda, hangi özniteliklerin kullanılacağı konusundaki çalışmalar devam etmektedir.

B. Topolojik Modeller

Metrik haritalara alternatif olarak topolojik haritalar önerilmiştir [39]. Topolojik haritalar, kaydedilen yerlerin allotetik veya robot merkezli karakterizasyonlarıdır [40]. Topolojik

haritada uzamsal bilgi, bitişiklik diyagramları kullanan ve patikalarla bağlanan yerlerin derlemesi olan bir çizge olarak tanımlanır. Bu çizgenin düğümleri farklı yerleri temsil ederken, kenarlar yerler arasındaki bağıl yönelimleri veya patika uzunluğu gibi bitişiklik ilişkileri temsil ederler. Örnek olarak, [41]'de düğümler haritadaki belli başlı özel noktaları (köşe, kapı, koridor sonu v.s.) temsil ederken, kenarlar, bu düğümler arasındaki geçişlere karşılık gelmektedirler. Burada en önemli husus, yer tanımlarının nasıl yapılacağıdır [29]. Yer tanımları, bağlamsal tabanlı ve görünüş tabanlı olarak iki ana kategoriye ayrılmaktadır. Bağlamsal tabanlı yaklaşımlar gelen görsel veriyi doğrudan kodlayan yaklaşımlardır. Kullanılan yer tanımları içinde 9×9, 15 × 15 piksel gibi bağıl büyük imge parçaları [42], Ayrık Fourier Dönüşüm [43], öz imgelere parçalama [44], ana bileşenler analizi [45], dalgacık imge dönüşümü [46] gibi değişen seviyelerde uzamsal entegrasyonlu filtre yanıtlarını kullanılmaktadır. Örneğin, bir çalışmada harita farklı imge kareleri arasında bağıl konum şebekesi olarak tanımlanır [47]. Farklı bir yaklaşımda ise, imge alt imgelere bölündükten sonra elde edilen yönlü tekdüze örüntülerin histogramları betimleyici olarak kullanılmaktadır[48].

Diğer yaklaşım olan görünüş tabanlı yaklaşımlarda ise, ilk önce gelen görsel veriden bazı öznitelikler elde edilir ve bu öznitelikler ortamın tanımlanmasında kullanılır [49]. Örneğin, iç mekanların haritalanmasında, tipik olarak düğümler koridor gibi birleşme yerlerini temsil ederken, kenarlar ise bir birleşmeden diğerine olan patikalardır [50]. Bu tip haritalama yöntemlerinde kullanılan özniteliklerden bazıları, köşeler [51], SIFT [38], SURF [52] ve çeşitli filtrelerdir [53]. Örneğin, yer tanımları, [54]'de SIFT öznitelikleri, [55]'da SURF öznitelikleri kullanılarak oluşturulmaktadır. Bu tür öznitelikler, çok kullanılmakla beraber her ortamda istenen başarımı gösterememektedir. Bu soruna bir çözüm getirmek amacıyla, öznitelikleri farklı dönüşümler ile birlikte kullanarak, daha dayanıklı ve tıkız tanımlamaların elde edilmesi hedeflenmiştir. Örneğin, kelime çantası yaklaşımında, bir imgedeki ilginç noktalar bulunarak SIFT öznitelikleri ile yüksek boyutlu bir vektör uzayında tanımlandıktan sonra, sayısal olarak nicelenerek bir görsel kelime ile ifade edilir [56]. Bu niceleme sayesinde her bir imge görsel kelimeler histogramı ile gösterilir ve imge eşleme bu histogramlar temel alınarak yapılır [57]. Parmak izinde ise her bir imgedeki ilginç noktalar çıkartılarak farklı bir harf ile gösterilir ve imge bu harflerin dizisi olarak betimlenir [58]. İki

farklı imgenin benzerliği ise bu harf dizisinin karşılaştırılması ile hesaplanır. Eğer ortamda küçük de girişimler varsa, bundan imgenin sadece bazı noktaları etkileneceğinden, tanımlamada ufak değişimler olacaktır. Dolayısı ile, veritabanındaki arama veya karşılaştırma çok daha dayanıklı olacaktır. Buna yönelik geliştirilen sözlük ağacı, sözcükleri hiyerarşik olarak organize eder [59]. Bir başka yaklaşımda ise, tüm öznitelikleri kullanmak yerine, çok daha küçük bir altkümenin kullanıldığı iskelet modeli önerilmiştir [60]. Ne var ki, görsel sözlüklerin öğretilmesinde, özniteliklerin fazla sayıda olması veya bu sözlükleri oluştururken kullanılan yöntemin görsel bozulmalara karşı dayanıklılığın sağlanması gibi belli zorluklar vardır [61]. Bunlara ek olarak, tüm bu yaklaşımlar, imgenin yapısal bilgisini kaybetme sorunuyla karşı karşıyadır [62]. Yer tanımlarının uzamsal bilgi içermesi ile bilgi içeriklerinin zenginleşeceği kesin olduğundan, bunu dikkate alan çeşitli çalışmalar yapılmıştır. Bunlardan ilki, imgeleri bir küre üzerine izdüşünümünün yapıldığı benmerkezci algılama kürelerdir [63]. Benzer şekilde, küreyi bir veritabanı yapısı olarak kullanarak verileri saklayan yaklaşımlar önerilmiştir [64]. Tanımlamalar, imgelere küresel bir izdüşüm uygulandıktan sonra, küresel harmonik katsayıları ile yapılarak daha da tıkız hale getirilmişlerdir[65]. Farklı bir yaklaşım olan baloncuk hafıza modelinde ise, görsel öznitelikler ve aralarındaki uzamsal ilişkiler robot merkezli olarak birlikte kodlanmaktadırlar[66]. Esasen topolojik temelli olmakla beraber, öznitelikler arasındaki geometrik ilişkiler küresel uzayda kodlandığından, karma bir özelliğe de sahiptir. Baloncuk uzayında ise, baloncuk hafıza modeli, farklı robot konumlarına ve farklı öznitelikleri tanımlayacak şekilde genişletilmektedir [67]. Sonuç olarak, ortak kullanımına karşı, topolojik haritaların tanımına ve nasıl oluşturulduklarına dair bir görüş birliği olmayıp, düğümler ile kenarların anlamları kullanılan öznitelikler veya uygulamaya göre değişebilmektedir [39].

C. Karma Modeller

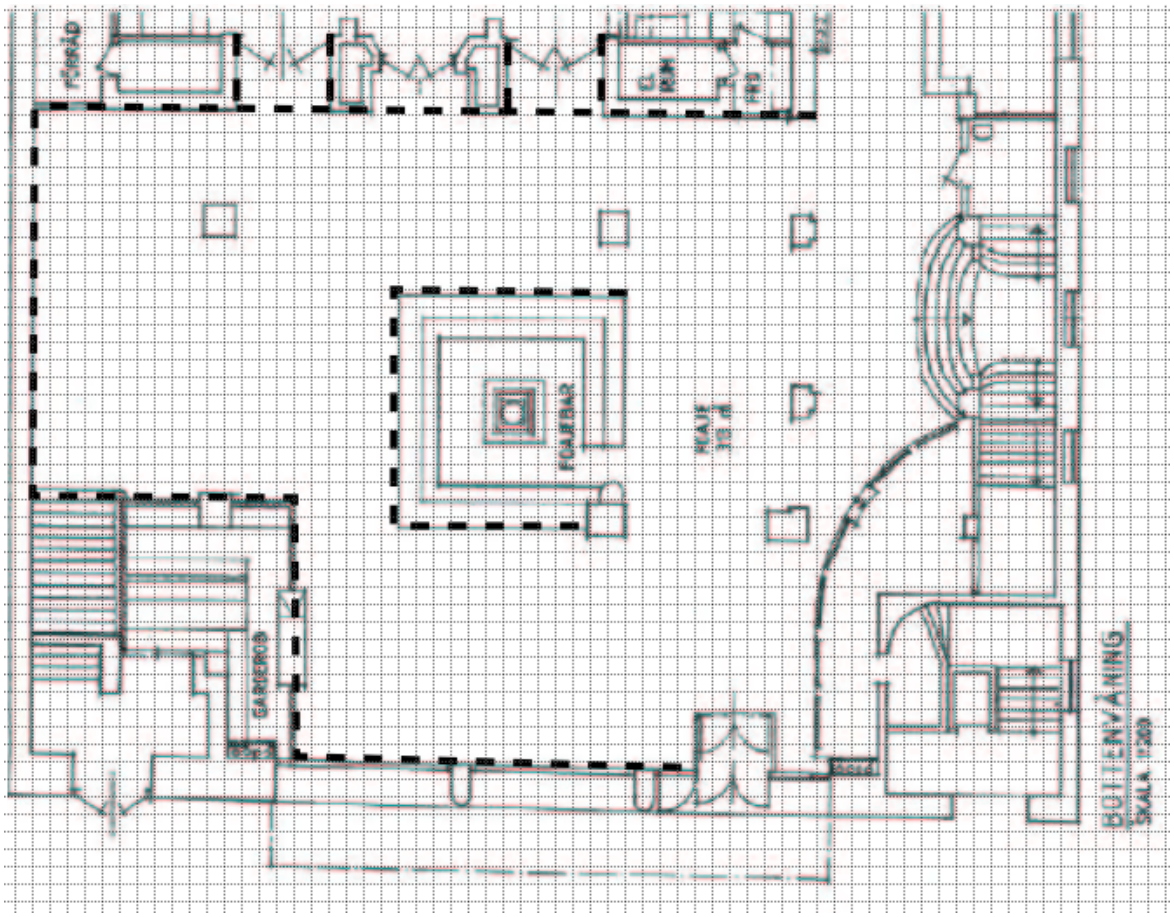
Metrik ve topolojik gösterimlerin karakter olarak birbirinden oldukça farklı olduğu göz önünde bulundurulmalıdır. Metrik haritalar, algısal sınırlar içindeki yapıyı belirtik olarak yakalarken, topolojik haritalar geniş alanın yapısını anlatır. Genel olarak metrik harita, oluşturulduğu her bölgede geometrik olarak çok detaylıyken topolojik haritada çevrenin detaylarının eksik olduğunu söyleyebiliriz [30]. Bu sorunları çözmek için küçük ölçekte metrik gösterimin, büyük ölçekte topolojik haritanın kullanıldığı bir melez harita önerilmiştir [29]. Bu yaklaşımda, yerel metrik haritalar doluluk kafesi tabanlıdır. Topolojik haritalar ise düğüm ve kenarlardan oluşan çizgelerdir [68]. Her bir düğüm robot tarafından

görülen bir yeri ifade eder ve içinde o bölgeye ait metrik bir yerel harita barındırır. Bu yaklaşımdaki önemli bir sorun, bu bölünmenin tam olarak nasıl yapılacağıdır. Bunun için farklı yöntemler önerilmiştir. Bu yöntemler, esasen çizge temelli yaklaşımlardır. Örneğin, alt seviyede bir metrik haritadan, izgel topaklama kullanılarak [33] veya çizge bölütleme uygulanarak, üst seviyede görsel tabanlı bir topolojik harita oluşturulmaktadır [69]. Ancak metrik haritalar büyüdükçe bu haritaların oluşturulması gittikçe zorlaşmaktadır. Buna alternatif olarak, ortamın metrik doluluk kafeslerini boş alanın ve ortamın bağlantı yapısının daha tıkız bir gösterimini sağlayan Voronoli diyagramları önerilmiştir [70]. Ancak pratik uygulamada, bu diyagramların çıkarımları kolay değildir.

Genel olarak robot uygulamalarında "Izgara Tabanlı Harita" ve "Öznitelik Tabanlı Harita" olmak üzere iki harita gösterimi vardır.[3]

2.2.1.1 Izgara Tabanlı Haritalama

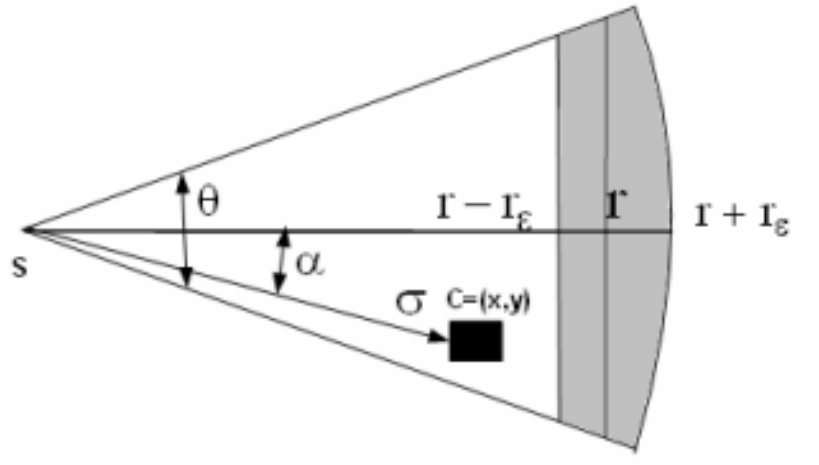
Haritalama yapılacak ortamın belirli büyüklükteki parçalara bölünmesi ile gerçekleştirilen bir uygulamadır. Bölünme yapılarak elde edilen her bir parça kendine özgü bilgiler barındırır. Bu bilgiler genel olarak olasılıksal veya parçanın dolu, boş veya daha tanımlanmamış olduğu hakkındadır. Bu haritanın oluşabilmesi gezgin robotun çevreyi dolaşması ve sırada sensörler vasıtasıyla elde ettiği sonuç ve bilgilerle ilgilidir.



