İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

OTONOM MOBİL ROBOTLARDA DAĞILIMLI KALMAN FİLTRESİ TABANLI EŞ ZAMANLI LOKALİZASYON VE HARİTALAMA

YÜKSEK LİSANS TEZİ

Kadir PASLIOĞLU

Anabilim Dalı : Mekatronik Mühendisliği

Programı : Mekatronik Mühendsiliği

Ocak 2010

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

OTONOM MOBİL ROBOTLARDA DAĞILIMLI KALMAN FİLTRESİ TABANLI EŞ ZAMANLI LOKALİZASYON VE HARİTALAMA

YÜKSEK LİSANS TEZİ

Kadir Paslıoğlu

Tezin Enstitüye Verildiği Tarih : 25 Aralık 2009

Tezin Savunulduğu Tarih : 25 Ocak 2010

Tez Danışmanı : Prof. Dr. Hakan TEMELTAŞ (İTÜ)

Diğer Jüri Üyeleri : Prof. Dr. Metin GÖKAŞAN(İTÜ)

Doç. Dr. Cenap Ş. ÖZBEN (İTÜ)

Ocak 2010

ÖNSÖZ

Yaptığım çalışma ve yüksek lisans boyunca bana verdiği destekten dolayı TUBİTAK'a, araştırmlarım için labaratuvar imkanları sağlayan ve çalışmam boyunca bana yol gösteren Prof. Dr. Hakan Temeltaş'a, yaptığım deneysel çalışmalar ve simülasyonlarda büyük yardımları dokunan M. Kürşat Yaçın, Emrah Adamey ve Onur Sencan'a son olarak tüm üniversite yaşamım boyunca bana destek olan yardımlarını hiçbir zaman esirgemeyen aileme teşekkürü bir borç bilirim.

Aralık 2009

Kadir Paslıoğlu

Fizik Mühendisi

İÇİNDEKİLER

Sayfa

ÖZET ..... **xv**
SUMMARY ..... **xvi**
1. GİRİŞ **1**
2. MOBİL ROBOTLARIN KİNEMATİK MODELLENMESİ **4**
2.1. Araç Modeli 4
2.1.1.UGV tahrik sistemleri 4
2.1.2. Diferansiyel araç modeli ve hesaplanması 10
2.2. Sensör Modeli 13
3. OPTİMAL DURUM KESTRİMİ **17**
3.1. Genişletilmiş Kalman Filtresi 17
3.2. Dağılımlı Kalman Filtresi 19
3.2.1. Dağılımlı dönüşüm 20
3.2.2. Ölçümlendirilmiş dağılımlı dönüşüm 22
3.2.3. Dağılımlı kalman filtresi 24
4. SLAM **28**
4.1. SLAM Probleminin Yapısı 29
4.1.1. Olasılıksal SLAM 31
4.2. SLAM Problemleri 31
4.2.1. Hesaplama problemi 32
4.2.1.1. Durum artırımı 33
4.2.1.2. Parçalanmış durumların güncellenmesi 34
4.2.1.3. Seyreltme 35
4.2.1.4. Altharitalar 36
4.2.2. Veri ilişkilendirme 36
4.2.2.1. Grup doğrulama 37
4.2.2.2. Görünüm tanıma 38
4.2.2.3. Çokluhipotez veri ilişkilendirme 38
4.2.3. Gürültü Problemi 38
4.2.4. Çevre Problemi 39
4.3. GKF-SLAM 45
4.4. FastSLAM 51
5. DKF Tabanlı SLAM **57**
6. UGULAMA **58**
7. SONUÇLAR VE TARTIŞMA **67**
KAYNAKLAR **69**
EKLER **70**
ÖZGEÇMİŞ **74**

KISALTMALAR

SLAM : Simultaneous Localization And Mapping, Eş Zamanlı Lokalizasyon ve Haritalama
GKF : Genişletilmiş Kalman Filtresi
FastSLAM : Hızlı olarak Eş Zamanlı Lokalizasyon ve Haritalama
DKF : Dağılımlı Kalman Filtresi
DD : Dağılımlı Dönüşüm
ÖDD : Ölçümlendirilmiş Dağılımlı Dönüşüm
MCL : Monte Carlo Lokalizasyonu
SIR : Ardışıl Önem Örneklemesi(Sequential Importance Resampling)

ÇİZELGE LİSTESİ

Sayfa

Çizelge 2.1 : Tekerlekli kara araçlarında çeşitli teker konfigürasyonları (Z.Kurt Y.L.Tezi). 7
Çizelge 2.2 : Ackermann Tekerlek Sisteminde Kullanılan Semboller. 9
Çizelge 2.3 : Diferansiyel Tekerlek Sisteminde Kullanılan Semboller. 12
Çizelge 3.1 : GKF Algoritması Denklemleri. 17
Çizelge 3.2 : DKF Algoritması Denklemleri. 28

ŞEKİL LİSTESİ

Sayfa

Şekil 1.1 : Ayrık Zamanlı Lineer Olmayan Sistem. 3
Şekil 2.1 : Holonomik olmayan Ackermann tekerlek sistemi. 6
Şekil 2.2 : Önde ve Arkada birer serbest tekerlek, ortada servo motor eksenli 2 tane ana tekerlek. 8
Şekil 2.3 : Diferansiyel tekerlek sisteminde Kullanılan semboller..... 9
Şekil 2.4 : Diferansiyel Araç Hareketi..... 11
Şekil 2.5 : Diferansiyel Araç Hareketi..... 12
Şekil 2.6 : Mobil Robotta 2 SICK LMS200 ün yerleşimi … 14
Şekil 2.7 : Lazer sensörün çalışmasının yandan görünüşü 14
Şekil 2.8 : LMS 200 ün taraması (13mslik 1^o aralıkla 180^o lik tarama)..... 15
Şekil 2.9 : Diferansiyel Aracın Lazerlerle Gözlem Yapması. 16
Şekil 3.1 : Kalman Filtresinin Yapısı 18
Şekil 3.2 : Ortalama ve Kovaryans Hesabı..... 23
Şekil 3.3 : Dağılımlı Dönüşümün Şematik Gösterimi..... 25
Şekil 4.1 : Otonom Diferansiyel Araçta SLAM problemi..... 30
Şekil 4.2 : CLSF. Ön SLAM (b) deki lokal haritaya göre oluşturuluyor ve periyodik olarak (a)'daki global haritaya kayıtlanarak (c)'deki optimal global kestirimi oluşturmaktadır 35
Şekil 4.3 : Gloabal (a) ve relatif (b) altharitalama örnekleri 37
Şekil 4.4 : Bir araçta Oluşabilecek atama Problemi. 38
Şekil 4.5 : Kestirim Aracı Yapısı..... 41
Şekil 4.6 : GKF Akış Diyagramı 43
Şekil 4.7 : SLAM Probleminin FastSLAM hali 45
Şekil 4.8 : FastSLAMın şematik gösterimi..... 45
Şekil 4.9 : RBKF Akış şeması 46
Şekil 5.1 : DKF Akış Diyagramı 51
Şekil 6.1 : Uygulama 1. 78
Şekil 6.2 : Uygulama 1 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata 79
Şekil 6.3 : Uygulama 1 işaretçi nesnelere ait belirsizlik elipsleri..... 79
Şekil 6.4 : Uygulama 2. 78
Şekil 6.5 : Uygulama 2 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata 79
Şekil 6.6: Uygulama 2 işaretçi nesnelere ait belirsizlik elipsleri..... 79
Şekil 6.7 : Uygulama 3. 78
Şekil 6.8 : Uygulama 3 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata 79
Şekil 6.9 : Uygulama 3 işaretçi nesnelere ait belirsizlik elipsleri..... 79
Şekil 6.10 : Uygulama 4. 78
Şekil 6.11 : Uygulama 4 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata 79
Şekil 6.12 : Uygulama 4 işaretçi nesnelere ait belirsizlik elipsleri..... 79

Şekil 6.13 : Uygulama 5. 78
Şekil 6.14 : Uygulama 5 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata 79
Şekil 6.15 : Uygulama 5 işaretçi nesnelere ait belirsizlik elipsleri 79
Şekil 6.16 : Uygulama 6. 78
Şekil 6.17 : Uygulama 6 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata 79
Şekil 6.18 : Uygulama 6 işaretçi nesnelere ait belirsizlik elipsleri 79
Şekil A.1 : DKF Simulink şeması ana görünüm 79
Şekil A.2 : Tek Araç için DKF-SLAM simulink bloğu ana görünüm 79
Şekil A.3 : DKF-SLAM simulink Bloğu 79
Şekil A.4 : DKF-SLAM de zaman güncellemesi adımında (a) sigma noktalarını Hesaplama (b) 79
Şekil A.5 : DKF-SLAM ölçüm güncellemesi 79
Şekil A.6 : Veri ilişkilendirme bloğu 79

OTONOM MOBİL ROBOTLARDA DAĞILIMLI KALMAN FİLTRESİ TABANLI EŞ ZAMANLI LOKALİZASYON VE HARİTALAMA

ÖZET

Kestirim araçları mobil robotların navigasyonunda kullanılan eş zamanlı haritalama ve navigasyon algroritmalarının tabanını oluşturur. Son yirmi yıldır en iyi bilinen teknik genişletilmiş kalman filtresi tabnalı eş zamanlı haritalama ve lokakizasyon dur. Bunun yanında Parçaçık Filtrenin kullanıldığı FastSLAM teknikleri de gelşitirilmiştir. Parçaçık filteresi yöntemi kendi içinde genişletilmiş kalman filtreleri içerir. Günümüzde ise araştırmalar Dağılımlı Kalman Filtresi tabanlı eş zamanlı lokalizasyon ve haritalama algoritmalarına odaklanmıştır. Kararlı ve doğru haritaların oluşturulması için kulanılan kestirim araçlarınında kararlı olması gerekir. Dağılımlı kalman filtresi tekniğinin geliştirilmesindeki amaç, diğer filtrelerdeki kusurları indirgeyerek bu kararlığı sağlamaktır.

Dağılımlı Kalman Filtresi, insansız araçların navigasyonunda en çok kullanılan genişletilmiş kalman filtresinin eksiklerini gidermek amacıyla, alternatif olarak geliştirilmiş bir kestirim aracıdır.Dağılımlı kalman filtrelerinde , Genişletilmiş Kalman Fİltresi de bulunan türev alma işlemleri(jacobian, hessian) yoktur. Bu nedenle Genişletilmiş Kalman Filtresi ye göre daha optimal çalışan bir filtredir.

Bu tezde dağılımlı kalman filtresi tabanlı eş zamanlı haritalama ve lokalizasyon algoritmaları oluşturulmuştur.Ve bu yöntemin mobil robotların navigasyon uygulamalarındaki uyumluluğunu görmek amaçlanmıştır. Sigma noktalı filtreler ailesinden olan Dağılımlı kalman filtresi incelenmiştir ve diğer bir kestirim yöntemi olan Genişletilmiş Kalman Filtresi ile karşılaştırılmıştır. Simulasyon çalışmalarımız, bilgisayar ortamında MATLAB programı kullanılarak yapılmıştır. Deneysel çalışmalarımız ise laboratuvarımızda kurulmuş olan insansız kara araçları üzerinde yapılmıştır. Çeşitli yörüngeler ve değişik işaretçi nesne sayıları ile testler yapılmıştır.

UNSCENTED KALMAN FILTER BASED SIMULTANEOUS LOCALIZATOIN AND MAPPING IN AUTONOMOUS MOBILE ROBOTS

SUMMARY

Estimation methods are base elements for simultaneous localization and mapping algorithms used for monile robot navigation. simultaneous localization and mapping with extended kalman filters has been best known method for last twenty years.Beside these method FastSLAM using particle filters techniques are developed too. Particle filters include extenden kalman filters in itself. Nowadays, researches focused on simultaneous localization and mapping algorithms with unscented kalman filters. For constructing stationary and reliable maps, used estimation tools must be consistent.The purpose for developin unscented kalman filter is to overcome some flaws in other estimation methods and make that consistency..