Şekil-7: Haritanın Eşit Parçalara Bölünmesi

Ortanım ızgara tabanlı haritasını çıkarmak için kullanılan en yaygın yöntem “Izgara doluluk tekniğidir” [5]. Izgara doluluk tekniği ilk olarak Elfes (1987) tarafından Bayesian olasılık teoremi kullanılarak ızgara hücrelerinin dolu veya boş olma olasılıklarının hesaplanması ile ortaya çıkmıştır. Bu çalışmada ses ötesi algılayıcılar kullanılarak yöntemin doğruluğu ispat edilmiştir. Bu yöntem başlangıçta boş veya belirsiz olarak tanımlanan hücreler ile oluşturulmuş haritanın, sonar ölçümler kullanılarak Bayesian olasılık teorileri ile güncellenmesi prensibine dayanmaktadır. Hücre (C) güncellemelerinin yapılabilmesi için gerekli olan sensör karakteristikleri ve diğer veriler aşağıda belirtilmiş olup sonar ölçümler aşağıdaki şekilde gösterildiği gibi birer yay olarak modellenmiştir.

- **r** sonar algılayıcıya dönen uzaklık mesafesi
- **r_{min}** sonar algılayıcı ile ölçülebilen en kısa mesafe
- **r_ε** ölçüm sapması
- **θ** sonar hüzme açısı
- **α** SC çizgisi ile ışının ana ekseni arasındaki açı
- **$S=(x_s,y_s)$** sonar sensör konumu
- **σ** $C=(x,y)$ ile $S=(x_s,y_s)$ noktaları arasındaki mesafe



Şekil-8 Sonar ölçüm yay şekli

Taranan parçaların doluluğu veya boş olma olasılığı iki farklı ölçüm ile hesaplanır. P_E, hücrenin boş olma olasılığı, P_O ise hücrenin dolu olma olasılığını gösterir. Aşağıda bu ifadelerin matematiksel ifadeleri gösterilmiştir.

$$p_E(x,y) = p[\text{hücre}(x,y) \quad \text{boş}] = E_r(\sigma).E_a(\alpha) \tag{2.8}$$

E r (σ),E a (α) sırasıyla, uzaklık ve açısal bağımlılıkların olasılık yoğunluk fonksiyonlarıdır.

$$E_r(\sigma) = \begin{cases} 1-[(\sigma - r_{min})/(r - r_\varepsilon - r_{min})]^2 & \sigma \in [r_{min}, r - r_\varepsilon], \\ 0 \quad \text{Diger Durumlarda} \end{cases} \tag{2.9}$$

$$E_a(\theta) = 1-(2\alpha/\theta)^2, \qquad \alpha \in [-\frac{\theta}{2}, \frac{\theta}{2}] \tag{2.10}$$

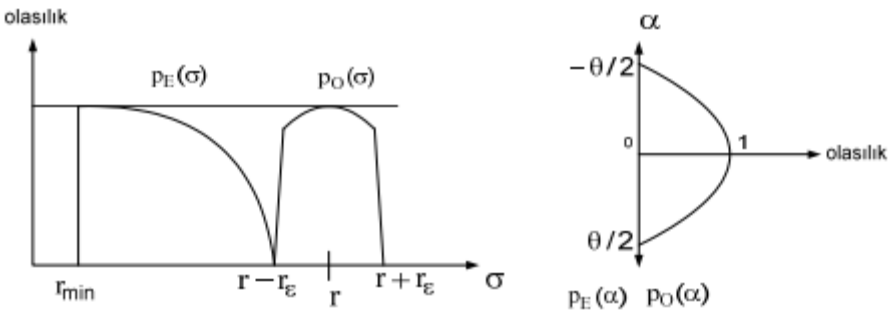
Aynı şekilde pO tanımlanırsa,

$$p_O(x,y) = p[\text{hücre}(x,y) \quad \text{dolu}] = O_r(\sigma).O_a(\alpha) \tag{2.11}$$

O r (σ),Oa (α sırasıyla, uzaklık ve açısal bağımlılıkların olasılık yoğunluk fonksiyonlarıdır.

(2.12)

$$O_a(\theta) = 1-(2\alpha/\theta)^2, \qquad \alpha \in [-\frac{\theta}{2}, \frac{\theta}{2}]$$



(2.13)

Harita oluşturma süreci, tüm hücrelere başlangıçta sıfır değerinin verilmesi ile başlar. Her bir sonarın görüş alanı içindeki [rmin,r − rε] aralığına düşen tüm hücrelerde gerekli eşitlikleri ve [r − rε,r + rε] aralığına düşen hücrelerde ise gerekli eşitlikleri kullanılarak hücrenin boş ve dolu olma olasılık değerleri hesaplanır. Daha sonra Bayesian teoremleri ile hücre olasılık değerleri güncellenir.

Hücrelerin boş olma olasılığı;

$$p_E(\text{hücre}) = p_E(\text{hücre}) + p_E(\text{ölçüm}) - p_E(\text{hücre}) * p_E(\text{ölçüm}) \quad (2.14)$$

Hücrelerin dolu olma olasılığı;

$$P_O(\text{hücre}) = p_O(\text{hücre}) + p_O(\text{ölçüm}) - p_O(\text{hücre}) * p_O(\text{ölçüm}) \quad (2.15)$$

Eşitlikleri ile güncellenir.

Böylece ortam haritası her yeni sonar ölçüm için ilgili hücrelerin olasılıksal değerlerinin güncellenmesiyle oluşturulur. Güncellenmiş pE ve pO olasılık değerleri [-1,1] aralığında değişir. Bu değerler [0,1] aralığını düşecek şekilde dönüştürme işlemi uygulanır. Daha sonra belirlenen eşik seviyesi altında kalan hücreler boş, eşik seviyesinin üstünde kalan hücreler dolu olarak yorumlanır [6].

Bu yöntemin önemli olarak iki dezavantajı vardır.

-Bellek İhtiyacı

Haritanın büyüklüğüne bağlı olarak kaplayacağı alan artar.

-Hesaplama Karmaşıklığı

Izgara güncellemesi yapılabilmesi için, hücre değerlerinin her ölçüm sonunda yeniden hesaplanması gerekmektedir. Bu süreci hızlandırmak adına, bazı ilintili varsayımlar yapılmasına karşın, her bir hücre üzerindeki cebirsel hesaplamalar nedeniyle hesaplama karmaşıklığı bu yöntem için önemli bir dezavantajıdır. Izgara tabanlı haritalar ise, konum izleme problemlerinin çözümünde sıkça tercih edilir [7]

2.2.1.2. Öznitelik Tabanlı Haritalama

Öznitelik tabanlı haritalamada haritalanan ortamın içindeki nesnelerin şekli hakkın bilgi edinilir ve bu şekillerin ortamdaki konumlarına göre ortamın betimlenmesi ile yapılır. Robot belirlenmiş olan yer göstericilere göre konumunu belirler. Bu kullanılan yer göstericiler yapay ve doğal diye olarak ikiye ayrılırlar.

2.2.1.2.1.Yapay İşaretçiler

İşaretçi yerini belli etmek için sinyali gönderiyorsa aktiftir, eğer herhangi bir sinyal göndermiyorsa pasiftir.

Aktif olan yer göstericiler gezgin robota konum bilgisi gönderirler. GPS'ler açık alanlarda en yaygın kullanılan ve en çok işe yarayan, başarılı sonuçlar alınan işaretçidir. Uydudan gönderilen radyo sinyallerinin havadan iletilip algılanmasıyla ilgilidir. Kapalı ortamlarda pek randımanlı değildir.

Robot tarafından algılanan işaretçilere pasif işaretçi denir. Robot un içindeki ortama yerleştirilmelidir. Robot un sensörleri algılanabilecek herhangi bir nesneyi algılayabilirler, bu pasif işaretçiler robota konumunu bildirir.

2.2.1.2.2. Doğal İşaretçiler

Gezgin robotun bulunduğu ortam içinde zaten var olan nesnelerdir. Robot bu nesneleri anlayarak, sınıflayarak ve konumlarını tespit ederek haritalama yapar. Kapalı alanlar için kapı, duvarlar, masa, kenarlar ve köşeler doğal işaretçilere örnek olarak verilebilir. Ortamın iki boyutlu olarak tanımlandığı durumlarda, yer gösterici tabanlı haritalar genelde düz çizgi ve noktalardan oluşur. Çizgiler, ortamdaki düzlemleri; noktalar ortamdaki kenarları ve köşeleri belirtir.

2.2.1.3. FastSLAM Yöntemi İle Haritalama

FastSLAM algoritması parçacık filtresi ile GKF algoritmasının bileşkesinden oluşmuştur. Bu yöntemle haritalama zamanı, robot konumlarına ve harita durumuna bağlı olarak öncül ihtimalleri çarpım şeklinde gösteriyoruz. Bunu Parçacık filtrelerinin çarpımı gibi de düşünebiliriz. Bu çarpım, her bir parçacık için ayrı olasılık fonksiyonu kuruluyor ve bunların bileşkesi gibi gösteriliyor. Geri kalan bütün durum güncellemesi zamanı haritadaki detayları algılamak için GKF kullanılıyor. FastSLAM algoritmasını en önemli üstünlüğü, veri ilişkilendirme kararlarında, parçacıklar üzerinden işlem yapmasıdır (Montemerlo, 2003). Diğer bir üstün özelliği ise FastSLAM algoritmasında PF kullanıldığından ve bu filtreleme yönteminde hareket modelinin doğrusal olmayabilmesidir ki, bu özellik onu GKF' den de seviyece yukarıya taşımaktadır. Çünkü GKF işlemleri zamanı kullanılan yaklaşık modelden daha iyi sonuçlar vermektedir.

2.2.1.3.1. FastSLAM 1.0 algoritması

FASTSLAM algoritmaları posterior' leri robotun yolu ve ya konumu olan p(s/z,u,n) üzerinden hesaplıyor. Gerekli formüldeki ikinci çarpan olan Nadet detay çarpanındaki detaylar ise GKF ile hesaplanıyor.

FASTSLAM' da parçacıklar

	Robot konumu	1. detay	2. detay	...	N. detay
1. Parçacık	$x\ y\ \theta$	μ_1, Σ_1	μ_2, Σ_2	...	μ_N, Σ_N
2. Parçacık	$x\ y\ \theta$	μ_1, Σ_1	μ_2, Σ_2	...	μ_N, Σ_N
...	...	...	...	...	...
M. Parçacık	$x\ y\ \theta$	μ_1, Σ_1	μ_2, Σ_2	...	μ_N, Σ_N

Şekil-9 Fast Slamda Parçacıklar

Yukarıdaki Çizelgede M adet parçacıktan oluşan FASTSLAM görüyoruz. Her bir parçacık için eşitlik yazabiliriz.

$$S_t^{[m]} = \langle s^{t,[m]}, \mu_{1,t}^{[m]}, \Sigma_{1,t}^{[m]}, \dots, \mu_{N,t}^{[m]}, \Sigma_{N,t}^{[m]} \rangle \qquad (2.16)$$

Filtreleme ise anındaki posterior'dan zamanındakini, parçacık kümesinden yeni kümesini oluşturarak hesaplamaktır. Bu işlemlerle senkronize olarak yeni kontrol ve ölçüm bilgisini (x' e bağlı olarak) oluşturuluyor. Bütün bunlar aşağıdaki adımlarla gerçekleştiriliyor.
Sampling (Sadeleştirme – Yol, konum posterioru'u yeni konumları sadeleştirerek genişlendirme) :

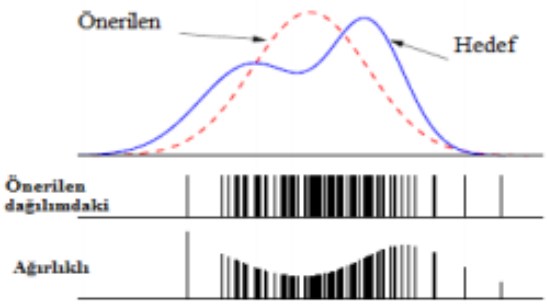
Updating (Güncelleme - Elde olunmuş harita detaylarının güncellenmesi) : Bu adımda bulunan her detay için ortalama değer ve kovaryans değerlerinin güncelleştirilmesi yapılacaktır. Yani, güncelleme aşaması anındaki fark edilmiş gürültü harita detayı mı değimli sorusunun cevabını araştırıyoruz.

Başka bir eşitlikte;

$$p(\theta_{n_t} \mid s^t, n^t, z^t) \;=\; \eta\, p(z_t \mid s_t, \theta_{n_t}, n_t)\; p(\theta_{n_t} \mid s^{t-1}, n^{t-1}, z^{t-1}) \qquad (2.17)$$

eşitliğinden yararlanarak GKF aşamalarını bir-bir gerçekleştirilmektedir. Resampling: Birinci asım olan Sampling' de konumları ölçüm değeri dikkate alınmadan kontrol değerlerine göre oluşturuluyor. Böylelikle örtüşmezlikler meydana çıkıyor. Bu örtüşmezlikleri aradan kaldırmak için ise Resampling adımı gerçekleştirilmektedir. FASTSLAM' da önerilen dağılım ' den bağımsızdır. Ama hedef dağılım değil. Parçacıklara ağırlık değerleri atayarak ve

sonra bu değerlere göre tekraren oluşturulan örnek parçacıklar aslında hedef dağılıma yakınsamış oluyor. Her bir parçacığın ağırlık değeri o parçacığın 'önemlilik faktörü' oluyor.



Hedef ve önerilen dağılımlar ve ağırlıkların histogramlarla gösterilmesi (Thrun vd., 2005)

Şekil-10

Şekilde önerilen ve hedef dağılımlar grafiklerle, önerile ve ağırlık değerleri bulunmuş parçacıklar histogramlarla gösterilmiştir. Resampling aşaması aslında hedef ve önerilen dağılımlar arasındaki fark ortaya çıkarmaktadır.

$$w_t^{[m]} = \frac{hedef\ dağılım}{önerilen\ dağılım} = \frac{p(s^{t,[m]} | z^t, u^t, n^t)}{p(s^{t,[m]} | z^{t-1}, u^t, n^{t-1})} = \eta\, p(z_t \mid s^{t,[m]}, z^{t-1}, n^t) \qquad (2.18)$$

W değerlerinin hesaplanması zamanı ise Taylor sırası kullanılarak doğrusallaştırma işlemi yapıldıktan sonra ,N(x; u,Σ) değerli Gauss dağılımlı yaklaşık değer, 2. aşama olan güncelleme adımında hesaplanıyor. Bu üç aşamadan oluşan işlemlerin sonucunda bunu diyebiliriz ki, zaman ve hafıza gereksinimlerinin hiçbiri anına bağlı değil.

2.2.1.3.2. FastSLAM 2.0 algoritması

2.0 algoritması genellik daha önceki FASTSLAM algoritmasına benziyor. Ama en önemli farklılığı, önerilen dağılım göz önünde bulundurarak hesaplanmasıdır. Bu da FASTSLAM

1.0’ daki sorunları aradan kaldırıyor. FASTSLAM 1.0’ da Sampling aşamasında konumları belirlemek için kontrol verisini, Resampling aşamasında ise ölçüm verisini kullanıyor. Robot sensorlarının kontrol verilerine az bağlı olduğu durumlarda ise sorunlarla karşılaşılıyor. Önerilen dağılım fonksiyonunun oluşturduğu çok sayıda örneklerden yalnız çok az bir kısmı 27 yüksek benzerlik göstermektedir (Bailey vd., 2004, Perdomo, 2009). Ona göre de önerilen dağılımın içine sensor verilerini de eklersek daha verimli bir çalışma yapmış oluruz.

2.2.1.3.3. Veri İlişkilendirme Problemi

Sensor ölçümleri yardımıyla bulunan harita özellikleri oluşturulan haritadaki detaylarla kıyaslandığı zaman farklılıklar meydana çıkmaktadır. Bu problem veri ilişkilendirme problemi olarak bilinmekte ve EZKBH meselesinin en karmaşık problemidir (Durrant vd., 2006). Her iki FASTSLAM algoritmasının en büyük kısıtlaması bilinen veri ilişkilendirme üzerinden işlem yapmalarıdır. Gerçek dünyanın özellikleri belirsiz olduğundan bu bölümde n değişkenlerinin, kendi aralarında haberleşmesinin bilinmeyen olduğunu varsayarak anlatmaya çalışalım. Veri ilişkilendirmeni kısaca şöyle tanımlayabiliriz: t anında n değişkenini mümkün verilere dayanarak belirlemek. Veri ilişkilendirmenin en eski çözümü ise n’i sensor ölçümlerinin maksimum bezerlik gösterdiği için seçmektir. Yani

$$\hat{n}_t = \text{argmax}\, p(z_t \mid n_t, \hat{n}^{t-1}, s^t, z^{t-1}, u^{t-1}) \tag{2.19}$$

Böyle bir tahmin ‘maksimum olabilirlik’ adlandırılıyor. FASTSLAM algoritmalarında ise her bir parçacık için n_t veri ilişkilendirme değişkenleri kümesi tanımlanmaktadır. Sensor ölçüm dağılımı belirlenmiş bir eşik değer etrafında toplanmışsa bu kümeye yeni detay özelliği eklenerek küme doldurulmaktadır.

3. YER GÖSTERİCİ ÇIKARMA TEKNİKLERİ

Öznitelik tabanlı haritalama, EKBH uygulamalarında daha çok tercih edilen haritalama tekniğidir. EKBH'nin yapılabilmesi için gezinim sırasında ortamdaki özniteliklerin çıkartılması ve çıkartılan bu özniteliklerin birer yer gösterici olarak kullanılması gerekir. EKBH için iki boyutlu düzlemde harita oluşturulması ve haritadaki özniteliklerin çıkartılması, robot algılayıcılarının (sonar, lazer mesafe ölçer) ortam üzerinden elde edecekleri verilerin işlenmesi ile olur. Birden fazla algılayıcı bir arada kullanılarak daha maliyetli (lazer mesafe ölçer+kamere) çözümlerde literatürde yer almaktadır [8]. Fakat mesafe sensörleri de kullanılarak haritalanacak ortamıdaki kenar köşeler kullanılarak öz nitelik tabanlı haritalama oluşturulabilir.

Bu durumda ortamı tanımlayan bu özniteliklerin mesafe ölçerler ile tespit edilmesi ve sınıflandırılması gerekir. Sonar algılayıcı tabanlı çalışmalarda Hough Dönüşümü (doğru belirleme) veya Üçgenleme Tabanlı Birleşim (kenar belirleme) teknikleri ile çözüme gidilir[9]

3.1. Hough Dönüşüm Metodu

Hough dönüşümü, P.V.C Hough (Hough1962) tarafından geliştirilen,

$$g(x,c)=0,\ x \in X,\ c \in C \tag{3.1}$$

eşitliği kullanılarak, D ∈ X veri dizisinden eğri tanımlama yöntemidir. Burada x noktasal konum verilerinden oluşan bir vektör, c ise vektör katsayılarıdır. Hough dönüşüm tekniğinin en genel uygulaması nokta veri dizisinden doğru parçası tanımlama olarak karşımıza çıkar.[3]

$$D=\{(x_1,y_1),(x_2,y_2),(x_3,y_3),\ldots(x_n,y_n)\} \tag{3.2}$$

Bu durumda g(x,c) fonksiyonu (3.11) eşitliğindeki gibi tanımlanır.

$$x\cos(\varphi)+y\sin(\varphi)-\rho=0,\quad (x,y)\in\Re,\quad (\varphi,\rho)\in C \tag{3.3}$$

Eğer amaç doğru parçası bulmak ise, C için parametre uzayı (Hough uzayı) aşağıdaki gibidir.

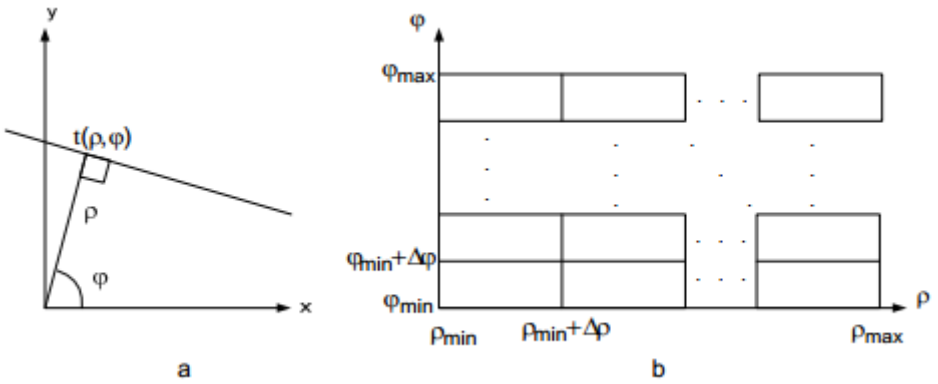
$$C=\{(\rho,\varphi)\ \big|\rho\in\Re,\ -\frac{\pi}{2}\le\varphi\le\frac{\pi}{2}\} \tag{3.4}$$

Bir çok uygulamada ilgili parametre uzayı sınırlandırılmıştır. Bu durumda C yeniden tanımlanacak olursa,

$$C=\{(\rho,\varphi)\ \big|\ \rho_{min}\le\rho\le p_{max},\ \varphi_{min}\le\varphi\le\varphi_{max}\} \tag{3.5}$$

gibi olur.

Hough algoritması bir çeşit oylama yöntemidir. Bu oylama kesikli parametrik uzayda yani Hough uzayında yapılır. Hough uzayı tüm olası özniteliklerin konumlarını gösterir. Hough uzayında en çok oy alan hücre ilgili özniteliğin konumu ile ilgili polar koordinatları verir.[3]



(3.6)

Şekilde kartezyen koordinat uzayında doğru parçasının gösterimi yer almaktadır. Bu tanımlama doğrunun orijine göre olan yönelim açısı (ϕ) ve uzaklık değeri (ρ) parametreleriyle yapılır. Şekilde $\Delta\phi$ ve $\Delta\rho$ Hough uzayını belirli miktarda hücrelere böler. Izgara yapıdaki uzayda her bir hücre toplayıcı (Aij) olarak adlandırılır.[3]

```
1: procedure HoughTransform (x,y,ρ,φ)
2:     for i ← 1, (ρmax − ρmin)/Δρ do
           for j ← 1, (φmax − φmin)/Δφ do
3:             A_ij ← 0
           end for
4:     end for
5:     for all (x_k, y_k) ∈ D do
6:         for φ ← φmin, Δφ → φmax do
7:             ρ ← x_k cos(φ) + y_k sin(φ)
8:             if ρmin ≤ ρ ≤ ρmax then
9:                 A_ij ← A_ij + 1
10:            end if
11:        end for
12:    end for
13: end procedure
```

Her toplayıcı sayısal bir değer tutar. Algoritma başlangıcında tüm toplayıcılar sıfırlanır.

Şekilde Hough dönüşüm algoritmasına göre D veri kümesine ait tüm noktalar sırasıyla kartezyen koordinat uzayından Hough uzayına aktarılarak Aij hücre güncellemesi yapılır. Böylece tüm veri kümesi ele alındıktan sonra, Hough uzayındakienyüksek Aijmax değerine sahip hücre, doğru parçasına teğet olan polar koordinatınt(ρ,ϕ) yakınındaki bir değeri gösterir.[3]

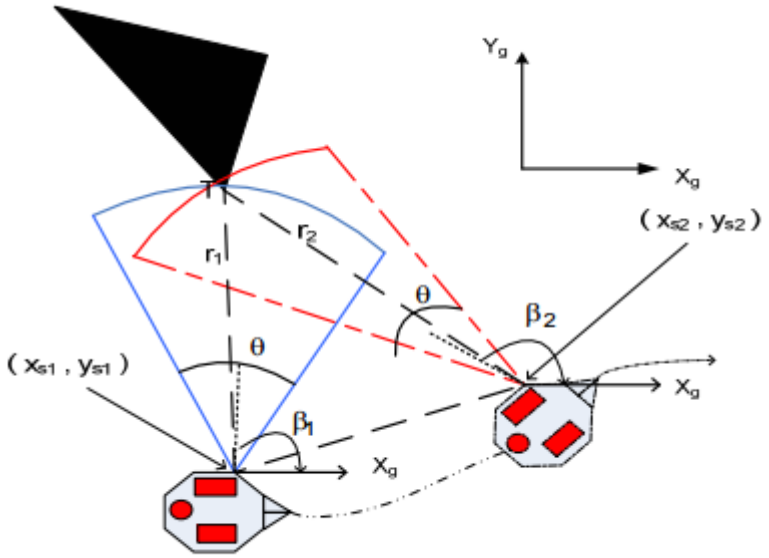
Hough dönüşüm algoritması, bir çok EKBH uygulamasında kullanılmıştır. [14]'e göre bu algoritma kullanılarak, ortamdaki hem düzlemsel bölgeler hem de köşe-kenar noktalar, sonar algılayıcılar kullanılarak tespit edilmiştir. Her bir veri kümesi robotun ortam içindeki 1.5 metrelik gezinimi sonunda elde ettiği sonar ölçümlerden oluşmuştur. $\Delta\phi$ ve $\Delta\rho$ sırasıyla, 3.50 ve 4cm olarak seçilmiştir.[3]

3.2. Üçgenleme Tabanlı Birleşim Metodu

Ses ötesi algılayıcılar ile ortam haritası oluşturmak için kullanılan yöntemlerden biri de Üçgenleme Tabanlı Birleşim metodudur. ÜTB metodu ortam içinde bulunan dikey kenarların (masa ayakları, kapı dikmesi gibi…) belirlenmesi ve konumlarının çıkartılması için geliştirilmiş bir yöntemdir.[3]

Bu yöntemde sonar ölçümler sabit açılı yay olarak modellenir. Temel prensip aynı nesne üzerinden farklı konumlarda alınmış sonar ölçümler arasında kesişim noktaları bulmak ve kenar sınıflaması yapmaktır.[3]

ÜTB metodunun Hough metodundan farkı ortam içindeki geometriksel nesnelerin direk tespit edebilmesidir. Herhangi bir dönüşüm işlemi uygulanmaz. Fakat ortamdaki düzlemleri tespit etmek için kullanılan bir yöntem değildir. Sadece kenar noktaları çıkarır [6].