Unscented Kalman Filter, is an alternative estimation method to mostly used extended kalman filters to deal with its flaws, in unmanned vehicles navigation. In unscented kalman filter, there is no need to calculate jacobian or hessians like GKF. Beacuse of that it works more optimal then GKF.

Uncented Kalman filters based simultaneous localization and mapping algorithms are developed in this thesis. And compatibility of this method in mobile robot navigation implementations is observed. Unscneted Kalman Filter from the family of sigma point kalman filters is examined. Simulation works is done by using MATLab program. We did the experimental works in our laboratory with previously made unmanned vehicles. Several test are made with different trajectories and different landmark numbers.

1. GİRİŞ

Bir mobil robot hiç bilinmeyen bir ortama bırakıldığında, aynı anda hem bu oratamın haritasını çıkarıtıp (haritalama) hemde kendi konumunu belirleyebilir mi (lokalizasyon)? Literatürde SLAM problemi olarak yer alan " Eş zamanlı Haritalama ve Lokalizasyon" pobleminin temeli bu sorudur. Burada görülmektedir ki mobil robottan bilmediği konumunu, belirsiz bir çevrede bulması istenmektedir. Bu navigasyon işleminin en temel methodlarından biri uzaktan kumanda ile aracın yönetilmesidir. Fakat bu işlemin otonom olarak yapılabilmesi için SLAM olarak tanımlanmış olan Eş zamanlı haritalama ve lokalizasyon algoritmaları kullanılmaktadır. SLAM ile ilgili ilk çalışmalar 1980 li yıllarda başlanmıştır. İlk ele alınan çalışma Durrant-Whyte,Crovley ve Cheeseman ın 1986 da yaptıkları olasılıksal metodlar içeren bir çalışmadır. Ölçüm ve aracın konumundaki belirsizliklerin indirgenmesi ve daha güvenilir veriler elde edilmesi için günümüzde SLAM algoritmaları geliştrilmektedir.

SLAM probleminin ilk olarak ortaya, 1986 yılında Cheeseman, Crowley ve Whyte tarafından çıkartılmıştır. Bu yıllarda olasılıksal metodlar henüz ki robotiğe ve yapay zekaya adapte edilmeye çalışılıyordu. Yapılan çalışmada haritalama ve lokalizasyon problemlerine teorik kestirim metodları ile yaklaşılmıştır.Daha sonraki bir kaç yıl içinde Smith ve Cheeseman, Whtye ın çalışmalarında işaretçi nesneler arasındaki ilişki ve geometrideki belirsizlikleri giderme çalışmaları ortaya konulmuştur. Bu çalışmların gösterdiği en önemli nokta ise, haritdaki konumları ölçülen işaretçi nesneler ile yapılan kestirimler arasında yüksek derecede bir korelasyon olduğu ve bu korelasyonun başarılı gözlemlerle güçlendirilebileceğidir[1].

Aynı zamanda yapılan Ayache ve Faugeras , Crowley ve Chatlia nın yaptığı çalışmalar ise, görsel navigasyon ve sonara dayalı navigosyon metodlarını içermektedir. Bu çalışmalardan çıkna sonuç; bilinmeyen bir ortamda ilerleyen mobil robotun yaptığı birbiriyle bağıntılı işaretçi nesne gözlemleri, kestirilen araç pozisyonundaki genel hata nedeni ile bir birlerine bağlıdırlar. Bu durum ise kararlı bir çözümün, hem araç pozisyounun hemde her işaretçi nesneyi içeren bir durum

vektörü içermesi gerketiğini, ve bu vektörün yapılan her yeni işaretçi gözleminde güncellenmesi gerektiğini doğurur. Fakat böyle bir durum kestirim aracının çok büyük bir durum vekörü ile hesap yapması demekdir ki boyutları ölçülen işaretçi sayısının karesi kadardır.Yapılan bu son çalışmalarda haritanın yakınsama özellklerine veya durağan haldeki davranışlarına değinilmemiştir. Aslında, o zamanda tamamen kabul edilen, kestrilmiş harita hatalrının yakınsamıyacağı aksine kontrolsüz bir hata artışı ile bir rastgele yürüyüş davranışı ortaya koycağıdır.Böylece araştırmacılar, haritalama probleminin hesaplama yoğunluğu ve haritanın yakısama davranışı konuları yerine, daha önce süregelen haritalama problemlerine bir kaç yaklaşımda bulunmuşlardır. Bunların sonucunda, haritalama ve lokalizasyona ayrı ayrı problemmiş gibi yaklaşım yapıldığından, birleşik haritalama ve lokalizasyon problemi geçici olarak bir çıkmaza girmiştir[1].

Çıkış noktasına, ayrı ayrı düşünülen haritalama ve lokalizasyon problemlerinin, gerçektende bir noktada keşitiğinin farkedilmesi ile ulaşılmıştır.Bir çok araştırmacının minimize etmeye çalıştığı, işaretçiler arasındaki korelasyonun problemin kritik bir noktasını oluşturduğu farkedilmiştir. İşaretçi nesneler arasındaki ilişki ne kadar artarsa o kadar iyi çözüm elde edileceği görülmüştür.Bu son gelişmeler ışığın SLAM yapısı 1996 yılında Whyte, Rye ve Nebot un yapıtığı çalışma ile ortaya konulmuştur.

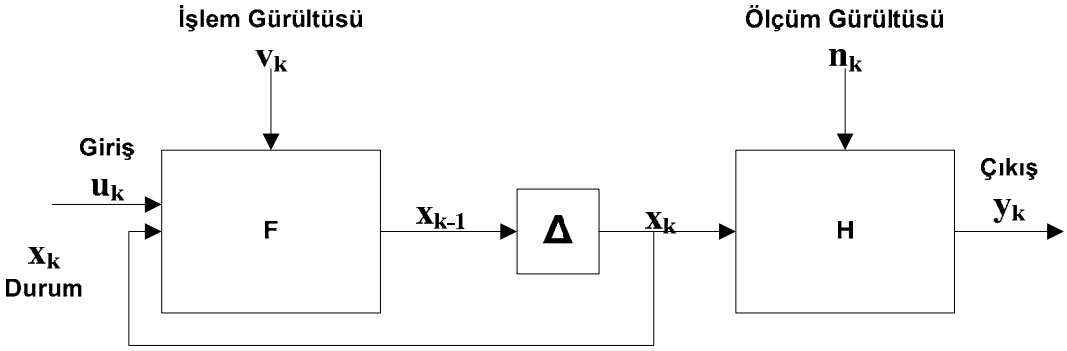
Daha sonra çalışmalar hesaplama karmaşıklığını indirgeme ve veri ilişkilendirme konuları üzerine yoğunlaşmıştır. 1999 da Thrun, kalman filtresi tabanlı SLAM ve olasılıksal lokalizasyon ve haritalama metodlarını sunmuştur[1].

Bu çalışmalar, SLAM içinde kestirim algoritmaları ve veri ilişkilendirme algoritmaları geliştirmete yoğunlaşmaktadır. İlk olarak Smith tarafından geliştirilen genişletilmiş kalman filtresi kestirim aracı olarak kullanılmıştır. GKF tabanlı SLAM de amaç robot konumunu ve işaretçilerin konumunu birleştirlmesiyle, bir ayrık zamanlı durum vektörü oluşturmaktır. Bu vektör içinde yer alan sensör verileri ve araç bilgileri ardaşıl olarak güncellenir. Yakın zaman da GKF tabanlı SLAM tekniğinde görülen bazı eksiklikleri gidermek için yeni methodlar geliştirilmiştir. Bunlara örnek olarak Sigma noktalı kalman filtrelerini (SPKF) verebiliriz. SPKF ailesinden biri olan, 1996 yıkında Julier ve Uhlmann ın çalışması olan Dağılımlı(Dağılımlı) Kalman Filtresi bu tezde incelediğimiz ve GKF ye alternatif

olan bir filtredir. Daha sonra 2000 yılında Wan ve Merwe yaptıkları çalışmalarla DKF nin kullanım alanını genişletmişlerdir. Diğer filtreler gibi DKF de lineer olmayan uygulamaların hepsinde kullanılarbilir. Şekil 3.1 de gösterilmiş olan lineer olmayan herhangi bir dinamik sistemde, diğer filtrelerde olduğu gibi DKF de durum ve parametre kestirimi yapabilir.,

$$x_{k+1} = f(x_k, w_k, k) \quad \textbf{(1.1)}$$

$$y_k = h(x_k, k) + v_k \quad \textbf{(1.2)}$$



Şekil 1.1: Ayrık Zamanlı Lineer Olmayan Sistem

DKF nin temelinde dağılımı(Dağılımlı) dönüşüm(DD) yatar unscneted filtering and nonlinear estimation julier. DD rastgele bir değişkenin lineer olmayan bir dönüşümde istatistiğini hesaplamak için kullanılan bir metoddur. Bu method, bir olasılık dağılımına yaklaşım yapmak, rastgele seçilmiş lineer olmayan bir fonksiyona yaklaşım yapmaktan daha basittir, temeline dayanır[2].

Bu tezde DKF tabanlı SLAM algoritmaları kullanılarak insansız kara aracında uyglaması yapılmıştır.İlk olarak bilgisayar ortamında, çeşitli yürüngeler ve farklı işratçi nesene sayıları kullanılarak simulasyon çalışmaları yapılmıştır. Daha sonra laboratuvarımızda bulunan iki insansız araç üzerinde algoritmalar denenmiştir. Ve GKF tabanlı SLAM methodu ile karşlılaştırılmıştır.

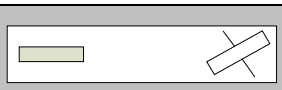
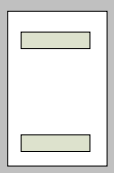
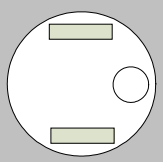
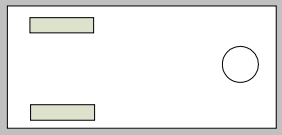
2. MOBİL ROBOTLARIN KİNEMATİK MODELLEMESİ

2.1 Araç Modeli

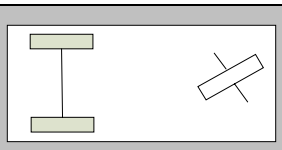
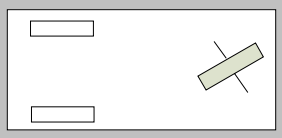
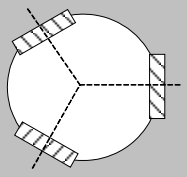
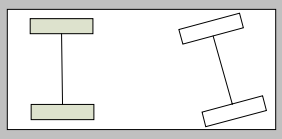
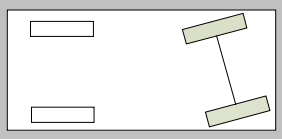
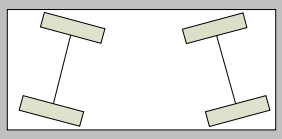
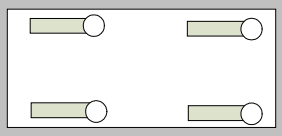
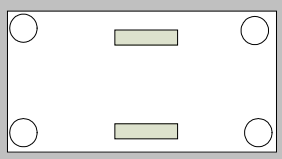
2.1.1 UGV hareket sistemleri

Mobil robotların hareket sistemlerinin SLAM algoritmalarında önemli bir yeri vardır. Bu sistemler tekerlekli, paletli veya ayaklı olabilirler.Yapılan çalışmada literatürdeki tahrik sistemleri incelenerek en uygun olanı seçilmiştir. Ve seçilen sistemin fiziksel modelinin denklemlerinden faydalanılmıştır.