Şekil-11 Üçgenleme Tabanlı Metod

Şekilde farklı robot konumlarından alınan iki sonar ölçüm görülmektedir. Burada r algılayıcının ölçtüğü mesafeyi, β sonar merkezi ile bütünsel yatay düzlem arasında kalan açıyı gösterir. Bu açı sonar merkezinin bütünsel düzlem üzerindeki yönelim doğrultusunu diğer bir ifade ile görüş alanını tanımlar. θ ise yay açıklığını ifade eden açısal bir değerdir. Bu kullanılan sonar algılayıcıların fiziksel yapısına göre değişebilir [3].

Şekilde iki ölçümün de T(xT,yT) noktası üzerinden alındığı varsayılmıştır. Sadece tek bir sonar ölçüm ile bu konum tespit edilemez. Çünkü sonarın hüzme açıklığı (θ) nedeniyle yay üzerindeki herhangi bir yerden yansıma alınmış olabilir. Konumsal belirsizlik ancak ikinci bir ölçümün daha aynı nesne üzerinden alınmasıyla düzeltilir. Bundan dolayı bu yöntem üçgenleme tabanlı olarak adlandırılır.[3]

Yayların kesişim noktası (T), her iki sonardan elde edilen bilgilerle ve eşitlikleri kullanılarak hesaplanır.

$$(x_T - x_{si})^2 + (y_T - y_{si})^2 = r_i^2, \quad i = 1, 2 \tag{3.7}$$

$$\arctan\left(\frac{y_T - y_{s_i}}{x_T - x_{s_i}}\right) \in \left[\beta_i - \frac{\theta}{2}, \beta_i + \frac{\theta}{2}\right] \quad i = 1, 2$$

$$\hat{x}_T = x_{s_2} - \frac{1}{d_s^2}\left(d_{x_s} d_r^2 \pm \left|d_{y_s}\right| \sqrt{r_2^2 d_s^2 - d_r^4}\right)$$

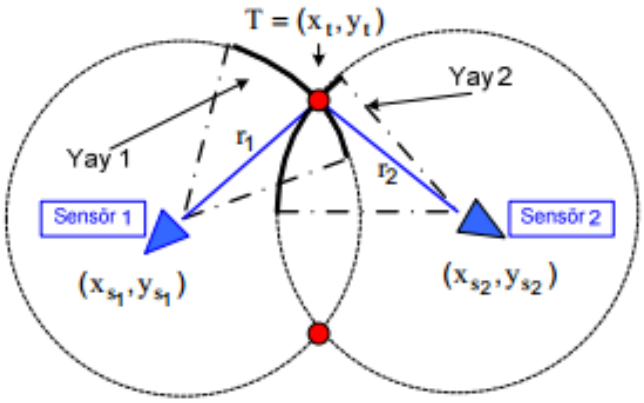
$$\hat{y}_T = y_{s_2} - \frac{1}{d_s^2}\left(d_{y_s} d_r^2 \pm \left|d_{x_s}\right| \sqrt{r_2^2 d_s^2 - d_r^4}\right)$$

$$d_{x_s} = x_{s_1} - x_{s_2}$$

$$d_{y_s} = y_{s_1} - y_{s_2}$$

$$d_s^2 = d_{x_s}^2 + d_{y_s}^2$$

$$d_r^2 = \frac{r_1^2 - r_2^2 - d_s^2}{2}$$



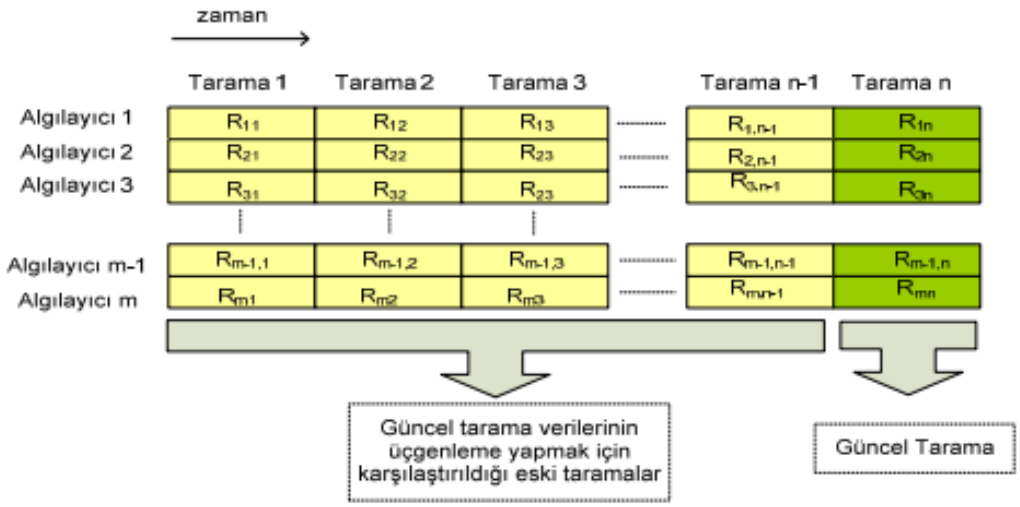
ÜTB algoritmasını uygulamak için öncelikle robotun hareketi esnasında elde ettiği sonar verilerin tutulduğu bir veri tabanı oluşturulur. Bu veri tabanı hareketli pencere olarak da adlandırılır. Bu pencere şekil 3.5'te görüldüğü gibi m satır ve n sütundan oluşmaktadır. m kullanılan sonar algılayıcı sayısını ifade eder. n ise robot gezinimi sırasında yapmış olduğu sonar ölçümlerin toplamını gösterir. Pencerenin her bir hücresindeki Rij (i=1,…,m ve j=1,…,n) veri paketi bütünsel düzlem üzerinde sonar yayları tanımlayan sonar algılayıcı bilgilerini içerir.[3]

$$R_{ij} = (x_{sij}, y_{sij}, \beta_{ij}, r_{ij}) \tag{3.8}$$

xsij ve ysij bütünsel düzlem üzerindeki algılayıcı konumu, βij sensör merkezinin bütünsel yatay eksen ile yapmış olduğu açıyı, rij ise algılayıcı ile engel arasındaki

uzaklığı ifade eder. Bu bilgiler ile aslında ortam üzerinde merkez koordinatı (xsij,ysij), yarıçapı rij ve görüş alanı olan bir yay tanımlanmıştır.[3]

$$(\beta_{ij}+\frac{\theta}{2},\beta_{ij}-\frac{\theta}{2}) \tag{3.9}$$



Şekil-30 Algılayıcı Verileri

ÜTB algoritması, hareketli pencere içersinde tanımlanan sonar yaylar arasında kesişim noktaları tespit eder. Bu pencerenin en sağ sütunu en güncel sonar ölçümleri göstermektedir. Amaç bu güncel ölçümler ile daha önceden yapılan ölçümler arasında kesişim noktası yakalayabilmektir. Bunun içinde en sağdaki sütunun birinci satırındaki sonar ölçüm R1n, diğer sütunlardaki her bir sonar ölçüm ile karşılaştırılır. Bu karşılaştırma sonucunda hesaplanan her kesişim noktası bir öncekinin üzerine eklenerek ortalaması alınır ve R1n için kesişim sayacı bir artırılır. Başlangıçta kesişim sayısı sıfır olarak atanır ve her başarılı kesişim sonunda nt bir artırılır (nt=nt+1). Diğer bir husus ise birden fazla kesişim olduğunda başlangıç varsayımına göre başarılı kesişimler arasındaki maksimum sapmanın ne kadar değiştiğidir. Bu değişim aslında alınan ölçümün kenar bölgeden mi yoksa duvar gibi düzlemsel bir bölgeden mi geldiğini göstermek için kullanılır. Diğer bir ifade ile kenar-düzlem sınıflaması bu sapma göz önünde bulundurularak yapılır.[3]

Algoritma içindeki adımlar numaralandırılmış ve detayları sayfada açıklanmıştır.

1: **procedure** ÜTB (x_s, y_s, β_s, r_s)

2: **repeat**

3: **for** $i \leftarrow 1, m$ **do**

4: **if** $r_{in} < r_{max}$ **then**

5: $n_t \leftarrow 0,\ x_T \leftarrow x_{s_{in}} + r_{in}\cos(\beta_{in}),\ y_T \leftarrow y_{s_{in}} + r_{in}\sin(\beta_{in})$

6: $x_{min} \leftarrow x_T,\ x_{max} \leftarrow x_T,\ y_{min} \leftarrow y_T,\ y_{max} \leftarrow y_T$

7: **for** $j \leftarrow n-1, 1$ **do**

8: **for** $k \leftarrow 1, m$ **do**

9: **if** $r_{kj} < r_{max}$ **then**

10: **if** $(x_T, y_T) \in R_{kj}$ **then**

11: $r_e = \sqrt{(x_T - x_{s_{kj}})^2 - (y_T - y_{s_{kj}})^2}$

12: **if** $|r_e - r_{kj}| < \frac{d_1}{n_t + 1}$ **then**

13: **if** Kesişim var mı? (çıkış: $x^{üçg}, y^{üçg}$) **then**

14: $x_T \leftarrow \frac{n_t x_T + x^{üçg}}{n_t + 1}$

15: $y_T \leftarrow \frac{n_t y_T + y^{üçg}}{n_t + 1}$

16: $n_t \leftarrow n_t + 1$

17: **if** $x^{üçg} < x_{min}$ **then** $x_{min} \leftarrow x^{üçg}$ **end if**

18: **if** $x^{üçg} > x_{max}$ **then** $x_{max} \leftarrow x^{üçg}$ **end if**

19: **if** $y^{üçg} < y_{min}$ **then** $y_{min} \leftarrow y^{üçg}$ **end if**

20: **if** $y^{üçg} > y_{max}$ **then** $y_{max} \leftarrow y^{üçg}$ **end if**

21: **end if**

22: **end if**

23: **end if**

24: **end if**

25: **end for**

26: **end for**

27: **end if**

```
28:         if n_t ≥ 1 then
29:             if (x_max − x_min)+(y_max − y_min)> d_2  then
30:                 n_t = −n_t
31:             end if
32:         end if
33:   end for
34:   if Yeni ölçümler geldi mi? then
35:         Pencere sütunlarını bir sola kaydır, do
36:         Güncel ölçümleri pencerenin en sağ sütuna yerleştir, do
37:   end if
38:  until gezinim bitene kadar
39: end procedure
```

3. adım: Pencerenin en sağ sütununda yer alan her bir sonar veri paketi (Rij) için algoritma kesişim noktası arar. Yukarıdaki algoritmada i=1 için adımlar detaylı olarak açıklanmıştır.

4. adım: r1n mesafe değerinin, maksimum ölçülebilen mesafeden küçük olması şartı aranır. Maksimum ölçülebilen mesafe sonarın fiziksel özelliğine bağlı olarak değişebilir.

5. adım: nt, xT, yT değişkenlerine başlangıç koşulları atanır. Başlangıç koşulu olarak nt = 0'dır. Bu koşul yay üzerinde kesişim noktası bulunmadığı durumdur. Bu durumda yankının sonar merkez doğrultusundan geldiği kabul edilir.

6. adım: Üçgenlemeler arasındaki maksimum sapmayı hesaplamak için başlangıç koşulu atanır.

7-8. adımlar: (n-1)'nci sütundan 1'nci sütuna (sağdan-sola doğru) tüm hücreler gezilecek şekilde döngü yapılır.

9-12. adımlar: Ðlk olarak 4. adımdaki koşulu rkj'nin de sağlayıp sağlamadığı kontrol edilir. Konum kestirimi yapılan T (xT, yT) noktasının Rkj yayının görüş alanı içinde olup olmadığına bakılır. Bunun için (3.7) eşitliği kullanılır. Son olarak beklenen mesafe re ile gerçek mesafe rkj arasındaki fark hesaplanır. Bu farkın d1 / (nt +1) 'den küçük olması beklenir. Ðzin verilen farkın nt arttıkça ,yada diğer bir ifade ile kesişim sayışı arttıkça azaldığı görülür.

13. adım: Eğer algoritmada bu adıma gelindiyse, büyük olasılıkla R1n, Rkj ölçümleri aynı nesne üzerinden alınmıştır. (3.6) ve (3.7) eşitlikleri kullanılarak yaylar arasındaki kesişim noktaları bulunur.

14-16. adımlar: Bu adımda 13. adımdan gelen her (xtri,ytri) kesişim konumu ile yankının geldiği noktanın kestirimi yinelemeli olarak güncellenir. Böylece engelin (nesnenin) konumunu kestirilmeye çalışılmış olur. Her bir üçgenleme işlemi sonunda üçgenleme sayacı bir artırılır.

17-20. adımlar: Başarılı üçgenlemeler arasında maksimum sapma güncellemesi yapılır.

28-36. adımlar: Başarılı üçgenlemeler arasındaki maksimum sapma belirlenen eşik değerinin üzerinde çıkarsa bu noktanın kenar ifade etmediği, büyük olasılıkla düzlem üzerinde olduğu sonucuna varılır. Bu sınıflamayı yapmak için yayların kesişim noktaları arasındaki maksimum sapma, eşik değerini aşarsa nt işaret değiştirir ve negatif değer alır. Negatif değerli (-nt) noktalar ortamdaki düzlemsel bölgeleri ifade etmektedir. Pozitif yüksek kesişim sayısına (nt) sahip noktalar ortamdaki kenar bölgeleri temsil eder. Artık R1n için pencere içindeki tüm hücreler gezilmiştir. En sağ sütundaki en son ölçüm R16n için de aynı işlemler tekrarlandıktan sonra yeni tarama ölçümlerinin pencereye alınması için sütunlar sola kaydırılır. En sol sütundaki tarama pencere dışına atılırken, yeni gelen güncel

tarama en sağ sütundaki yerini alır. Böylece hareketli pencere güncellemesi de yapılmış olur.[3]

4. OLASILIKSAL KONUM BELİRLEME YÖTEMLERİ

Olasılıksal konum belirleme yöntemlerinde yapılan işlemler hareket ve duyarga güncellemesi olarak ayrılabilir. Hareket güncellemesi robotun hareketi zamanı her hangi bir 1–t zamanı için kontrol işareti ile tanında nerede olabileceğini anlayabilmesidir. Algı aşamasında ise robot belli bir zaman aralığında yaptığı ölçümler sonucunda kendi konumunu belirler. Kullanılan temel parametrik yöntemlerden Kalman, parametrik olmayanlardan ise Parçacık filtrelerini gösterebiliriz.

4.1. Kalman Filtreleri

Devamlı sistemler için en iyi öğrenilmiş ve gerçekleştirilmiş Bayes filtresi Kalman Filtresidir. Kalman Filtresi Bayes filtresinin parametrize edilmiş formasıdır. Bu filtrede durum bilgisi Gauss fonksiyonu şeklinde hesaplanır. Uygun olarak kontrol ve ölçüm değerleri de Gauss gürültüsü eklenmiş parametreler ile gösterilir.

Kalman filtresinde inanç, μt ortalama değerli, Σt kovaryanslı Gauss fonksiyonu gibi yazılır.

$$p(x) = \det(2\pi)^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(x-\mu)^{T-1}(x-\mu)\right\} \quad (4.1)$$

Eşitliğindeki □vektörü x'la aynı boyutta, ise, x'in boyutunda kare matristir. Yukarıdaki denklemden yola çıkarsak ve durum vektörü için ihtimalini yazacak olursak, bu ihtimal o zaman parametrelerinin Gauss gürültüsü eklenmiş liner fonksiyonu olacaktır.

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t \quad (4.2)$$

Burada ve t anındaki durum vektörleri, kontrol vektörüdür. Onları aşağıdaki gibi dikey vektörler şeklinde gösterebiliriz:

$$x_t = \begin{pmatrix} x_{1,t} \\ x_{2,t} \\ . \\ . \\ . \\ x_{n,t} \end{pmatrix} \qquad u_t = \begin{pmatrix} u_{1,t} \\ u_{2,t} \\ . \\ . \\ . \\ u_{m,t} \end{pmatrix} \tag{4.3}$$

A kare matris olup n × m boyutlarında, B ise n× m boyutlarında geçiş matrisleridir. Burada *n* ve *m* sırasıyla x ve u vektörlerinin uzunluklarıdır. € rastgele değişeni durum değişkeni ile aynı ölçülerde olup u = 0 ve x= R şartları ödemektedir. Denklemini A_tx_{t-1} +B_tu_t ortalama değerli ve R kovaryansı için yazarsak:

$$p(x_t|x_{t-1}, u_t) = \det(2\pi R_t)^{-\frac{1}{2}} \, exp\left\{-\frac{1}{2}(x_t - A_t x_{t-1} - B_t u_t)^T R_t^{-1}(x_t - A_t x_{t-1} - B_t u_t)\right\} \tag{4.4}$$

bu görünümü kazanır.

p(zt|xt) de kendi parametrelerinin doğrusal fonksiyonu olduğuna göre

$$z_t = C_t x_t + \delta_t \tag{4.5}$$

olmalıdır. Burada da C_tk * n ölçülü geçiş matrisi (k-z ölçüm vektörünün uzunluğudur). Sigma ise ölçüm gürültüsünü belirtmektedir. Sigma için de u= □0 ve = Q değerleri temel alınacaktır. Bütün bunları yeni denklemde yerine yazarsak, o zaman denklemimiz

$$p(z_t|x_t) = \det(2\pi Q_t)^{-\frac{1}{2}} \, exp\left\{-\frac{1}{2}(z_t - C_t x_t)^T Q_t^{-1}(z_t - C_t x_t)\right\} \tag{4.6}$$

şeklini almaktadır. Son olarak başlangıç inanç olan ’ da normal dağılımlı olmalıdır

$$bel(x_0) = p(x_0) = \det(2\pi_0)^{-\frac{1}{2}} \, exp\left\{-\frac{1}{2}(x_0 - \mu_0)^T{}_0^{-1}(x_0 - \mu_0)\right\} \tag{4.7}$$

4.1.1. Kalman filtresi algoritması

Yukarıda belirttiğimiz gibi Kalman filtresi zamanı bütün veriler, yani hareket ve ölçüm modelleri Gauss dağılımı ile gösterilmeli, her bir inanç fonksiyonu kendine özgün ortalama değer ve kovaryans ile belirtilmelidir. Bir önceki ortalama değer ve kovaryans hareket ve duyarga bilgilerine dayanarak sonraki ortalama değer ve kovaryanslar hesaplanmalıdır. Kalman sürecinde kontrol verileri bellidir. Ama sistemin gürültüsünü bilmediğimizden sürecin sonunda elde edeceğimiz küme gerçek kümeye ola bildiğince yakın olmalıdır.
6 Kalman filtresi geniş kullanılmasına rağmen veri ilişkilendirme gibi önemli bir probleme çözüm getirememektedir.
Bundan başka Kalman filtresinin haritalamadaki kısıtları aşağıdakilerdir:

- Yapay sınırlara veya duvarlara ihtiyaç duyulur
- Direk ham algılayıcı datası kullanılmaz, veriler ön işlemlerden geçirilmelidir
- Algılayıcı verisinin sahip olduğu gürültü tipi Gauss dağılımında olmalıdır
- Başlangıçta robot pozisyonunda kayma olmadıysa harita iyi çıkar, ancak kayma olduysa, zaman geçtikçe bu kayma büyüyeceğinden kötü sonuçlar alınmaktadır.
- Sadece doğrusal sistemler için çalışabilmektedir.

4.1.2 Genişletilmiş Kalman Filtresi

Bir önceki başlıkta Kalman filtresinin bazı artı ve eksileri gösterilmiştir. Eksilerinden en önemlisi ise sadece doğrusal sistemler için çalışmasıdır. Yani, durum ve ölçüm modelleri doğrusal denklemlerle ifade edilmelidir. Ama ya robot herhangi bir dairesel yolu takip ederse veya kontrol fonksiyonu doğrusal değilse? Bu sorulara cevap vermek için filtremiz doğrusal olmayan yani non-linear sistemler için çalışmalıdır (Kalman,

1960). Bu yüzden Kalman filtresinin daha genel ifadesi olan Genişletilmiş Kalman Filtresi meydana çıkmıştır.

GKF' de bir sonraki durum ve ölçüm olasılık fonksiyonları sırasıyla ó ve h ile kumanda edilmektedir.

$$x_t = g(u_t, x_{t-1}) + \varepsilon(t)$$
$$z_t = h(x_t) + \delta(t) \tag{4.8}$$

Burada g fonksiyonu Kalman filtresindeki A ve B′ in, h ise C′ in yerine geçmiştir. Ama KF' den farklı olarak GKF gerçek inanç değerine yakın bir değer elde etmektedir.

Şekillerde denklemlerindeki g ve h fonksiyonları Gauss dağılımına sahip olmadıklarını biliyoruz. GKF ilk adımı bu fonksiyonları doğrusallaştırma olacaktır. Bu işlemi de Taylor sırası açılımına göre yazabiliriz. O zaman,

$$g'(u_t, x_{t-1}) = \frac{\partial g(u_t, x_{t-1})}{\partial x_{t-1}} \tag{4.9}$$

olacaktır. Doğrusal olmayan ó fonksiyonunu Gauss fonksiyonu doğrultusunda doğrusallaştırmak için parametreleri daha uygun seçmek gerekir. Bu durumda g için gerçek değerlere yakın bir parametre olarak μ_{t-1} seçilmiştir.

$$g(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + \underbrace{g'(u_t, \mu_{t-1})}_{=:G_t}(x_{t-1} - \mu_{t-1}) =$$
$$= g(u_t, \mu_{t-1}) + G_t(x_{t-1} - \mu_{t-1}) \tag{4.10}$$

O zaman bir sonraki durum ihtimali yaklaşık olarak Gauss fonksiyonu gibi şöyle yazılabilir:

$$p(x_t|x_{t-1}, u_t) \approx \det(2\pi R_t)^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}[x_t - g(u_t, \mu_{t-1}) - G_t(x_{t-1} - \mu_{t-1})]^T R_t^{-1}[x_t - g(u_t, \mu_{t-1}) - G_t(x_{t-1} - \mu_{t-1})]\right\} \tag{4.11}$$

Burada matrisi n × n ölçülerinde olup (n – durum verisinin ölçüsüdür) Jacobian olarak da bilinmektedir. Bu Jacobian' ın değeri u_t ve u_{t-1}değerlerine bağlıdır ve her bir nokta için farklıdır.

Aynı yolla bütün bu denklemleri h ölçüm fonksiyonu için yazabiliriz:

$$h'(x_t) = \frac{\partial h(x_t)}{\partial x_t}$$
$$h(x_t) \approx h(\bar{\mu}_t) + \underbrace{h'(\bar{\mu}_t)}_{=:H_t}(x_t - \bar{\mu}_t) = h(\bar{\mu}_t) + H_t(x_t - \bar{\mu}_t) \quad (4.12)$$

Şekle göre Gauss fonksiyonu yazacak olursak;

$$p(z_t|x_t) \approx \det(2\pi Q_t)^{-\frac{1}{2}} exp\left\{-\frac{1}{2}[z_t - h(\bar{\mu}_t) - H_t(x_t - \bar{\mu}_t)]^T Q_t^{-1}[z_t - h(\bar{\mu}_t) - H_t(x_t - \bar{\mu}_t)]\right\} \quad (4.13)$$

GKF' in bazı artıları olduğu gibi birkaç eksisi de mevcuttur.

- Yaklaşık Gauss fonksiyonu hesaplanması zamanı sistemin derecesi belli olmadığından, bu işlemlerin karmaşıklığını de beraberinde getirir. Sistemin derecesi arttıkça Taylor sırası uzar.

- Jacobian matrislerinin hesaplanması işlemlerin sayının artmasına getirir.

- Doğrusallaştırma, bizi gerçek inanç değerinden uzaklaştırır.

4.2. Parçacık Filtresi

Parçacık filtresi Bayes filtrelerinin parametrik olmayan alternatif realizasyonu ve sonlu sayıda parametrenin reel inanç değerine yakınsamasıdır. PF' in ana konusu **bel**(x_t) değerini, ondan önceki **bel**(x_{t-1}) değerinden, rastgele durum örnekleri kümesi ile oluşturulmasıdır (Crisan ve

Doucet, 2002). Ama GKF' de olduğu gibi bu değerin kendisini değil de ona yakın bir değeri bulmaktadır. Bunu yaparken dağılımların Gauss dağılımı olması zorunluluğu da yoktur.
PF' de önceki durum dağılımlarını 'parçacık' adlandırılmakta ve aşağıdaki gibi gösterilmektedir:

$$\chi_t := x_t^{[1]}, x_t^{[2]}, \dots, x_t^{[M]} \quad (4.14)$$

Burada her bir x_t (1<m<M) parçacığı, t anındaki durumu gösteriyor. M ise burada Xt parçacık kümesindeki parçacıklarını sayını göstermektedir. Genellikle parçacık sayısı
M= 1000 gibi büyük rakamlar oluyor. Bazen de M,t'nin fonksiyonu olabilir. Yukarıdaki gösterimden yola çıkarsak:

$$x_t^{[m]} \sim p(x_t | z_{1:t}, u_{1:t}) \quad (4.15)$$

Gibi yazılabilir.