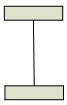
Çizelge 2.1: Tekerlekli kara araçlarında çeşitli teker konfigürasyonları (Z.KurtY.L.Tezi)

Teker sayısı	**Teker Düzeni**	**Açıklama**	Örnek
2		Yön belirleyen bir ön teker, gövdeyi ön tekerin çektiği yere sürükleyen bir arka teker vardır.	Bisiklet, Motorsiklet
		İki teker bir mille bağlıdır, ağırlık merkezi milin orta noktasıdır.	
3		Önde bir teker, arkada farklı sürüş merkezleri olan iki teker bulunmaktadır.	
		**Önde çok yönlü bir teker, arkada iki bağımsız teker vardır.**	Çoğu kapalı ortam robotları (Plgmalion ve Alice)

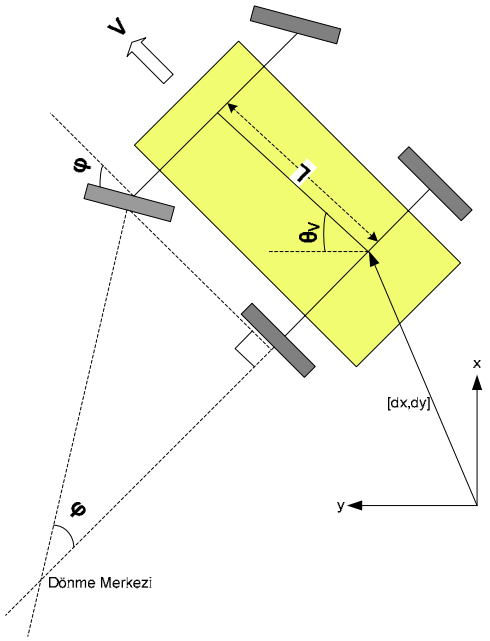
Çizelge 2.1: (devam) Tekerlekli kara araçlarında çeşitli teker konfigürasyonları (Z.KurtY.L.Tezi)

3		Önde yön veren bir teker, arkada bir mille birbirine bağlı iki ayrı teker vardır.	
		Önde yön veren bir teker, arkada iki teker vardır.	Neptune (Carnegie Mellon Üniversitesi)
		Bir üçgen düzenine oturtulmuş üç adet çok yönlü İsveç tekeri veya küresel teker.	Tribolo EPFL, Palm Pilot Robot Kit (Carneige Mellon Üniversitesi)
4		Önde yön veren iki teker, arkada motorlu iki teker bulunmaktadır.	Arkadan sürüşlü arabalar
		Önde iki motorlu ve yön veren teker, arkada iki serbest teker vardır.	Önden sürüşlü arabalar
		Dört adet yön belirleyen ve motorlu teker vardır.	Dört tekerli sürüşe sahip Hyperion (Carnegie Mellon Üniversitesi)
		Dört adet motorlu ve yön belirleyen taşıyıcı tekerler vardır.	Nomad XR4000
6		**Merkezde iki adet çekici (sürükleyici) teker, her köşede birer tane çok yönlü teker vardır.**	Terragator (Carnegie Mellon Üniversitesi)

Çizelge 2.1: (devam) Tekerlekli kara araçlarında çeşitli teker konfigürasyonları (Z.KurtY.L.Tezi)

	Güç verilmemiş çok yönlü teker (küresel, nakliye tekeri, İsveç tekeri)
	Motorlu İsveç tekeri (Stanford tekeri)
	Güç verilmemiş standat teker
	Motorlu standart teker
	Motorlu ve yön belirleyen nakliye tekeri
	Yön belirleyen standart teker
	Bağlı tekerler

Günümüzdede en çok kullanılan Ackermann Tekerlek modeli kullanılmıştır. Bu sistemde ön, arka veya dört tekere birden tahrik vererek çekiş sağlanmaktadır.



Şekil 2.1 : Holonomik olmayan Ackermann tekerlek sistemi

Çizelge 2.2 : Ackermann Tekerlek Sisteminde Kullanılan Semboller

L	**Aracın tekerlekler arası boyu**
V(k)	Aracın k anındaki hızı
Φ(k)	Aracın tekerleğine k anında uygulanan dönme açısı
θ(k)	Aracın xy koordinat düzlemine göre dönme miktarı
(x,y)	Aracın düzlemdeki konumu

Şekil 2.1 de gösterilen tahrik sistemi ile ilgili kinematik denklemler aşağıda veriilmiştir.

$$x_v = \begin{bmatrix} x_v \\ y_v \\ \theta_v \end{bmatrix} \quad \textbf{(2.1)}$$

$$\dot{x} = V.\cos\theta \quad \textbf{(2.2)}$$

$$\dot{y} = V.\sin\theta \quad \textbf{(2.3)}$$

$$\dot{\theta} = V\frac{\tan\theta}{L} \quad \textbf{(2.4)}$$

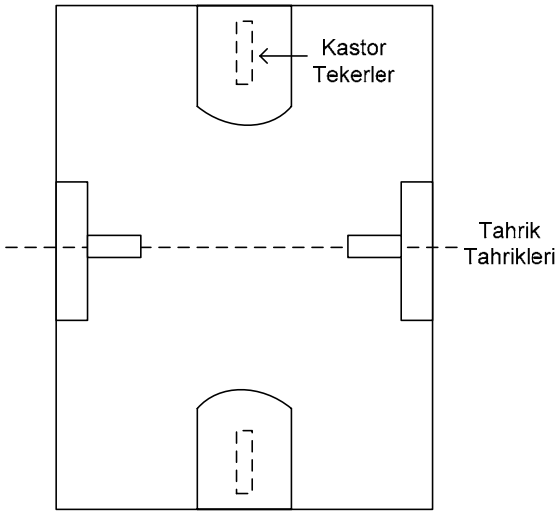
$$u(k) = \begin{bmatrix} V(t) \\ \varphi(t) \end{bmatrix} \quad \textbf{(2.5)}$$

Aracın k zamanında u(k) kontrol girişi uygulandığında u(t+1) zamanına ötelendiğindeki pozisyonu ise aşağıdaki gibi gösterilir.

$$\begin{bmatrix} x_v(t+1) \\ y_v(t+1) \\ \theta_v(t+1) \end{bmatrix} = \begin{bmatrix} x_v(t) + \Delta T.V(t).\cos(\theta_v(t)) \\ y_v(t) + \Delta T.V(t).\sin(\theta_v(t)) \\ \theta_v(t) + \frac{\Delta T.V(t).\tan(\varphi_v(t))}{L} \end{bmatrix} \quad \textbf{(2.6)}$$

Uygulamamızda diferansiyel tekerlek modeli tercih edilmiştir. Mobil robotta kullanılan model şekil 2.3 de gösterilmektedir. Şekilde gözüktüğü gibi kullanılan model çicelge 2.1 deki 3 numaralı modellere benzemektedir. Farklı olarak burada kastor teker sayısı önde ve arakda olmak üzere ikidir. Diferansiyel tekerlek modelini tercih etmemizin sebebi ana tekerleri yerleştiriliş motorlarla olduğu yerde yön değiştirme yeteneği olması ve fazla hareket gerektirmeden olduğu yerden geri geri çıkabilmesidir. Araçta bulunan kastor tekerler ise araç yükünü alma ve hareket serbestliği katma amaçlıdırlar.

Şekil2.2'de üretilmiş aracın modeli ve çizelge 2.3'te şekil 2.3'e ait semboller gözükmektedir.



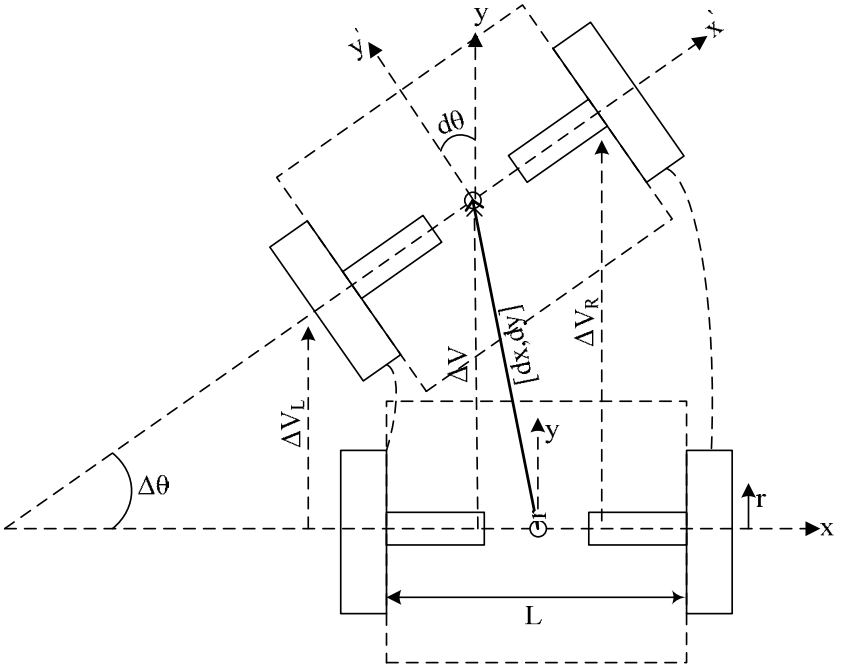
Şekil 2.2 : Önde ve Arkada birer serbest tekerlek, ortada servo motor eksenli 2 tane ana tekerlek

Araç t zamanından t+1 zamanına ötelenirse kinematik denklemleri şöyle yazılabilir [4].

$$c_m = \frac{\pi . D_n}{n . C_e} \tag{2.7}$$

$$\begin{bmatrix} dx \\ dy \\ d\theta \end{bmatrix} = \begin{bmatrix} \Delta V . \cos\big(\theta(t+1)\big) \\ \Delta V . \sin\big(\theta(t+1)\big) \\ \Delta\theta \end{bmatrix} \tag{2.8}$$

$$\left.\begin{matrix} \Delta V_L = c_m . N_R \\ \Delta V_R = c_m . N_R \end{matrix}\right\} \Longrightarrow \Delta V = \frac{\Delta V_R + \Delta V_L}{2} ; \Delta\theta = \frac{\Delta V_R - \Delta V_L}{L} \tag{2.9}$$



Şekil 2.3 : Diferansiyel tekerlek yapısının ötelenmesi

Çizelge 2.3 : Diferansiyel Tekerlek Sisteminde Kullanılan Semboller

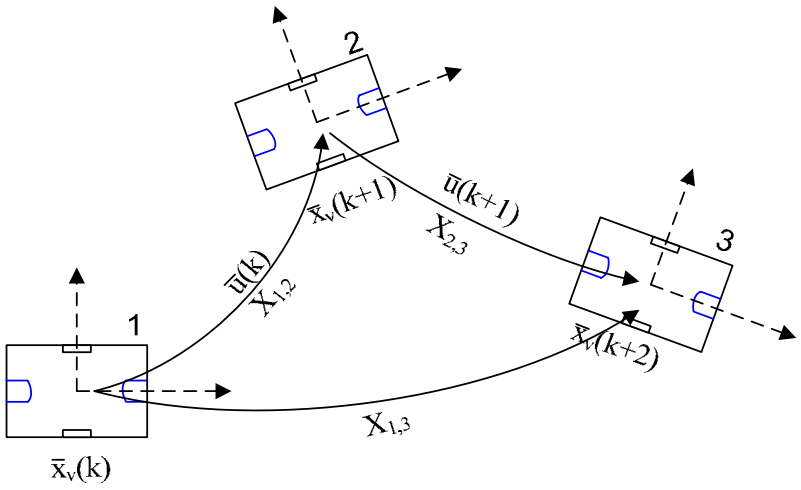
$\mathbf{D_n}$	nominal tekerlek çapı
$\mathbf{C_e}$	encoder çözünürlüğü
n	dişli oranı
$\mathbf{\Delta U_L}$	sol tekerlek hız değişimi
$\mathbf{\Delta U_R}$	sağ tekerlek hız değişimi
$\mathbf{\Delta\theta}$	aracın yönündeki açı değişimi
$\mathbf{N_L}$	sol tekerlek encoder değeri farkı
$\mathbf{N_R}$	sağ tekerlek encoder değeri farkı

$$\Delta V = r.\frac{\Delta\omega_R + \Delta\omega_L}{2}$$
$$\Delta\omega = r.\frac{\Delta\omega_R - \Delta\omega_L}{L} \quad \textbf{(2.10)}$$
$$\Delta\theta = \Delta\omega$$

$$x_v(t+1) = x_v(t) + \Delta V.\cos(\theta(t) + \Delta\theta)$$
$$y_v(t+1) = y_v(t) + \Delta V \sin(\theta(t) + \Delta\theta) \quad \textbf{(2.11)}$$
$$\theta(t+1) = \theta(t) + \Delta\theta$$

2.1.2 Diferansiyel Araç Modeli ve Hesaplanması

Mobil robotların çoğunda şürüş mekanizaması olarak diferansiyel sürüş sistemleri kullanılır. Bu sistemde eş bir eksene merkezlenmiş iki adet sürücü teker bulunur ve her teker bağımsız olarak ileri veya geri hareket yapabilir. Aracın tekerlkelerine değişik hızlar vererek ekseni etrafında dönme hareketi yaptırmak istediğimizde, robotun, tekerlerk eksenleri üzerinde bir noktada dönemesi gerekmektedir.