PF algoritması da GKF gibi inanç değerlerini yaklaşık olarak hesaplamaktadır. Tam dakik olmayan böyle hesaplamalar zamanı yakınsama hataları meydana çıkmaktadır. Aşağıda bazı hatalar ve oluşma sebepleri verilmiştir:

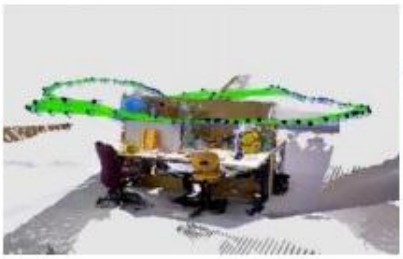
• Parçacık sayısının sonlu olması: Mesela M = 1 varsayalım. O zaman döngü bir alt aşamaya geçmeyerek ölçüm verilerini kullanmayacak ve yeni parçacıklar sadece p(xt ut-1) değerinden hesaplanacaktır.[12]

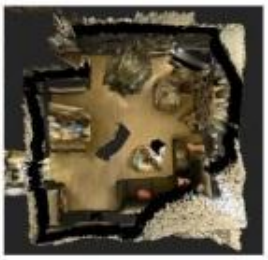
• 'Resampling' aşamasının rastgele olması: Bu hatayı göstermek için en uç örneği vermemiz lazım. Varsayalım ki robot durumu değişmiyor, yani xt=xt-1 Yani,
9 hareket etmeyen bir robot örneğini inceliyoruz. Bir de robotun sensor ölçümü yapmadan durum algılaması yapamayacağını düşünelim. Bu durumda ise yeni durum örneği

hesaplanamayacak. Sonuç olarak, bütün durum örnekleri bir birinin kopyaları olarak belirecek ve robotun sensorlarının çalışmadığı gibi hiç de gerçek olmayan bir bilgi elde edilmiş olacaktır.[13]

- Hedef ve göreceli dağılımların oranı: Sensorlara ve robot hareketine bağlı olarak bu oran değişmektedir. Sensor verilerinin gürültülü olduğu durumlarda p(z/x)değeri bütün durumlar için sıfır değeri alacak. Bu durumda dağılım fonksiyonu z ölçüm değerleri dâhilinde örnek x üretmeyecektir. Böyle olan halde ağırlık değerli de sıfıra eşit olmaktadır. Sonraki döngü ise devam etmeyecektir.[14]
- Yüksek dereceli sistemler daha fazla sayıda parçacık ile filtreleme talep ediyor ki bu da beraberinde işlem karmaşıklığını dolayısıyla sistemin pahalı olmasına getirip çıkarıyor.

5. BAZI ÖRNEK HARİTALAMALAR





SONUÇ

Gezgin robotlar Amaçlarını yerine getirmek için gezinim sırasında ilk olarak konumlarını belirlemelidirler. Robotun bulunduğu ortamdaki konumunun belirlenmesi odometrik veriler, sensör ölçümler ve harita bilgilerinin birleştirilmesiyle oluşturulur. Haritalarla ilgili veriler başlangıçta robota verilebileceği gibi robotun hareketi esnasında robot tarafından da oluşturulabilir. Bu durumda robotun konumunun belirlenmesi ve eş zamanlı olarak haritalama yapabilmesi için çeşitli yaklaşımlar ve uygulamalar incelenmekte ve geliştirilmektedir. Eş zamanlı olarak ortam haritasının çıkartılması ve bu haritanın kullanılmasıyla konum belirlenmesi gelişen teknoloji ve değişen jenerasyonda birçok açıdan gerekli olduğundan bu yaklaşımların incelenmesini ve geliştirilmesini zorunlu hale getirmiştir. Bu tez çalışmasında ortam haritası oluşturulurken, robot konumunun da eş zamanlı olarak belirlenmesi için gereken hazırlıklar ve yöntemler incelenmiştir.

Sonuç olarak bu çalışmada özellikle ucuza mal edilebilen, sınırlı bilgi toplayabilen sensör çeşitleri ile bile gerekti yaklaşımlar dikkatli olarak yapıldığında; Eş Zamanlı Konum Belirleme Ve Haritalama işleminin yapılabileceği gösterilmiştir.

EKLER

Ek:1 FastSLAM Algoritmaları ve açıklamalar

Ek:2 FastSLAM Algoritmasının uygulaması

Ek-1 FastSLAM Algoritmaları ve Açıklamalar

FastSLAM algoritması 1.0 (z_t, u_t, S_{t-1}):

1. For $m = 1$ to M do

   S_{t-1} 'den $\langle s_{t-1}^{[m]},\ N_{t-1}^{[m]},\ \langle \mu_{1,t-1}^{[m]}, \Sigma_{1,t-1}^{[m]}, i_1^{[m]} \rangle, \dots, \langle \mu_{N_{t-1}^{[m]},t-1}^{[m]}, \Sigma_{N_{t-1}^{[m]},t-1}^{[m]}, i_{N_{t-1}^{[m]}}^{[m]} \rangle \rangle$ oku

   Bütün parçacıkları oluşturulması, işaretlenmesi

2. $s_t^{[m]} \sim p(s_t \mid s_{t-1}^{[m]}, u_t)$

   Yeni robot konumunun oluşturulması

3. For n = 1 to $N_{t-1}^{[m]}$ do

   Sensor ölçüm benzerliklerinin hesaplanması

4. $\hat{z}_n = g(\mu_{n,t-1}^{[m]}, s_t^{[m]})$

   Ölçüm tahmini hesaplanması

5. $G_n = g'(s_t^{[m]}, \mu_{n,t-1}^{[m]})$

   Jakobyen hesaplanması

6. $Q_n = G_n^T \Sigma_{n,t-1}^{[m]} G_n + R_t$

   Sensor ölçüm kovaryansının hesaplanması

7. $w_n = |2\pi Q_n|^{-\frac{1}{2}} exp\left\{-\frac{1}{2}(z_t - \hat{z}_n)^T Q_n^{-1}(z_t - \hat{z}_n)\right\}$

   Haberleşmenin benzerliğinin bulunması

   Endfor

8. $w_{N_{t-1}^{[m]}+1} = p_0$

   Yeni harita detayının önemlilik derecesinin hesaplanması

9. $\hat{n} = \underbrace{argmax}_{n=1,\dots,N_{t-1}^{[m]}+1} w_n$

   Max benzerlik haberleşmesi

10. $N_t^{[m]} = max\left\{N_{t-1}^{[m]},\ \hat{n}\right\}$

    Haritadaki detayların yeni sayı

11. For $n = 0$ to $N_t^{[m]}$ do

    Kalman filtresinin güncellenmesi

12. İf $n = N_{t-1}^{[m]} + 1$ then

    Yeni harita detayı mı?

13. $\mu_{n,t}^{[m]} = g^{-1}(z_t, s_t^{[m]})$

Ortalama değerin alınması

14. $\Sigma_{n,t}^{[m]} = G_{\hat{n}}^{-1} R_t (G_{\hat{n}}^{-1})^T$

Kovaryansın alınması

15. $i_{n,t}^{[m]} = 1$

Sayacın başlatılması

16. Else if $n = \hat{n}$ *then*

Daha önce görülmüş detay mı?

17. $K = \Sigma_{n,t-1}^{[m]} G_{\hat{n}} Q_{\hat{n}}^{-1}$

Kalman sayısının(gain) hesaplanması

18. $\mu_{n,t}^{[m]} = \mu_{n,t-1}^{[m]} + K(z_t - \hat{z}_n)^T$

Ortalama değer güncellenmesi

19. $\Sigma_{n,t}^{[m]} = (I - K G_{\hat{n}}^T) \Sigma_{n,t-1}^{[m]}$

Artımlı sayaç

20. $i_{n,t}^{[m]} = i_{n,t-1}^{[m]} + 1$

İşlemler diğer kalan detaylar için tekrarlanıyor

21. Else

Bir önceki ortalama değerin alınması

22. $\mu_{n,t}^{[m]} = \mu_{n,t-1}^{[m]}$

Bir önceki kovaryansın alınması

23. $\Sigma_{n,t}^{[m]} = \Sigma_{n,t-1}^{[m]}$

Detay bulunmuş olabilir mi?

24. İf $\mu_{n,t-1}^{[m]}$, $s_t^{[m]}$ ' algı alanında değilse then

Hayırsa, hiçbir şeyi değiştirme

25. $i_{n,t}^{[m]} = i_{n,t-1}^{[m]}$

Evetse, sayıyı bir azalt

Else

26. $i_{n,t}^{[m]} = i_{n,t-1}^{[m]} - 1$

Tekrarlanan detayları temizle

27. İf $i_{n,t-1}^{[m]} < 0$ then n numaralı harita özelliğini at, endif

Endif

Endif

Endfor

S_{aux} 'e $\langle s_t^{[m]},\ N_t^{[m]},\ \langle \mu_{1,t}^{[m]}, \Sigma_{1,t}^{[m]}, i_1^{[m]} \rangle, \ldots,\ \langle \mu_{N_t^{[m]},t}^{[m]}, \Sigma_{N_t^{[m]},t}^{[m]}, i_{N_t^{[m]}}^{[m]} \rangle \rangle$ ekle

Yeni parçacık kümesi oluştur

Endfor

28. $S_t = \emptyset$

M adet parçacığı bütünleştir

29. For $m' = 1\ to\ M\ do$

30. $\propto w_t^{[m]}$ Olasılıklı m rastgele indisini oluştur

S_t 'e $\langle s_t^{[m]},\ N_t^{[m]},\ \langle \mu_{1,t}^{[m]}, \Sigma_{1,t}^{[m]}, i_1^{[m]} \rangle, \ldots,\ \langle \mu_{N_t^{[m]},t}^{[m]}, \Sigma_{N_t^{[m]},t}^{[m]}, i_{N_t^{[m]}}^{[m]} \rangle \rangle$ ekle

Bütünleştirme aşaması

Endfor

Return S_t

End(algoritma)

FastSLAM algoritması 2.0 (z_t, u_t, S_{t-1}):

1. For $m = 1$ to M do

   S_{t-1} 'den $\langle s_{t-1}^{[m]},\ N_{t-1}^{[m]},\ \langle \mu_{1,t-1}^{[m]}, \Sigma_{1,t-1}^{[m]}, i_1^{[m]} \rangle, \dots, \langle \mu_{N_{t-1}^{[m]},t-1}^{[m]}, \Sigma_{N_{t-1}^{[m]},t-1}^{[m]}, i_{N_{t-1}^{[m]}}^{[m]} \rangle \rangle$ oku

   Bütün parçacıkları oluşturulması, işaretlenmesi

2. For n = 1 to $N_{t-1}^{[m]}$ do

   $\hat{s}_t^{[m]} = h\left(s_{t-1}^{[m]}, u_t\right);\ \hat{z}_{t,n_t}^{[m]} = g(\mu_{n_t,t-1}^{[m]}, \hat{s}_t^{[m]})$

   $G_{\theta,n_t} = \nabla_{\theta_{n_t}} g(\theta_{n_t}, s_t)\Big|_{s_t=\hat{s}_t^{[m]};\theta_{n_t}=\mu_{n_t,t-1}^{[m]}}$; $G_{s,n_t} = \nabla_{s_t} g(\theta_{n_t}, s_t)\Big|_{s_t=\hat{s}_t^{[m]};\theta_{n_t}=\mu_{n_t,t-1}^{[m]}}$

   $Q_{t,n_t}^{[m]} = R_t + G_{\theta,n_t}\Sigma_{n_t,t-1}^{[m]} G_{\theta,n_t}^T$

   $\Sigma_{s_t,n_t}^{[m]} = [G_{s,n_t}^T Q_{t,n_t}^{[m]-1} G_{s,n_t} + P_t^{-1}]^{-1}$; $\mu_{s_t,n_t}^{[m]} = \Sigma_{s_t,n_t}^{[m]} G_{s,n_t}^T Q_{t,n_t}^{[m]-1}\left(z_t - \hat{z}_{t,n_t}^{[m]}\right) + \hat{s}_t^{[m]}$

   Sampling dağılımının hesaplanması

3. $s_{n_t,t}^{[m]} \sim \mathcal{N}\left(\mu_{s_t,n_t}^{[m]}, \Sigma_{s_t,n_t}^{[m]}\right)$

   $$p_{n_t} = \left|2\pi\, Q_{t,n_t}^{[m]}\right|^{-\frac{1}{2}} exp\left\{-\tfrac{1}{2}(z_t - g(\mu_{n_t,t-1}^{[m]}, s_{n_t,t}^{[m]}))^T Q_{t,n_t}^{[m]-1}(z_t - g(\mu_{n_t,t-1}^{[m]}, s_{n_t,t}^{[m]}))\right\}$$

   Örnek konumun bulunması

   Endfor

4. $p_{N_{t-1}^{[m]}+1} = p_0$

   Yeni detayın benzerliği

5. $\hat{n}_t^{[m]} = argmax_{n_t} p_{n_t}$ Ve ya $\propto p_{n_t}$ olasılıklı $\hat{n}_t^{[m]}$ 'i rastgele oluştur

   Veri ilişkilendirme

6. For $n = 1\ to\ N_{t-1}^{[m]} + 1\ do$

   Proses ölçümü

7. İf $n_t = \hat{n}_t \leq N_{t-1}^{[m]}$ then

   $N_t^{[m]} = N_{t-1}^{[m]}$; $\tau_{n_t,t}^{[m]} = \tau_{n_t,t}^{[m]} + \rho^+$; $K_t^{[m]} = \Sigma_{\hat{n}_t,t-1}^{[m]} G_{\theta,\hat{n}_t}^T Q_{t,\hat{n}_t}^{[m]-1}$

   $\mu_{\hat{n}_t,t}^{[m]} = \mu_{\hat{n}_t,t-1}^{[m]} + K_t^{[m]}\left(z_t - \hat{z}_{t,\hat{n}_t}^{[m]}\right)$; $\Sigma_{\hat{n}_t,t}^{[m]} = (I - K_t^{[m]} G_{\theta,\hat{n}_t})\Sigma_{\hat{n}_t,t-1}^{[m]}$

   $L_t^{[t]} = G_{s,\hat{n}_t} P_t G_{s,\hat{n}_t}^T + G_{\theta,\hat{n}_t}\Sigma_{n_t,t-1}^{[m]} G_{\theta,\hat{n}_t}^T + R_t$

   $$w_t^{[m]} = \left|2\pi\, L_t^{[t]}\right|^{-\frac{1}{2}} exp\left\{-\tfrac{1}{2}(z_t - \hat{z}_{t,\hat{n}_t})^T L_t^{[t]-1}(z_t - \hat{z}_{t,\hat{n}_t})\right\}$$

   Daha önceden bilinen detay mı?

8. Else if $n_t = \hat{n}_t = N_{t-1}^{[m]} + 1$ then

$n_t = N_t^{[m]} = N_{t-1}^{[m]} + 1;\ \tau_{n_t,t}^{[m]} = \rho^+;\ w_t^{[m]} = p_0$

$s_{n,t}^{[m]} \sim p(s_t \mid s_{t-1}^{[m]}, u_t);\ G_{\theta,n} = \nabla_{\theta_n} g(\theta_n, s_t)\big|_{s_t = s_{n,t}^{[m]}, \theta_n = \mu_{n,t}^{[m]}}$

$\mu_{n,t}^{[m]} = g^{-1}\left(z_t,\ s_{n,t}^{[m]}\right);\ \Sigma_{n,t}^{[m]} = (G_{\theta,n} R_t^{-1} G_{\theta,n}^T)^{-1}$

Yeni detay mı?

9. Else if $n_t \neq \hat{n}_t$ *and* $n_t \leq N_{t-1}^{[m]}$

   $\mu_{n,t}^{[m]} = \mu_{n,t-1}^{[m]};\ \Sigma_{n,t}^{[m]} = \Sigma_{n,t-1}^{[m]}$

   Bulunamamış detayların ele alınması

10. İf $\mu_{n,t}^{[m]}$, $(s_{\hat{n}_t,t}^{[m]})$ algı alanı dışında kalıyorsa then

    Sensor algı alanının dışında mı kalıyor?

11. $\tau_{n_t,t}^{[m]} = \tau_{n_t,t-1}^{[m]}$

    Sensor algı alanının içinde mi kalıyor?

12. Else

    Detaydan vazgeçelim mi?

13. $\tau_{n_t,t}^{[m]} = \tau_{n_t,t-1}^{[m]} - \rho^-$; if $\tau_{n_t,t}^{[m]} < 0$ then n_t' at

    Endif

    endif

    endfor

    S_{aux} 'e $\langle s_t^{[m]},\ N_t^{[m]},\ \langle \mu_{1,t}^{[m]}, \Sigma_{1,t}^{[m]}, i_1^{[m]} \rangle, \ldots,\ \langle \mu_{N_t^{[m]},t}^{[m]}, \Sigma_{N_t^{[m]},t}^{[m]}, i_{N_t^{[m]}}^{[m]} \rangle\rangle$ ekle

    Bütün parçacıkların alınmasının durdurulması

    Endfor

14. $S_t = \emptyset$

    Yeni parçacık kümesini oluştur

15. For $m' = 1$ *to* M *do*

    M adet parçacık oluştur

16. $\propto w_t^{[m]}$ Olasılıklı m rastgele indisini oluştur

    S_t 'e $\langle s_t^{[m]},\ N_t^{[m]},\ \langle \mu_{1,t}^{[m]}, \Sigma_{1,t}^{[m]}, i_1^{[m]} \rangle, \ldots,\ \langle \mu_{N_t^{[m]},t}^{[m]}, \Sigma_{N_t^{[m]},t}^{[m]}, i_{N_t^{[m]}}^{[m]} \rangle\rangle$ ekle

    Bütünleştir (resample)

    Endfor

    Return S_t

    Endalgoritma

Ek 2 FastSLAM Algoritmasının Uygulaması

FastSLAM algoritmasının iki farklı sürümü bulunmaktadır. Çalışmada FastSLAM2.0 algoritması kullanılmıştır. Simülasyonun çalışma sürecinde çalışılan sisteme ve robot konfigürasyonuna has birtakım parametreler mevcuttur. Bu parametrelerin değer tamaları sistemde 'Constants' klasında yapılmaktadır. Harita detaylarının (engel, detay) pozisyonu ise simülasyonda engel ölçümlerini hesaplamak için ve alınan ölçümlerden yola çıkılarak bulunan engel pozisyonları ile gerçek engel pozisyonlarını haritada gösterebilmek için gereklidir.

Uygulamada, kontrol parametreleri, doğrusal ve açısal hızdan oluşmaktadır. Sırayla [**V**; **G**] değişkenleriyle gösterilir. Birimleri sırayla cm/sn ve radyan/sn verilmiştir. İlk önce robot konfigürasyonuna dair atamalar yapılır. Örneğin, ön tekerin kontrol işaretleri sonucunda sahip olabileceği maksimum açı değeri radyan cinsinden **MAXG**'de tutulmaktadır. Ön tekerin saniyedeki maksimum değişim oranı (açısal hızı) radyan/sn cinsinden **RATEG**'de; robotun aks (dingil) yani ön teker ve arka teker arasında uzanan birleştirici doğrunun uzunluğu (ön teker-arka teker mesafesi) metre cinsinden **WHEELBASE**'de tutulmaktadır. Simülasyonda kullanılan robot modelinde bu mesafe 16 cm olarak belirlenmiştir Bunların dışında algı ile ilgili belirlenmesi gereken parametreler aşağıdaki gibidir:

- Peş-peşe gönderilen iki kontrol işareti arasında geçen süre saniye cinsinden **DT_CONTROLS**'te,
- Kontrol işaretlerini (doğrusal ve açısal hızlar) gönderirken oluşabilecek (Gaussian tipteki) gürültülerin standart sapma değerleri sırasıyla **sigmaV** (cm/sn cinsinden) ve **sigmaG**'de (rad/sn cinsinden),

- Alınan gözlemlerde görülebilecek maksimum mesafe santimetre cinsinden **MAX_RANGE**' de,

- Alınan gözlemler arasında geçen süre sn cinsinden **DT_OBSERVE**'de;
- Gözlem esnasında oluşabilecek gürültülerin standart sapmaları **sigmaR** (santimetre cinsinden-doğrusal mesafe gürültüsü için) ve **sigmaB**'de (rad cinsinden-açısal fark (eğim farkı) gürültüsü için);

- Gezinme sürecinin kaç defa yapılacağı **NUMBER_LOOPS**'ta;
- Parçacık adedi **NPARTICLES**'ta;
- Yeniden örnekleme (resampling) yapmadan önce işe yarayan (efektif) parçacık adedi **NEFFECTIVE**'de;
- Kontrol, gözlem ve tahmin süreçlerine gürültü eklenip eklenmeyeceğine karar verecek olan kontroller sırayla **SWITCH_CONTROL_NOISE, SWITCH_SENSOR_NOISE, SWITCH_PREDICT_NOISE**'te tutulmaktadır.

Öncelikle NPARTICLES adet parçacığa ilk değer ataması yapılır. İlk robot pozisyonu (xtrue) belirlendikten sonra ilk açısal hız (G) değerine 0 atanır. Q: kontrol gürültülerinin (doğrusal ve açısal hız için) standart sapmalarının karelerini (varyans) diagonelinde bulunduran kovaryans matrisi; R: gözlem gürültülerinin (doğrusal ve açısal mesafe ölçümü için) varyanslarını diagonelinde bulunduran kovaryans matrisidir. Q'da ve R'de doğrusal ve açısal gürültüler birbirinden bağımsızdır, dolayısıyla sadece diagonelleri doludur. (Yani Q(1,2)=Q(2,1)=0 olur: doğrusal ve açısal hız gürültü ilişkisizdir; aynı şekilde R(1,2)=R(2,1)=0'dır.)

Daha sonra yeni robot pozisyonu hesaplandıktan (**xtrue)**. Kontrol gürültüsü eklenerek gürültülü kontrol datası üretiliyor **([Vn,Gn]).**

Tahmin adımında her bir parçacık robot pozisyonuna dair kendi tahminini buna kendi tahmin gürültüsünü ekleyerek yapar. Sonra gözlem işlemleri yapılır. Gözlem adımında peş-peşe yapılan iki gözlem arasında geçmesi gereken min süreye ulaşılmışsa, gözlem alınır ve sayaç sıfırlanır(**dtsum**). Robotun görüş alanına giren harita detaylarının gerçek ölçüm değerleri (robot-detay arası gerçek mesafeler) ve bu engellerin indisleri bulunur (**[z,ftag_visible]).** Hesaplanan gerçek ölçümlere (**z**) gürültü eklenir.

Nf: 1.parçacıkta associated (önceden görülen) detay sayısıdır. Tüm parçacıklar eşit sayıda detay içerdiği için herhangi birindeki detay sayısını (**xf**' deki eleman sayısını) almak yeterlidir. Daha sonra ‘‘veri ilişkilendirme ’’ olup olmadığı hesaplanır ve **da_table** değişkeninde önceden görülen detayların indisleri utulur**. [zf,idf,zn,da_table]** Burada **zf** ve **idf**: Önceden görülen detayların ölçüm ve indis değerleri; **zn:** yeni bulunan detayların ölçümünü göstermektedir.

Güncelleme adımında önceden görülen detay'ler kümesi **zf** doluysa: **i** numaralı parçacığın ağırlığı (**w**), **i** numaralı parçacığı, **zf**, **idf** ve **R**-gözlem gürültü matrisini kullanarak bulunur. Sonraki adımda **i** numaralı parçacığın ağırlığı (w alanı) güncellenir. **i** numaralı parçacığın detay konumu tahmin dağılımının Multivariate Gauss olduğunu varsayılmaktadır. Detay

konum tahmininde dağılımın **xf** ortalama değer vektorü ve **Pf** kovaryans matrisi güncellenir. Bir sonraki adımda yeni detay bulunduysa harita büyütülmeli ve parçacıklara ekleme yapılmalıdır ve işlem sırasında **i** numaralı parçacığa **zn** ve **R**'yi kullanarak, yeni bulunan ağırlıkların tahminini eklenmelidir. Bundan sonra ise parçacıklar, önerilen dağılıma göre yeniden örneklenir (Resampling) ve çıktı olarak, son durumu gösteren parçacıklar kümesi döndürülür (**data= particles**).

FastSLAM'de gereken ilk değer atamaları yapıldıktan sonra **NPARTICLES** adet parçacığın ilk değer atamaları yapılır. (**np**: **NPARTICLES**). **1/ NPARTICLES** ağırlık değerleri başta hepsine eşit atanır. **xv i** numaralı parçacığın robot pozisyonu tahmini "**xv**" alanındadır ve [x;y;teta] şeklinde, **xf** i numaralı parçacığın detay pozisyonlarının tahminidir ve [x;y;teta] şeklindedir. **Pf** ise i. parçacığın detay konum tahmin dağılımının Multivariate Gauss olduğunu yukarıda kabul etmiştik. Detay konum tahminde **xf** bu dağılımın ortalama değer vektorü, **Pf** ise kovaryans matrisidir.

Prior state tahmini aşamasında ise **[xfi,Pfi]**, innovasyon matrisleri **[v=zp-z, R]** ve doğrusallaştırılmış gözlem modeli **(H)** kullanılarak KF güncelleştirmesini yapılırken kullanılan geçiş matrisidir. **Pf** boyutu: (2x2xdetay sayısı)'dir. Gerçek pozisyondan, kontrol parametreleri [V; G]'den, **dt** ve robot konfigürasyon parametrelerinden yola çıkılarak, yeni robot pozisyonu hesaplanır ve döndürülür (**xv**).

Bir adım sonra ise kontrol gürültüsü eklenerek gürültülü kontrol verisi **(Vn, Gn)** üretilip gönderilecektir.

Harita detaylarının algılanması aşamasında detaylarla robot konumu arasındaki mesafe ve eğim açısı farkları bulunur (**dx, dy, phi) z**, doğrusal ve açısal gözlem mesafeleri hatası bağımsız olduğundan R kovaryans matrisinin sadece diyagonali kullanılarak buna gürültü eklenir. Sonra sırasıyla **[zp,Hv,Hf,Sf]** Jakobian hesapları yapılır. Burada;

zp: Parçacığın tahmin ettiği ölçüm (robot-landmark arası doğrusal ve açısal mesafe); **Hv:** robot durumunu hesaplayan Jacobian matrisi;

Hf: detay durumunu hesaplayan Jacobian matrisidir ve Kalman Filtresinde gözlem modeli gibidir;

Sf: robot durumu verildiğinde, detay durumunu hesaplayan matristir ve robotun konumundan harita detayı değerini bulan yeni bir kovaryans matristir.

Yapılan bütün bu işlemlerden sonra parçacıklar yeniden örneklenmelidir. Bu aşamada ağırlık dizisini (**w**) üzerinden; stratified (çok katmanlı) yeniden örnekleme yapılır ve çıkışta korunacak parçacıkların indisi ve efektif olan parçacık adedi elde edilir. Yeniden örnekleme

sonucunda bazı parçacıklar yeni kümeye giremez, bazısı ise birden fazla kez girer. Hata ne kadar düşükse ağırlık o kadar yüksek olur; hata arttıkça da ağırlık değeri azalır.

KAYNAKLAR

1. Lakaemper R., Jan Latecki L., Sun X. and Wolter D., "Geometric Robot Mapping", Discrete Geometry for Computer Imagery: 12th International Conference, DGCI 2005, Poitiers,Fransa, Nisan 2005.