Şekil 2.4: Diferansiyel Araç Hareketi

Şekil 2.4 te verilen girişlere karşılık aracımızın yaptığı hareketi görmekteyiz. Aracımızın ilk pozisyonunda son pozisyona gideiblmesi ve bu noktalardaki lokasyonunun belirlenbilmesi için aracın koordinat eksenlerindeki dönüşümlerden yaralanacağız. Bu dönüşüm işlemlerinde tanımlayacağımız iki $\oplus$ ve Θ operatörü kullanacağız. Şekil 2.1 de görüldüğü gibi $x_{1,3} = x_{1,2} \oplus x_{2,3}$ dir. $x_1=(x_1,y_1,\theta_1)$ ve $x_2=(x_2,y_2,\theta_2)$ olarak tanımlansın [4].

$$\bar{x}_1 \oplus \bar{x}_2 = \begin{bmatrix} x_1 + x_2\cos\theta_1 - y_2\sin\theta_1 \\ y_1 + x_2\cos\theta_1 - y_2\sin\theta_1 \\ \theta_1 + \theta_2 \end{bmatrix} \quad \textbf{(2.12)}$$

$$\Theta x_1 = \begin{bmatrix} -x_1\cos\theta_1 - y_1\sin\theta_1 \\ x_1\sin\theta_1 - y_1\cos\theta_1 \\ -\theta_1 \end{bmatrix} \quad \textbf{(2.13)}$$

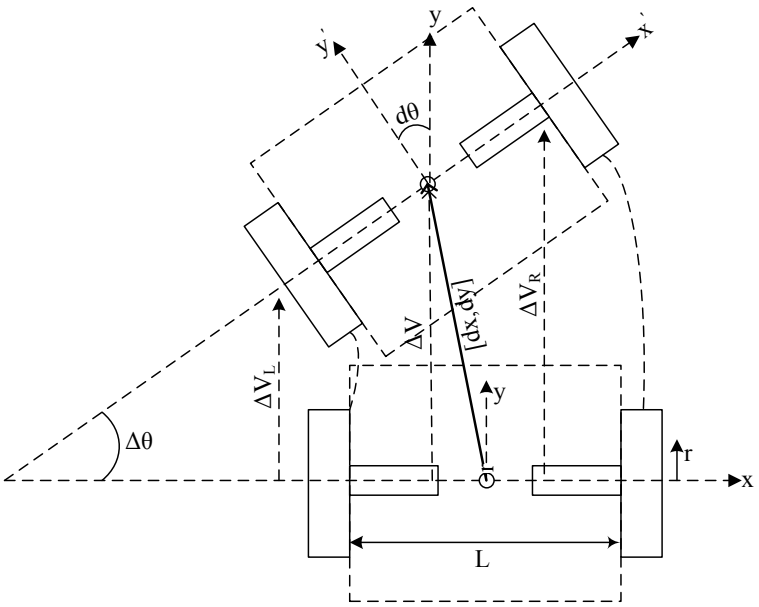
Denklem 2.12 ve 2.13 te aracın birinci eksendeki pozisyonlarının ikinci eksendeki pozisyonuna karşılıklarının ifadelerini görmekteyiz.Bir önceki hesapta $x_v(k)$ pozisyonunda olan aracımız, verilen u(k) hareketi ile bir sonraki $x_v(k+1)$ pozisyonuna gitmektedir. Bu yeni pozisyonun hesabu denklem 2.14 te gosterilmektedir.

$$x_v(k+1) = x_v(k) \oplus u(k) \quad \textbf{(2.14)}$$

Denkelm 2.14 te u(k) aracın ilk pozisoynunadan ikinci pozisyona hareket ettiren kontrol işaretidir ve denklem 2.15 ile hesaplanır.

$$u(k) = \Theta x_v(k) \oplus x_v(k+1) \quad \textbf{(2.15)}$$

Burada verilen denklemler bir araç modeli kapsamında oluşturulursa, yapılcaka simulasyonlar için ayrık hale getirilmiş bir lineer olmayan model halini alacaktır[3] deniz kavak.



Şekil 2.5: Diferensiyel Aracın Hareketi

Diferansiyel arac bir yerden bir yere öteleğinde aracın pozisyonunun bulunması için gerekli olan kinematik denklemler.

D_n: nominal tekerlek çapı

C_e: encoder çözünürlüğü

n: dişli oranı

ΔU_L: sol tekerlek hız değişimi

ΔU_R: sağ tekerlek hız değişimi

$\Delta\theta$: aracın yönündeki açı değişimi

N_L: sol tekerlek encoder değeri farkı

N_R: sağ tekerlek encoder değeri farkı

Denklem 7.10'da gösterilen hareket vektörü aracı 1.eksenden 2.eksene taşıyan kontrol işaretinin ürettiği parakete hesabı ile hesaplanan vektördür. Aracın hareketleri artımlı hareket olarak düşünülebilir. Yani araca uygulanan kontrol işaretinin etkisiyle aracın x yönündeki pozisyonu d_x , y yönündeki pozisyonu d_y ve aracın yönü ise $d\theta$ kadar değişir. Bu artırımlı hareketle aracın yeni pozisyonu önceki pozisyonu ile artırımlı hareketin transformasyonel toplanması ile elde edilir. Bu transformasyonel ifade denklem 7.11 ile gösterilmiştir. Aracın dinamik modeli bize aracın eski pozisyonu ve kontrol işareti ile yeni pozisyonunun bilgisini vermektedir. Bu bağlamda denklem 7.11 ile gösterilen formül aracın modelinin bir ifadesi olmaktadır. Aracın önceki pozisyonu ile uygulanan kontrol işaretinin etkisiyle elde edilen x,y ve θ'deki değişimin transformasyonel toplanması aracın yeni pozisyonunu belirleyen bir model olmaktadır.

$$u(k) = \begin{bmatrix} dx \\ dy \\ d\theta \end{bmatrix}$$

$$\bar{x}_v(k+1) = f(\bar{x}_v(k), \bar{u}(k)$$
$$\bar{x}_v(k+1) = \bar{x}_v(k) \oplus \bar{u}(k)$$

Aracın x,y ve θ'daki yer değiştirmeleri encoder'den alınan bilgiyle yapılmaktadır. Buda belli bir gürültüyü beraberinde getirmektedir. Bu gürültüyü sistem gürültü kovaryans matrisi Q ile ifade edeceğiz. Q x,y ve θ'daki değişimlerin hangi standart sapma ile gerçekleşeceğinin bilgisini taşımaktadır. Burada iki varsayım yapılmıştır. Bu varsayımlardan birincisi x,y ve θ'daki gürültülerin birbirinden bağımsız oldukları,

ikincisi ise bu gürültülerin gaussian olduklarıdır. Q sistem gürültü kovaryans matrisi Denklem 2.18'de verilmiştir. d-r ifadesi parakete hesabı'in kısaltmasıdır.

$$Q = \mathrm{U}_{\text{d-r}} = \begin{bmatrix} \sigma^2_{(d-r)x} & 0 & 0 \\ 0 & \sigma^2_{(d-r)y} & 0 \\ 0 & 0 & \sigma^2_{(d-r)\theta} \end{bmatrix} \quad \textbf{(2.18)}$$

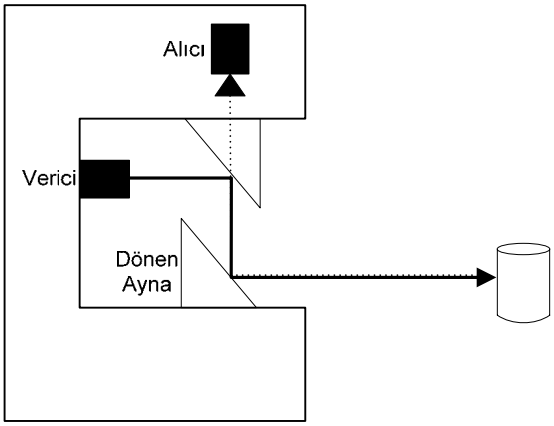
2.2 Araç Sensörleri ve Modelleri

Aracımızın etrafındaki nesneleri algılayabilmesi için laser tarayıcılar kullanıldı. Bu sensörler diğer sensörler kadar etrafın gürültüsünden etkilenmediklerinden tercih eidlmişlerdir. Uygulamamızda biri ön biri arka olmak üzere aracımızda iki adet lazer sensör kullanılmıştır. Böylece aracın hareketi sırasında ön ve arka taraf tanıları yer değiştirebilmektedir ayrıca 360 derece tarama sağlanmaktadır.



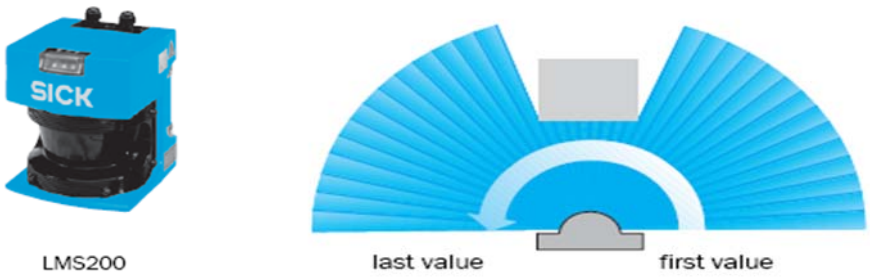
Şekil 2.6 : Mobil Robottaki 2 SICK LMS200'ün yerleşimi

Kullandığımız LMS200 tipi lazer sensörler neseneler lazer dalga boyunda ışık göndererek yansıyan ışığın uçuş süresine göre ölçüm yapar.Işık gönderildiğinde çalışmaya başlıyan sayaç yansıyan ışığı aldığında durur. Sayaç değeri gönderilen yol ile ilişkilendirilir ve nesnenin mesafi ortaya çıkar. Işığın nesneye çarpıp geri dönmesi ışık hızında olduğu için aynanın hareketinin ölçüme bir etkisi olmaz.



Şekil 2.7 : Lazer Sensörün Çalışmasının Yandan Görünümü

Sensörlerimiz ile 180°'lik görüş alanı taranmaktadır. Bu taram 0.25°'lik bir hassasiyetle yapılır. Uygulamamızda ise 1°'lik hassasiyeti ile 180°'lik taramalar yapılmıştır.