2. Hogue Oral, "Simultaneous Localization and Mapping Techniques Oral Exam": June 20,2005

3. Orkun ALP, Gezgin Robotlarda Eş Anlı Haritalama Ve Konum Belirleme, Başkent Üniversitesi Fen Bilimleri Enstitüsü.

4. S.Thrun, W. Burgard, D. Fox. "Probabilistic Robotics", Intelligent Robotics and Autonomous Agents series, The Massachusetts Institude of Technology Press. Htpp://mitpress.mit.edu

5. A. Elfes, "Using Occupancy Grids for Mobile Perception and Navigation. Computer", 22(6):46-57,1989

6. Billur Barshan, Directional Processing of Ultrasonic Arc Maps and its Comparison with Existing Techniques, 2007 The International Journal of Robotics

7. Elif Eroğlu, Gezgin Robotlarda Ultrasonik Mesafe Algılayıcılarla Robot Davranışlarının Kontrolü ve Çevre Haritalama, Yüksek Lisans Tezi, Haziran 2006

8. Thrun, S., Fox, D. and Burgard, W.,1998, A probabilistic approach to concurrent mapping and localization for mobile robots, Machine Learning, Vol.31
9. Olle Wijk, Henrik I. Christensen "Triangulation-Based Fusion of Sonar Data with Application in Robot Pose Tracking", 2000 IEEE Transactions on Robotics and Automation Vol.16, No.6

10. Kurt, Z. (2007), "Es Zamanlı Konum Belirleme ve Haritalamaya Yönelik Akıllı Algoritmaların Gelistirilmesi", Yüksek Lisans Tezi, Yıldız Teknik Üniversitesi, İstanbul.

11. Siegwart, R. ve Nourbakhsh, I.R. (2004), Introduction to Autonomous Mobile Robots, The MIT Pres, Cambridge, Massachusetts.

12. Thrun, S. , Burgard, W. ve Fox, D. (2005), Probabilistic Robotics, The MIT Press, Cambridge, Massachusetts.

13. Montemerlo M. (2003) ''FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem With Unknown Data Association'', Doktora tezi, Carnegie Mellon University.

14. Hahnel D. , Burgard W. , Fox D. ve 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, 206–211.

15. *Griepentrog, Hans Werner, B. Simon Blackmore, and Stravros G. Vougioukas. 2006. Section 4.2*

16. *Positioning and Navigation, pp. 195-204 of Chapter 4 Mechatronics and Applications, in CIGR*

17. *Handbook of Agricultural Engineering Volume VI Information Technology. Edited by CIGR-The*

18. *International Commission of Agricultural Engineering; Volume Editor, Axel Munack. St. Joseph,*

19. *Michigan, USA: ASABE. Copyright American Society of Agricultural Engineers.*

20. Afig HABİBOV, Gezgin Robotlarda Eş Zamanlı Konum Belirleme Ve Haritalama, İstanbul, 2011.

21. LeMaster, E. 2003. GPS on the web—Applications of GPS pseudolites. *GPS Solutions* 6:268-270.

22. Borenstein, J. 1996. Measurement and correction of systematic odometry errors in mobile
robots. *IEEE Trans. Robot. Autom.* 12(6): 869-880.

23. Borenstein, J., H. R. Everett, L. Feng, and D. Wehe. 1997. Mobile robot positioning—Sensors and techniques. *J. Robotic Systems* 14(4): 231-249.

24. Olson, C. F. 2002. Selecting landmarks for localization in natural terrain. *Autonomous Robots* 12(2): 201-210.

25. Shmulevich, I., G. Zeltzer, and A. Brunfeld. 1989. Laser scanning method for guidance of
field machinery. *Trans. ASAE* 32(2): 425-430.

26. Yang, H., K. Park, J. G. Lee, and H. Chung. 2000. A rotating sonar and a differential encoder data fusion for map-based dynamic positioning. *J. Intelligent and Robotic Systems* 29(3): 211-232.

27. De Schutter, J., J. De Geeter, T. Lefebvre, and H. Bruyninckx. 1999. Kalman filters—A tutorial. Leuven, Belgium: Katholieke Universiteit Leuven.

28. Han, S., Q. Zhang, and H. Noh. 2002. Kalman filtering of DGPS positions for a parallel tracking application. *Trans. ASAE* 45(3): 553-560.

29. Kuipers B. , JJ. Modayil, P. Beeson, M. MacMahon ve F. Savelli, Local Metrical and Global Topological Maps in the

Hybrid Spatial Semantic Hierarchy, in *IEEE Int. Conf. on Rob. and Aut.*, ss. 4851-4845, 2004

30. Thrun S., Robotic mapping: A survey, *Exploring Artificial Intelligence in the New Millennium*, 2002.

31. Klatzky R.L., Allocentric and egocentric spatial representations: Definitions, distinctions,and interconnections, *Spatial cognitionAn interdisciplinary approach to representation and processing of spatial knowledge (Lecture Notes in Artificial Intelligence 1404)*, ss. 1-17, Springer-Verlag, 1998

32. Elfes A. Sonar-based real-world mapping and navigation. *IEEE Trans. on Rob. and Aut.*, Cilt 3(3) ss. 249-265, 1987

33. Brunskill E., T. Kollar ve N. Roy, Topological mapping using spectral clustering and classification, *Proc. of IEEE/RSJ Int. Conf. on Robots and Systems*, ss. 3491-3496, 2007

34. Paskin M. A., Thin junction tree filters for simultaneous localization and mapping, *Proc. of Int. Joint Conf. on Artificial Intelligence*, ss. 1157-1164, 2003

35. Chang H., C. Lee, Y. Hu ve Y.-H. Lu, Multi-robot slam with topological/metric maps, *Proc. of IEEE/RSJ Int. Conf. on Robots and Systems*, ss. 1467-1472, 2007

36. Ni K., D. Steedly ve F. Dellaert, Tectonic sam: exact, out-ofcore, submap-based slam, *IEEE Int. Conf. on Rob. and Aut.*, ss.1678-1685, 2007

37. Jaulin L., Range-only slam with occupancy maps: A setmembership approach, *IEEE Trans. on Rob.*, Cilt 27(5), ss. 1004-1010, 2011

38. Se S. , D. Lowe ve J. Little, Vision-based global localization and mapping for mobile robots, *IEEE Trans. on Rob.*, Cilt 21(3), ss. 364-375, 2005

39. Remolina E. ve B. Kuipers, Towards a general theory of topological maps, *Artificial Intelligence*, Cilt 152(1), ss. 47-104, 2004

40. Fenwick J., P. Newman ve J. Leonard, Cooperative concurrent mapping and localization, *IEEE Int. Conf. on Rob. and Aut.*, Cilt 2, ss. 1810-1817, 2002.

41. Dedeoglu G. ve G. S. Sukhatme, Landmark based matching algorithm for cooperative mapping by autonomous robots, *Distributed Autonomous Robotic Systems*, Cilt 4, ss. 251-60, 2000

42. Davison A., I. Reid, N. Molton ve O. Stasse, Monoslam: Realtime single camera slam, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Cilt 29(6), ss. 1052-1067, 2007

43. Oliva A. ve A. Torralba, Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope.*Int. J. of Computer Vision*, Cilt 42(3), ss. 145-175, 2001

44. Jogan M. ve A. Leonardis, Robust localization using panoramic view-based recognition, *Int. Conf. on Pattern Recognition*, Cilt 4, ss. 136-139, 2000

45. Krose B., N. Vlassis, R. Bunschoten ve Y. Motomura, A probabilistic model for appareance-based robot localization, *Image and Vision Computing*, Cilt 19, no. 6, ss. 381-391, 2001

46. Torralba A., K. P. Murphy, W. T. Freeman ve M. A.Rubin, Context-based vision system for place and object recognition, *IEEE Int. Conf. on Computer Vision*, Cilt 1, ss. 273, 2003

47. Wisniewski R., Towards Modelling of Hybrid Systems, *Proc. of the 45th IEEE Conf. Decision & Control*, ss. 911-916, 2006 [175] Wolf J., W. Burgard, ve H. Burkhardt, Robust vision

48. Fazl-Ersi E. ve J.K. Tsotsos, Histogram of Oriented Uniform Patterns for robust place recognition and categorization, *Int. J. Rob. Res.*, Cilt 31(4), ss. 468-483, 2012.

49. Ulrich I. ve I. Nourbakhsh, Appearance-based place recognition for topological localization, *IEEE Int. Conf. on Rob. and Aut.*, Cilt 2, ss. 1023-1029, 2000

50. Huang W. H. ve K. R. Beevers, Topological map merging, *Int. J. Rob. Res.*, Cilt 24(8), ss. 601-613, 2005

51. Harris C. ve M. Stephens,A combined corner and edge detector, *AVS88*, ss. 147-151, 1988

52. Bay H., Herbert, T. Tuytelaars ve L. Van Gool, Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, Cilt 110, ss. 346-359, 2008

53. Bozma H.I. , G. Çakiroglu ve Ç. Soyer, Biologically inspired Cartesian and non-Cartesian filters for attentional sequences. *Pat. Rec. Letters*, Cilt 24(9-10), ss. 1261-1274, 2003

54. Fraundorfer F., C. Engels ve D. Nister, Topological mapping, localization and navigation using image collections, *Proc. of*

IEEE/RSJ Int. Conf. on Robots and Systems, ss. 3872-3387, 2007

55. Murillo A., J. Guerrero ve C. Sagues, Surf features for efficient robot localization with omnidirectional images, *IEEE Int. Conf. on Rob. and Aut.*, ss. 3901-3907, 2007

56. Sivic J. ve A. Zisserman, Video Google: A text retrieval approach to object matching in videos. *Int. Conf. on Computer Vision,*, Cilt 2, ss. 1470-1477, 2003

57. Cummins M. ve P. Newman, Fab-map: Probabilistic localization and mapping in the space of appearance, *Int. J. Rob. Res.*, Cilt 27, ss. 647-665, 2008

58. Lamon R., I. Nourbakhsh, B. Jensenl ve R. Siegwart, Deriving and matching image fingerprint sequences for mobile robot localization, *Proc. of IEEE/RSJ Int. Conf. on Robots and Systems*, ss. 1609-1610, 2001

59. Nistér, D. ve H. Stewénius.. Scalable recognition with a vocabulary tree. *Proc. of the 2004 IEEE Int. Conf. on Comp. Vision and Patt. Recog.*, Cilt 2, ss. 2161-2168, 2006

60. Konolige K., J. Bowman, J. Chen, M. Mihelich, Patrick andCalonder, V. Lepetit, ve P. Fua, View-based maps, *Int. J. Rob. Res.*, Cilt 29(8), ss. 941-957, 2010

61. Cummins M. ve P. Newman, Appearance-only SLAM at large scale with fab-map 2.0, *Int. J. Rob. Res.*, Cilt 30(9), 2010

62. Zhou X. ve S. Roumeliotis, Multi-robot slam with unknown initial correspondence: The robot rendezvous case, *Proc. of*

IEEE/RSJ Int. Conf. on Robots and Systems, ss. 1785-1792, 2006

63. Albus J.S. , Outline for a Theory of Intelligence. IEEE Trans. on Sys., Man, and Cybern., Cilt 21(3), ss. 473-509, 1991

64. Meilland M., A.I. Comport ve P. Rives, A spherical robotcentered representation for urban navigation, *Proc. of IEEE/RSJ Int. Conf. on Robots and Systems*, ss. 5196-5201, 2010

65. Rossi F., A. Ranganathan, F. Dellaert ve E. Menegatti, Toward Topological Localization with Spherical Fourier Transform and Uncalibrated Camera. *Omnidirectional Robot Vision Workshop, Int. Conf. Simulation, Modeling, and Programming for Autonomous Robots*, ss. 319-330, 2008

66. Soyer C., H. I. Bozma ve Y. Istefanopulos, Apes-biologically motivated attentive robot, *Auton. Rob.*, Cilt 20, ss. 61-80, 2006

67. Se S. , D. Lowe ve J. Little, Vision-based global localization and mapping for mobile robots, *IEEE Trans. on Rob.*, Cilt 21(3), ss. 364-375, 2005

68. Desai J. P., A graph theoretic approach for modeling mobile robot team formations, *J. Robot. Syst.*, 19(11), ss. 511-555, 2002.

69. Zivkovic Z., B. Bakker ve B. Krose, Hierarchical map building and planning based on graph partitioning, *IEEE Int. Conf. on Rob. and Aut.*, ss. 803-809, 2006

70. Friedman S., H. Pasula ve D. Fox, Voronoi random fields: Extracting topological structure of indoor environments via place labeling, *Proc. of the Int. Joint Conf.onArtificialIntelligence*
ss. 2109-2114, 2007

ÖZGEÇMİŞ

İsmail DEMİR 1990'da Adıyaman'da doğdu; ilk ve orta öğrenimini aynı şehirde tamamladı; Bilgi Anadolu Lisesinden mezun olduktan sonra 2011 yılında Karabük Üniversitesi, Mühendislik Fakültesi, Mekatronik Mühendisliği Bölümü'ne girdi. Halen bu bölümde eğitimini sürdürmektedir.

İletişim Bilgileri

Adres: Beşbinevler mah. 33. Sokak. 12/7
Merkez / KARABÜK

E-posta: idemir-@hotmail.com

Tel: 05310319113