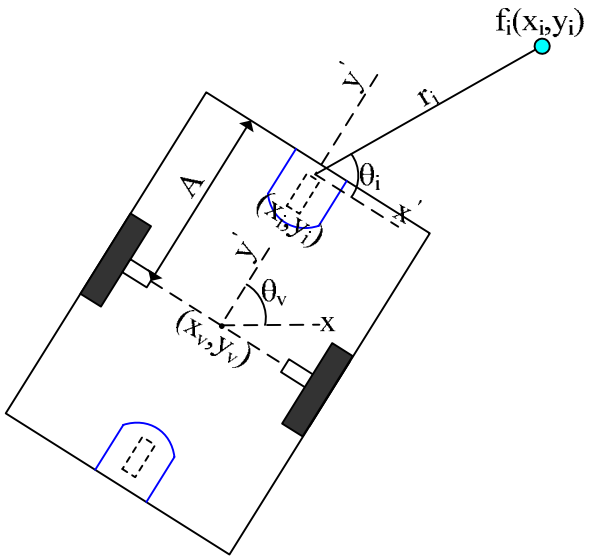
Şekil 2.8 : LMS200'ün tarama 13ms'lik 1° aralıkla 180°'lik taraması

Lazer tarayıcının, uygulamada ve simülasyonalrında kullancağımız hata kovarians matrisi daha önceden yapılan çalışmalarda hesaplanmıştır [5].

$$R = \begin{bmatrix} 0,58 & 0 \\ 0 & 0,16 \end{bmatrix} \qquad \textbf{(2.29)}$$

2.2.1 Sensör Modelleri

Bu bölümde SLAM amaçlı kullanılacak olan araç gözlem modeli oluşturulacaktır. Şekil 2.9 da görülmekte olduğu gibi araçta tek bir lazer sensörü vardır. Bu sensörün algıladığı işaretçi nesne ise $f_i(x_i, y_i)$ ile gösterilir.



Şekil 2.9: Diferansiyel Aracın Lazerle Gözlem Yapması

Araçtaki lazerle yapılan ölçümler lazer ve işaretçi nesne arasındaki mesafe r_i ve lazerle işaretçi nesne arasındaki x' eksenine göre olan açı θ_i 'yi vermektedir. Ölçüm vektörü r_i ve θ_i bilgilerini veren iki elemanlı bir vektördür. Lazer sensörünün araç üzerinde bulundupu nokta denklem 2.19 da verilmiştir [4].

$$(x_l, y_l) = (x_v, y_v+A) \tag{2.19}$$

Burada A sensörün araç koordinat merkezine olan uzaklığıdır.

$$\bar{z}(k) \cong \begin{bmatrix} r_i \\ \theta_i \end{bmatrix} \tag{2.20}$$

Denklem 2.20 deki ifadenin araç konumları cinsinden ifadesi denklem 2.21 de görülmektedir.

$$\bar{z}(k) = h(\bar{x}(k), \bar{w}(k)) \tag{2.21}$$

Gözlem modeli işaretçi nesne ve lazer arasındaki geometrik ilişki denklem 2.22 de verilmiştir. Gozlem modeli elde edilen r_i ve θ_i ölçümlerinden işaretçi nesne'ın x ve y eksenindeki değerleri x_i ve y_i 'yi vermektedir.

$$\bar{z}(k) \cong \begin{bmatrix} \sqrt{(x_i - x_l(k))^2 + (y_i - y_l(k))^2} \\ \alpha tan^2\left(\frac{y_i - y_l(k)}{x_i - x_l(k)}\right) - \theta_v \end{bmatrix} \quad \textbf{(2.22)}$$

Denklem 2.22 de verilen denklemi cebirsel olarak çözüp x_i ve y_i değerlerinin matematiksel olarak ifadesini bulmak gereksizdir. Matlab bu işlemi pol2cart komutu ile yapmaktadır. İki nokta arasındaki mesafe ve açı değerlerinden x ve y eksenlerindeki değişim bu komutla bulunabilmektedir.

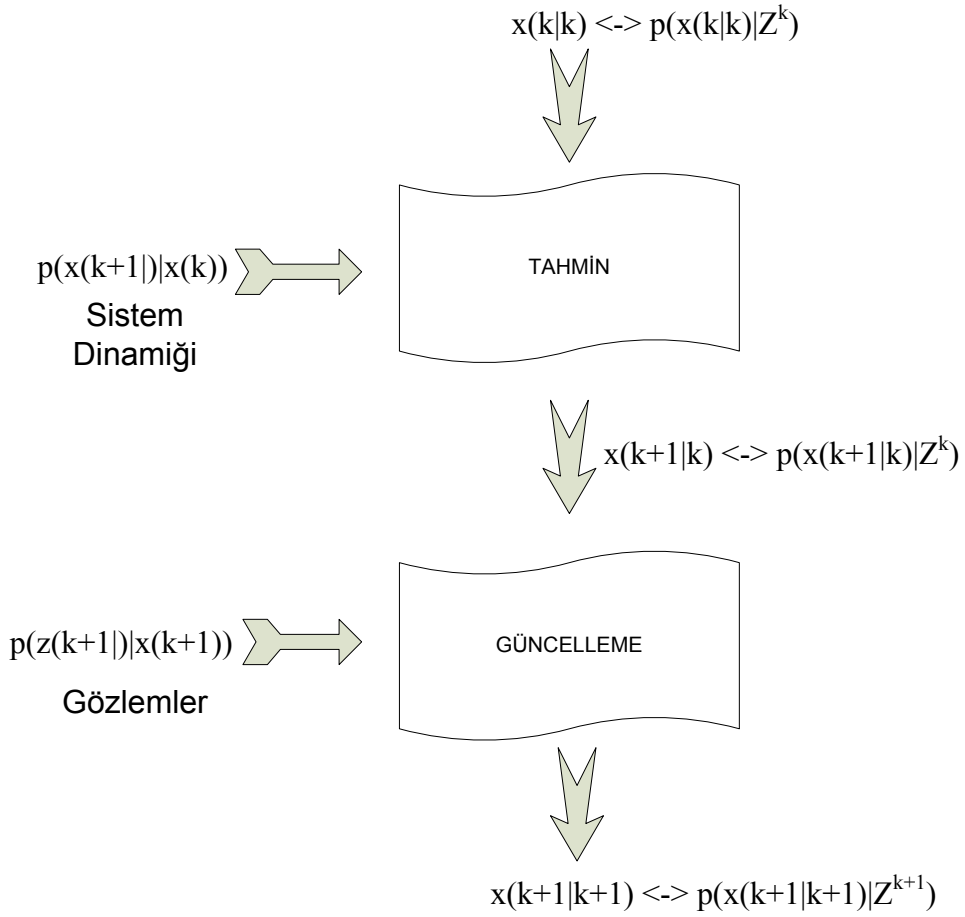
Gözlem modelinin kovaryans matrisi Denklem 2.23'te verilmiştir. Burada lazer mesafe gürültüsü ve açı gürültüsü arasında herhangi bir korelasyon bulunmadığı ve hatalarında gaussian olduğu varsayılmıştır. R matrisinin doğru bulunabilmesi için lazerin doğru modellenmesi gerekmektedir. Simulasyonda bu değerler rastgele seçilmektedir. Gerçek zamanlı uygulamalarda lazerle mesafe ve açı ölçümleri yapılarak gerçek değerlerinden ne kadar saptıkları belirlenmelidir. Bu hatalardan elde edilen dağılımın standart sapmaları hesaplanarak R kovaryans matrisi oluşturulacaktır.

$$R = \begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_\theta^2 \end{bmatrix} \quad \textbf{(2.23)}$$

3. OPTİMAL DURUM KESTiRİMİ

3.1 Genişletilmiş Kalman Filtresi

Kalman Filtre algoritma yapısı aşağıdaki şekilde verilmiştir [4].



Şekil 3.1: Kalman Filtre Yapısı

Şekil 3.1 deki yapıda Sistem (Araç) dinamiği ve gözlemler lineer olarak ele alınmıştır. Lineer modellerde yukarıdaki Kalman Filtre kestirimi uygulanabilir. Fakat gerçek hayatta da sistemler ve gözlemler non-lineerdir. Kalman Filtre denklemlerine

bu sistemlere uygulanamaz.. Nonlineer modellerdeki Kalman Filtre uygulamasına Genişletilmiş Kalman Filtre (GKF) denir.

Genişletilmiş Kalman filtresinde, sistem dinamiklerinin lineer olmadığı durum için filtreleme problemi incelenir. Lineer olmayan bir sistem düşünelim

$$x_{k+1} = f_k(x_k) + w_k \quad \textbf{(2.24)}$$

$$y_k = h_k(x_k) + v_k \quad \textbf{(2.25)}$$

Burada w_k ve v_k sıfır ortalama değer ve kovaryansı olan gaussian gürültülerdir.

Genişletilmiş Kalman filtresi optimal kestirime bir yaklaşım yapar.Lineer olmayan sistem dinamiklerine son duru kestirimi etrafında linnerleştrilmiş nonlineer sistem modeli ile yaklaşım yapılır. Bu yaklaşımın geçerli olabilmesi için, yapılan lineerleştirme işleminin tüm belirsizlik bölgelerinde durum kestirmi ile birlikte çok iyi yapılmış olması gerekmektedir.

Bir genişletilmiş kalman filtresi iterasyonu sıralı bir kaç adımdan oluşur. ilk olarak filtrelenmiş son durum kestirimi $\hat{x}(k|k)$göz önüne alınır. İkinci adımda sistem dinamiklerini $x_{k+1} = f_k(x_k) + w_k$ son durum kesitirmi etrafında lineerleştirme işlemi yapılır. Daha sonra Kalman filtresiin tahmin adımı linnerleştirilmiş sistem dinamiklerine uygulanır ve $\hat{x}(k+1|k)$ ve $P(k+1|k)$ elde edilir. Gözlem dinamikleri $y_k = h_k(x_k) + v_k$, $\hat{x}((k+1|k)$ etrafında lineerleştirilir. Ve Kalman filtresinin güncelleme adımı lineerleştirilmiş gözlem modelline uygulanır ve $\hat{x}(k+1|k+1)$ ve $P(k+1|k+1)$ elde edilir.

Çizelgi 3.1 denklemler Genişletilmiş Kalman Filtresi denklemleridir. Lineer Kalman denklemine çok benzemektedir. Lineer Kalman Filtre denklemlerinde jacobian'ın yerine F ve H geliyor.

GKF Kestirim kararlığı testleri durum vektörü ve kovaryans matrisini kestirilmesi için önemlidir. x_k doğru durum vektörü, $\hat{x}_k$ ve P_k ise kestirilen durum vektörü ve kovaryans matrisleri olsun.

$$E[x_k - \hat{x}_k] = 0 \quad \textbf{(2.32)}$$

$$E[(x_k - \hat{x}_k).(x_k - \hat{x}_k)^T] = P_k \quad \textbf{(2.33)}$$

Denklem 2.32 ve 2.33'yi sağlayan kestirim metodları kararlıdır diyebiliriz.

Çizelge 3.1 de Genişletilmiş Kalman Filtresiin dinamik yapısı gösterilmiştir[4].

Çizelge 3.1 : GKF Algoritması Denklemleri

Tahmin Adımı

$$\hat{x}(k|k-1) = f(\hat{x}(k-1|k-1), u(k), k)$$

(2.26)

$$P(k|k-1) = \nabla F_x . P(k-1|k-1) . \nabla F_x^{\mathrm{T}} + \nabla F_v . \mathrm{Q} . \nabla F_v^{\mathrm{T}}$$

(2.27)

$$z(k|k-1) = h(\hat{x}(k|k-1))$$

(2.28)

Güncelleme Adımı

$$\hat{x}(k|k) = \hat{x}(k|k-1) + W . \vartheta(k)$$

(2.29)

$$P(k|k) = P(k|k-1) - WSW^T$$

(2.30)

$$\vartheta(k) = z(k) - z(k|k-1)$$

$$W = P(k|k-1) . \nabla H_x^T + R$$

(2.31)

$$S = \nabla H^x . P(k|k-1) . \nabla H_x^T + R$$

$$\nabla F_x = \frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_m} \end{bmatrix}, \nabla F_U = \frac{\partial f}{\partial u} = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \cdots & \frac{\partial f_1}{\partial u_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial u_1} & \cdots & \frac{\partial f_n}{\partial u_m} \end{bmatrix}, \nabla H_x = \frac{\partial h}{\partial x} = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \cdots & \frac{\partial h_1}{\partial x_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_n}{\partial x_1} & \cdots & \frac{\partial h_n}{\partial x_m} \end{bmatrix}$$

3.2 Dağılımlı Kalman Filtresi

UKF ile GKF arasında temel fark, lineer olmayan bir modelin durum dağılımına yaklaşım tekniğinden ileri gelir.Julier tarafından geliştirilen DKF , lineer olmayan bir dönüşüm olan Dağılımlı dönüşüm den faydalanır. Bu dönüşümde durum olasılık

dağılımı, örneklenmiş bir sigma noktaları grubu tarafından ifade edilir. Bu sigma noktaları, durum dağılımının gerçek ortalama değer ve kovaryansını parametrelerle tanımlamak için kullanılır. Bu nedenle DKF simülasyon tabanlı lokal linner olmayan filtreler ailesi içindedir, bir bütün olarak sigma nokta filtresi olarak tanımlanabilir. Gerçek bir nonlineer sistemden geçirildiklerinde bu noktalar, bir sonraki ortalama değer ve kovariansı, herhangi bir doğursalsızlık için, ikinci dereceye kadar sağlar. Şimdi Dağılımlı dönüşümden bahsedelim.

3.2.1 Dağılımlı dönüşüm

Dağılımlı dönüşüm rastgele bir değişkenin lineer olmayan bir dönüşümde istatistiğini hesaplamak için kullanılan bir metottur. Dağılımlı dönüşüm, bir olasılık dağılımına yaklaşım yapmak rastgele seçilmiş lineer olmayan bir fonksiyona yaklaşım yapmaktan daha basittir, prensibine dayanır [6]. Ortalama değeri $\bar{x}$ ve kovariansı P_x rastgele bir değişkenin **x** lineer olmayan bir fonksiyonda $y = f(x)$ dönüşümüne bakalım. y nin istatistiğni hesaplamak için (2n+1) ölçümlendirilmiş sima vektörlerinden oluşan yeni bir X matrisi aşağıdaki gibi oluşturulur[8];

$$\begin{aligned} &\mathcal{X}_{0=\bar{x}} && W_0 = \frac{\kappa}{(n+\kappa)} && i = 0 \\ &\mathcal{X}_{i=}\bar{x} + \left(\sqrt{(n+\lambda)P_{xx}}\right)_i && W_i = \frac{1}{2(n+\kappa)} && i = 1, \dots., n \\ &\mathcal{X}_{i=}\bar{x} - \left(\sqrt{(n+\lambda)P_{xx}}\right)_{i-n} && W_i = \frac{1}{2(n+\kappa)} && i = n+1, \dots.,2n \end{aligned} \quad \textbf{(3.1)}$$

Burada K (kapa) ,sigma noktalarının ölçüm yönünü tanımlayan bir ölçüm parametresidir. $\left(\sqrt{(n+\lambda)P_{xx}}\right)_i$ ise matris karekökünün i ‘ inci satırı veya sütunudur. Wi ler ise i inci nokta ile ilgili ağırlıklardır. Bu sigma point vektörlerine bir lineerleştirme yapmadan nonlineer dönüşüm yaprılır.

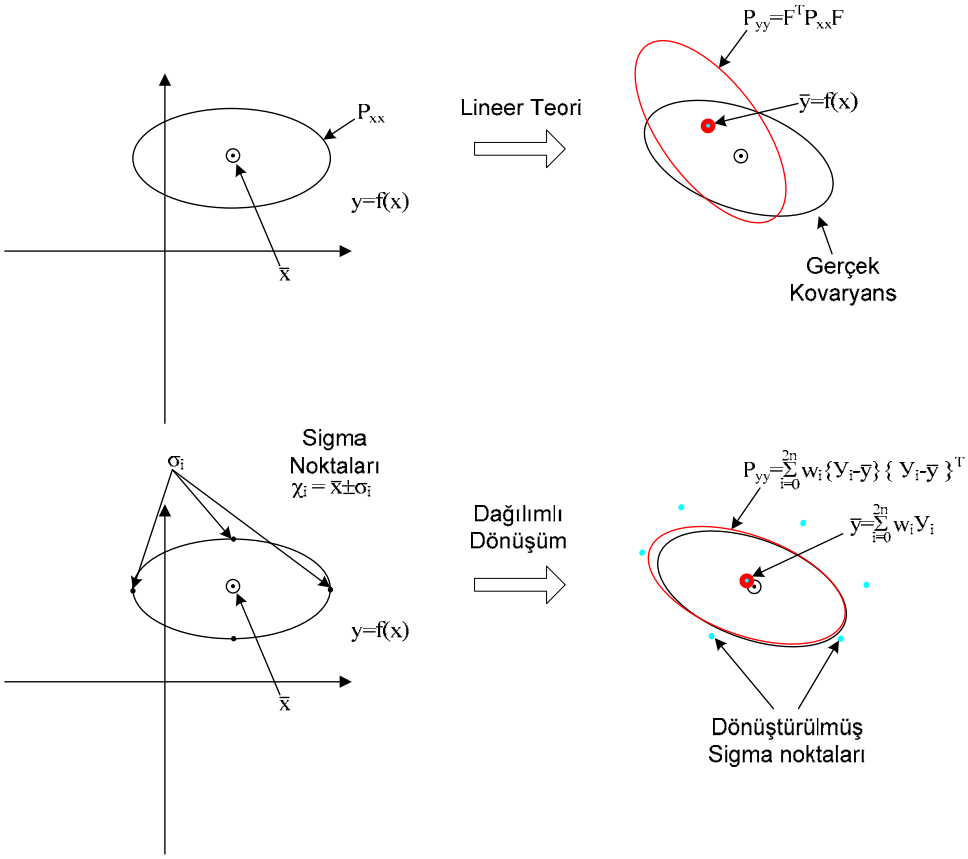
$$y_i = h(\mathcal{X}_i) \qquad i = 0, \dots \dots, 2n \quad \textbf{(3.2)}$$

Daha sonra y nin kestirilmiş ortalama değer ve kovariansı bu ağrılıklı örnek ortalama değer ve kovarians kullanarak hesaplanır.

$$\bar{y} = \sum_{i=0}^{2L} W_i^{(m)} Y_i \quad \textbf{(3.3)}$$

$$P_{y=} \sum_{i=0}^{2L} W_i^{(c)} (Y_i - \bar{y})(Y_i - \bar{y})^T \quad \textbf{(3.4)}$$

Ortalama değer ve kovariansın bu kestirimi, rastgel bir nonlineer fonksiyonnun Taylor açılımında, ikinci dereceye kadar doğru sonuçlar verir. Şekil 3.1de 2 boyutlu Dağılımlı dönüşümüne bir örnek gösterilmiştir.Alttaki şekil sigma nokta yaklaşımıyla yapılan Dağılımlı dönüşüm ile elde edilmiş ortalama değer ve kovariansı gösteririken, üsteki şekilse tek bir noktada etrafında lineerleştirme yaklaşımıyla hesaplanmış olanı gösterir. DD yaklaşımı Gausian girişler için üçüncü dereceye kadar hassasiyet gösterir.Bude tahmin edilen ortalama değer ve kovariansın gerçeğe o kadar yaklaşmasını sağlar. Buna karşılık olarak GKF deki lineerleştirme metodu yalnızca birinci dereceye kadar hassasiyet gösterir ve büyük hatalara yol açar. Dönüşümde kullanılan κ, yüksek dereceli yaklaşımlarda ince ayar yapmak için ekstra bir serbestllik derecesi katar. Çok boyutlu sistemlerde, eğer x bir gausian dağılım ise κ =3-n olarak kullanılır.Sigma noktasının, ortalama değerden uzaklığı $|\quad - \bar{}|$, $\sqrt{\quad +}$ ile orantılıdır.k=0 olduğunda mesafe $\sqrt{\quad}$ ile orantılıdır, k>0 olduğunda noktalar ortalama değer den daha uzakta bulunur ve k<0 olduğunda sigma noktaları ortalam değere yaklaşır.k nın positif veya negatif olabilmesine rağmen, negative k hesaplanan kovarians matrisinin özdeğerlerinin negatif olmasıyla sonuçlanabilir. Bu sorunun üstesinden gelebilmek içn ölçümlendirilmiş Dağılımlı dönüşüm geliştirilmiştir[7].



Şekil 3.2: Ortalama ve Kovaryans hesabı.

3.2.2 Ölçümlendirilmiş dağılımlı dönüşüm

Bu bölümde pozitif olmayan kovarians matirslerinden kurtulmak için sigma noktalarının ölçümlendirilmesi gösterilecektir.Ölçümlendirilmiş Dağılımlı dönüşüm, orjinal sigma noktlarını aşağıdaki kurala uyacak şekilde yenileriyle değiştirir.

$$\mathcal{X}_i' = \mathcal{X}_i + \alpha(\mathcal{X}_i - \mathcal{X}_0) \qquad \textbf{(3.5)}$$

Burada α sigma noktalrının yayılımını kontrol eden pozitif bir parametredir. Ve yüksek dereceli terimlerin filtreyi etkilememsi için yeteri kadar küçük değerler alabilir. Yukarıda verilen dönüşümle oluşturlan sigma noktaları kümesi S=(W,X) olsun, bu kümenin ölçümlendirilmiş S'=(W',X') haline dönüştürülülmesi için aiağıdaki formül kullanılır;

$$W_i' = \begin{cases} W_0/\alpha^2 + (1 - 1/\alpha^2) & i = 0 \\ W_i/\alpha^2 & i \neq 0 \end{cases} \qquad \textbf{(3.6)}$$

y nin tahmin edilen ortalama değer ve kovarians verileri için ise;

$$\mathcal{Y}_i' = h(\mathcal{X}_i') \quad \textbf{(3.7)}$$

$$\bar{y}' = \sum_{i=0}^{2n} W_i' \mathcal{Y}_i' \quad \textbf{(3.8)}$$

$$P_{yy}' = \sum_{i=0}^{2n} W_i' \{\mathcal{Y}_i' - \bar{y}'\}\{\mathcal{Y}_i' - \bar{y}'\}^T + (1 + \beta - \alpha^2)\{\mathcal{Y}_i' - \bar{y}'\}\{\mathcal{Y}_i' - \bar{y}'\}^T \quad \textbf{(3.9)}$$

Bu şekilde gösterilir. Burada β, sıfırıncı sigma noktasının ağırlığını kovarians hesabına katarak,daha yüksek dereceli terimlerin etkilerini dahil eder. β =2 Gausian dağılımlar için yugun olan değerdir. α sigma noktalarının dağılımının boyutunu kontrol eder ve ufak bir sayı olmalıdır (0< α<1).Sigma noktalarını seçen formul ve α yeni bir parametrede birleştirilir;

$$\lambda = \alpha^2(n + \kappa) - n \quad \textbf{(3.10)}$$

Ölçümlendirilmiş sigma noktalarının yeni gösterimi aşağıdaki gibidir;

$$\begin{aligned} \mathcal{X}_0 &= \bar{x} && i = 0 \\ \mathcal{X}_i &= \bar{x} + \left(\sqrt{(n+\lambda)P_{xx}}\right)_i && i = 1, \dots, n \\ \mathcal{X}_i &= \bar{x} - \left(\sqrt{(n+\lambda)P_{xx}}\right)_{i-n} && i = n+1, \dots, 2n \end{aligned} \quad \textbf{(3.11)}$$

$$W_i^{(m)} = \begin{cases} \frac{\lambda}{n} + \lambda & i = 0 \\ \frac{1}{\{2(n+\lambda)\}} & i = 1, \dots, 2n \end{cases} \quad \textbf{(3.12)}$$

$$W_i^{(c)} = \begin{cases} \frac{\lambda}{n+\lambda} + (1 - \alpha^2 + \beta) & i = 0 \\ \frac{1}{\{2(n+\lambda)\}} & i = 1, \dots, 2n \end{cases} \quad \textbf{(3.13)}$$

$W_i^{(m)}$, ortalama değer için ağırlık, $W_i^{(c)}$ ise kovarians için olan ağırlıklardır. Ölçümlendirilmiş Dağılımlı transform, ikinci derece filtrelerle aynı performansı göstermisine, ve hesaplama yoğunluğu GKF ile aynı olmasına rağmen, gerçek uygulamalarda hesaplama yoğunluğunun ve hafıza ihtiyaçlarının azaltılması gerekir. Julier , indirgenmiş sigma noktaları isimli çalışmasında, n boyutlu bir durum uzayında, 2n+1 den n+1 e olan geri dönüş noktalarını minmize etmiştir[4]. Bu indirgenmiş sigma noktaları, basit(tek yönlü) sigma noktalrı olarak adlandırılır ve hesaplama yoğunlu bakımından daha etkin sonuçlar vermiştir.

α, β ve κ parametrelerinin seçimi ile ilgili bir takım yollar gösterilmiştir. [7]. Burada κ parametres bütünlüğü sağlamak için kullanılır, ama genellikle sıfır alınır. DKF nin orjinak algoritmasındaki tek serbest parametre κ'dır vesigma noktalarının durum

ortalamasından fazla uzaklaşmasını önlemek için kullanılmıştır. Ölçümlendirilmiş versiyon uygulamaya başladığında α parametresinin tanımlanmasıyla κ gereksiz kalmıştır. β parametresi Julier tarafından, dördüncü derecede kovaryans dağılımındaki hatanın en aza indirilmesi için kullanılmıştır. Julier ayrıca Gaussian dağılımlar için $\beta = 2$ olması gerektiğini göstermiştir. α parametresi sigma noktalarının ortalama etrafında ölçülendirmek için kullanılmıştır. Örnek olarak, ölçümsüz DKF algoritmasında σ_f şu şekilde hesaplanır;

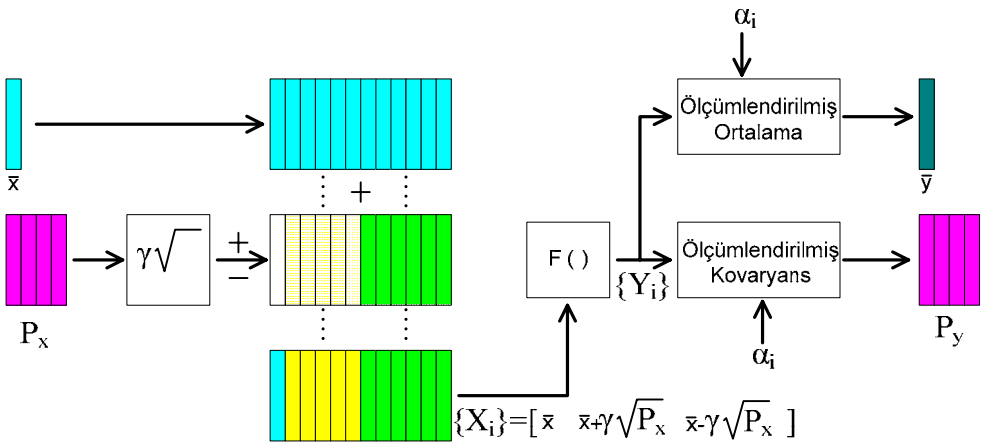
$$\sigma_f = \sqrt{(n^a + \kappa)} \qquad \textbf{(3.14)}$$

Burada görülmektedirki sigma noktalarının ortalama etrafında nasıl bir dağılım gösterdiği boyutsal olarak bağlantılıdır. Eğer κ sıfır olarak alınırsa sigma noktaları ortalamadan daha uzakta oluşturulurlar ve durum boyutu arttirilmiş olur.

Ölçümlendirilmiş versiyonda ise σ_f nin gösterimi;

$$\sigma_f = \sqrt{\alpha^2(n^a + \kappa)} \qquad \textbf{(3.15)}$$

Şeklindedir. Burada α sigma noktalarının, büyük boyutlu sistemlerde ortalamadan uzakta oluşturulmasını engeller. [0,1] arasında seçilmelidir. Ve boyutu [10^{-3} – 10^{-4}] aralığında olmalıdır [9].



Şekil 3.3: Dağılımlı Dönüşümün Şematik Gösterimi

3.3.3 Dağılımlı kalman filtresi

Bu bölüme kadar, ortalama değer ve kovariansı parametrelerle ifade etmek için dönüşümler gösterildi. Bu bölümde ölçümlendirilmiş Dağılımlı dönüşümün lineer olmayan bir filtreleme problemi üzerine uygulanmıştır. DKF , UT'nin dinamik ve ölçme modellerinde tekrarlı tahmin yapmak için kullanılan, basit bir uygulamasıdır.

DKF ayrık zamanlı lineer olmayan sistemler için geliştirilmiştir.Burada işlem gürültüsü eklenmemiştir, fakat ölçüm gürültüsü eklenmiştir. xk , k anında nx1 lik bir durum vektörü, yk ise mx1 lik bir ölçüm vektörü olduğunu düşünelim.wk, qx1 lik işlem gürültüsü vektörü ve vk rx1 lik ölçüm gürültüsü vektörü olsun, ve bunlar kovariansları Qk ve Rk olan gausian gürültüler olsun.İlk durum vektörü,işlem gürültüsü değişkenleri ve artırılmış kovarians matrisi ile birlikte, artırılmış durum vektörü olarak yeniden tanımlanır.

$$x_k^a = \begin{bmatrix} x_k \\ w_k \end{bmatrix}, P_k^a = \begin{bmatrix} P_k & P_k^{xw} \\ P_k^{xw} & Q_k \end{bmatrix} \quad \textbf{(3.16)}$$

Burada Pk,xw işlem gürültüsü ve durumu tahminindeki hata arasındaki korelasyondur. Artırılmış durumun {2(n+q)+1} sigma point kümesi, 13 te verilen dönüşümün 15 e uyugulanmasıyla oluşturulur. Durum ilerlemesi için, öngörülmüş durum vektörü xk+1,- ve bu vektörün öngülmüş kovariansı Pk+1,- Dağılımlı dönüşüm kullanarak hesaplanır.

$$\mathcal{X}_{i,k+1} = f\left(\mathcal{X}_{i,k}^a, k\right) \quad \textbf{(3.17)}$$

$$\hat{x}_{k+1}^- = \sum_{i=0}^{2(n+q)} W_i^{(m)} \mathcal{X}_{i,k+1} \quad \textbf{(3.18)}$$

$$P_{k+1}^- = \sum_{i=0}^{2(n+q)} W_i^{(c)} \{\mathcal{X}_{i,k+1} - \hat{x}_{k+1}^-\}\{\mathcal{X}_{i,k+1} - \hat{x}_{k+1}^-\}^T \quad \textbf{(3.19)}$$

Benzer şekilde öngörülmüş gözlem vektörü yk+1,- ve kovariansı Pk+1,yy hesaplanır,

$$\mathcal{Y}_{i,k+1} = h\left(\mathcal{X}_{i,k+1}, k+1\right) \quad \textbf{(3.20)}$$

$$\hat{y}_{k+1}^- = \sum_{i=0}^{2(n+q)} W_i^{(m)} \mathcal{Y}_{i,k+1} \quad \textbf{(3.21)}$$

$$P_{k+1}^{yy} = \sum_{i=0}^{2(n+q)} W_i^{(c)} \{\mathcal{Y}_{i,k+1} - \hat{y}_{k+1}^-\}\{\mathcal{Y}_{i,k+1} - \hat{y}_{k+1}^-\}^T \quad \textbf{(3.22)}$$

Gözlem gürültüsü eklenebilir ve bağımsız olduğu için, innovasyon kovariansı Pk+1,vv şu şekidle gösterilir.

$$P_{k+1}^{vv} = P_{k+1}^{yy} + R_{k+1} \quad \textbf{(3.23)}$$

Ve çapraz korelasyon matriside

$$P_{k+1}^{xy} = \sum_{i=0}^{2(n+q)} W_i^{(c)} \{\mathcal{X}_{i,k+1} - \hat{x}_{k+1}^{-}\}\{\mathcal{Y}_{i,k+1} - \hat{y}_{k+1}^{-}\}^T \quad \textbf{(3.24)}$$

şeklinde tanımlanır.Bu adımdan sonra kalman kazancı hesaplanır.

$$\mathcal{K}_{k+1} = P_{k+1}^{xy}(P_{k+1}^{vv})^{-1} \quad \textbf{(3.25)}$$

Tahmin adımından sonra ise güncellenmiş durum tahmini ve kovariansı şu şekilde elde edilir.

$$\hat{x}_{k+1}^{+} = \hat{x}_{k+1}^{-} + \mathcal{K}_{k+1}\mathcal{v}_{k+1} \quad \textbf{(3.26)}$$

$$P_{k+1}^{+} = P_{k+1}^{-} - \mathcal{K}_{k+1}P_{k+1}^{vv}\mathcal{K}_{k+1}^{T} \quad \textbf{(3.27)}$$

DKF algoritması çizelge 3.2 de özetlenmiştir. Formullerdede gözükmektedir ki, ilişikili gürültü kaynakları, filtre algoritmasında herhangi bir değişiklik yapmadan verimli bir biçimde eklenebilir. Hem işlem hemde ölçün gürültüsünün eklenebilri olduğu özel durumlarda, artırılmış durum vektöründe ayarlama yapılarak hesaplama yoğunluğu azaltılabilir. Diğer bir deyişle sistemin durumunun, rastgele gürültü değişkenleri ile artırılmış olmasına gerek yoktur.Burada dikkat edilmesi gereken, DKF de GKF olduğu gibi Jacobian veya Hessian matrislerinin hesaplmasının yapılmamasıdır. DKF, GKF den daha kesin ve kararlı durum ve kovarians tahmini yapabilir. DKF sürekli olmayan fonksiyonlarında kestrimini yapabilir. Ve son olarak ta DKF nin paralel işlem yapmayada uygun olduğunu söyliyebiliriz.

Çizelge 3.2 : DKF Algoritması Denklemleri

Başlangıç Durumu:
$\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{\boldsymbol{a}} = \boldsymbol{E}[\boldsymbol{x}_{\boldsymbol{k}}^{\boldsymbol{a}}] = \boldsymbol{E}[\boldsymbol{x}_{\boldsymbol{k}}^{\boldsymbol{T}} \quad \boldsymbol{w}_{\boldsymbol{k}}^{\boldsymbol{T}}]^{\boldsymbol{T}} \in \mathfrak{R}^{\boldsymbol{n+q}}$
$\boldsymbol{P}_{\boldsymbol{k}}^{\boldsymbol{a}} = \boldsymbol{E}[(\boldsymbol{x}_{\boldsymbol{k}}^{\boldsymbol{a}} - \hat{\boldsymbol{x}}_{\boldsymbol{k}}^{\boldsymbol{a}}) \quad (\boldsymbol{x}_{\boldsymbol{k}}^{\boldsymbol{a}} - \hat{\boldsymbol{x}}_{\boldsymbol{k}}^{\boldsymbol{a}})^{\boldsymbol{T}}] = \begin{bmatrix} \boldsymbol{P}_{\boldsymbol{k}} & \boldsymbol{P}_{\boldsymbol{k}}^{\boldsymbol{xw}} \\ \boldsymbol{P}_{\boldsymbol{k}}^{\boldsymbol{xw}} & \boldsymbol{Q}_{\boldsymbol{k}} \end{bmatrix}$
Sigma Noktaları
$\boldsymbol{\sigma}_{\boldsymbol{k}}^{\boldsymbol{a}} = \sqrt{(\boldsymbol{n+q+\lambda})\boldsymbol{P}_{\boldsymbol{k}}^{\boldsymbol{a}}}$

Çizelge 3.2: (devam) DKF Algoritması Denklemleri

$\boldsymbol{\mathcal{X}}_{i,k}^{a} = ([\hat{\boldsymbol{x}}_{k}^{a}\hat{\boldsymbol{x}}_{k}^{a} + \boldsymbol{\sigma}_{k}^{a}\hat{\boldsymbol{x}}_{k}^{a} - \boldsymbol{\sigma}_{k}^{a}])_{i}$
İlerleme
$\boldsymbol{\mathcal{X}}_{i,k+1} = \boldsymbol{f}(\boldsymbol{\mathcal{X}}_{i,k}^{a}, \boldsymbol{k})$
$\hat{\boldsymbol{x}}_{k+1}^{-} = \sum_{i=0}^{2(n+q)} \boldsymbol{W}_{i}^{(m)} \boldsymbol{\mathcal{X}}_{i,k+1}$
$\boldsymbol{P}_{k+1}^{-} = \sum_{i=0}^{2(n+q)} \boldsymbol{W}_{i}^{(c)}\{\boldsymbol{\mathcal{X}}_{i,k+1} - \hat{\boldsymbol{x}}_{k+1}^{-}\}\{\boldsymbol{\mathcal{X}}_{i,k+1} - \hat{\boldsymbol{x}}_{k+1}^{-}\}^{\boldsymbol{T}}$
$\boldsymbol{\mathcal{Y}}_{i,k+1} = \boldsymbol{h}(\boldsymbol{\mathcal{X}}_{i,k+1}, \boldsymbol{k+1})$
$\hat{\boldsymbol{y}}_{k+1}^{-} = \sum_{i=0}^{2(n+q)} \boldsymbol{W}_{i}^{(m)} \boldsymbol{\mathcal{Y}}_{i,k+1}$
$\boldsymbol{P}_{k+1}^{vv} = \sum_{i=0}^{2(n+q)} \boldsymbol{W}_{i}^{(c)}\{\boldsymbol{\mathcal{Y}}_{i,k+1} - \hat{\boldsymbol{y}}_{k+1}^{-}\}\{\boldsymbol{\mathcal{Y}}_{i,k+1} - \hat{\boldsymbol{y}}_{k+1}^{-}\}^{\boldsymbol{T}} + \boldsymbol{R}_{k+1}$
$\boldsymbol{P}_{k+1}^{xy} = \sum_{\text{İ}=0}^{2(n+q)} \boldsymbol{W}_{i}^{(c)}\{\boldsymbol{\mathcal{X}}_{i,k+1} - \hat{\boldsymbol{x}}_{k+1}^{-}\}\{\boldsymbol{\mathcal{Y}}_{i,k+1} - \hat{\boldsymbol{y}}_{k+1}^{-}\}^{\boldsymbol{T}}$
Güncelleme
$\boldsymbol{\mathcal{K}}_{k+1} = \boldsymbol{P}_{k+1}^{xy}(\boldsymbol{P}_{k+1}^{vv})^{-1}$
$\hat{\boldsymbol{x}}_{k+1}^{+} = \hat{\boldsymbol{x}}_{k+1}^{-} + \boldsymbol{\mathcal{K}}_{k+1}\boldsymbol{v}_{k+1}$
$\boldsymbol{P}_{k+1}^{+} = \boldsymbol{P}_{k+1}^{-} - \boldsymbol{\mathcal{K}}_{k+1}\boldsymbol{P}_{k+1}^{vv}\boldsymbol{\mathcal{K}}_{k+1}^{T}$

4. SLAM

Eş zamanlı haritalama ve lokalizasyon (SLAM) problemi, bilinmeyen bir çevrede bilinmeyen bir konuma bulunan mobil robotun, bu çevrenin haritasını çıkartıp aynı anda da kendi konumunu belirliyebilmesi, sorusunu sorar. SLAM probleminin çözümü robotik araştırmları için büyük önem içermektededir. Çünkü robotun tam anlamıyla otonom çalışmasını sağlayacak bir araçtır.

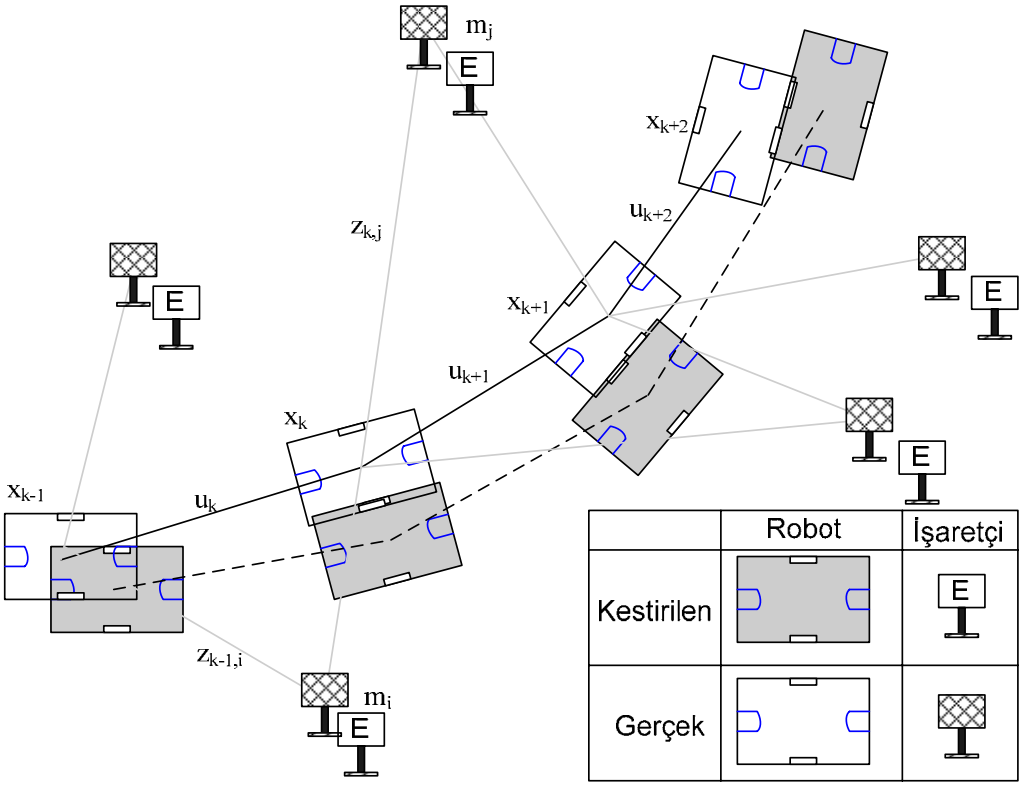
Geçtiğimiz on yıl içinde SLAM problemi üzerinde çeşitli sonuçlar üretilmiştir. Ve bu çözümler, kapalı alan, dış mekan, su altı ve hava gibi sistmelerde uygulamaya tabii tutulmuştur. Teorik olarak SLAM çözülmüş bir problem gibi göz önüne alınabilir. Fakat daha zengin haritalar ve doğru konumlamalar yapmak için çalışmalar devam etmektedir.

Bu bölümde SLAM problemlerinden bahsedilecektir. Teorik olarak SLAM in, standart Bayesian formundaki yapısı incelenecektir.

4.1 SLAM Probleminin Yapısı

SLAM, mobil robotun bir ortamın haritasını çıkartıp, kendi poziyonunu bulmak içinde bu haritayı kullanığı bir prosestir. SLAM de aracın yörüngesi ve işaretçi nesnelerin konumları araç hareket ederken kestirilir, bunu için geçmiş bir konum bilgisine ihtiyaç yoktur.

Herhangi bir ortamda hareket eden ve üzerindeki sensörler sayesinde sayısı belirsiz işraretçilerin bir birleriyle bağıntılı gözlemlerini yapan bir mobil robot düşünelim. t anında xt(aracın konumunu ve oryantasyonunu belirten durum vektörü),DD(control işareti [t-1 anında uygulanı aracı t anıdaki xk durumuna taşır]),mi(i inci işaretçi nesnenin konumunu belirten vektör),zik (i inci işaretçi ve t anında yapılmış gözlem) X0t(daha önceki araç konumları),U0t (daha önceki kontrol girişleri),m (tüm işareçilerin kümesi),Z0t (yapılan tüm gözlemler) olsun .



Şekil 4.1: Otonom diferansiyel araçta SLAM problemi

4.1.1 Olasılıksal SLAM

Olasıksal olarak SLAM aşağıda gösterilen dağılımın tüm t zamanları için hesaplanmasını gerektirir [10].

$$P(x_k, m|Z_{0,k}, U_{0,k}, x_0) \quad \textbf{(4.1)}$$

Bu olasılık dağılımı, t anında aracın başlangıç durumunuda içeren, yapılmış gözlemler ve girişler sonucundaki, işaretçi nesnelerin ve araç durumunu bir sonraki birleşik yoğunluğunu gösterir. Genel olarak SLAM problemlerinde rekürsiv çözümler istenir. k-1 $P(x_{k-1}, m|Z_{0,k-1}, U_{0,k}, x_0)$ dağılımına yapılan bir kestirimle başlar bunu girilen kontrol işateri u(k) ve gözlemler z(k) izler. Bu hesaplama Bayes teoremi ile yapılır. Bu hesaplamalar, durum geçişleri modelinin ve gözlem modelinin, kontrol girşi ve gözlemleri ayrı ayrı tanımlamasını gerektirir.

Gözlem modeli, aracın ve işaretçi nesnelerin konumu bilindiğinde, z(k) gözlemlerinin yapılması olasılığını tanımlar ve şu formda gösterilir;

$$P(z_k|x_k, m) \quad \textbf{(4.2)}$$

Araç konumu ve harita bir kere tanımlandığında, gözlemlerin verilen harita ve o anki arac durumundan yerine göre bağımsız olduğunu kabul etmek mantıklı bir yaklaşımdır.

Araç için hareket modeli, durum geçişlerinin olasılık dağılımı cinsinden tanımlanabilir.

$$P(x_k|x_{k-1}, u_k) \quad \textbf{(4.3)}$$

Burada görülen, durum geçişlerinin bir Markov süreci olduğudur. Bir sonraki durum x(k) sadece kendinden hemen önce gelen duruma x(k-1) ve kontrol girşine u(k) bağlıdır. Ve gözlemlerden ve haritadan bağımsızdır.

SLAM algoritmaları günüzmüde sıralı iki adımdan oluşmaktadır. Bunlardan birincisi zaman güncellenmesinin yapıldığı kestirim adımı, bir diğeri ise ölçüm güncellemesinin yapıldığı doğrulama adımıdır.

Zaman güncellemesi;

$$P\left(x_k, m\middle|Z_{0,k-1}, U_{0,k}, x_0\right) = \int P(x_k|x_{k-1}, u_k)$$

$$\times P\left(x_{k-1}, m\middle|Z_{0,k-1}, U_{0,k-1}, x_0\right)dx_{k-1} \quad \textbf{(4.4)}$$

Ölçüm güncellemesi;

$$P\left(x_k, m\middle|Z_{0,k}, U_{0,k}, x_0\right) = \frac{P(z_k|x_k, m)P\left(x_k, m\middle|Z_{0,k-1}, U_{0,k}, x_0\right)}{P\left(z_k\middle|Z_{0,k-1}, U_{0,k}\right)}$$

Denklem 4.4 ve 4.5,k anında yapılan tüm gözlemlere Z(0,k) ve tüm kontrol girişlerini U(0,k) hesaba katarak, robot durumu x(k) ve harita m için birsonraki birleşik dağılımı hesaplamada rekürsive bir yol izlenmesini sağlar. Tekrarlama ise araç modelinin ve gözlem modelinin bir fonksiyonudur.

Haritalama probleminin, koşullu yoğunluğu $P\left(m\middle|Z_{0,k-1}, U_{0,k}, X_{0,k}\right)$ hesaplayarak çözülmesi bir şeye değmez. Çünkü bu tür bir yaklaşımda, araç konumu x(k) tüm k zamanları için biliniyor kabul edilir. Ve m haritası, çeşitli konumlardan yapılan gözlemlerin birleştrilmesiyle oluşturulur. Aksine, haritalama problemi

$P0(x_k|Z_{0,k-1}, U_{0,k}, m)$ olasılık dağılımının hesaplanması ile çözülmelidir. Bu metodda işaretçi nesnelerin pozisyonları bilinmektedir. Ve amaç bu işaretçi nesnelere göre araç konumunun kestirimini yapmaktır.

Son zamanlarda lineer olmayan durum tahmini tekniklerinde GKF ye alternatif olarak türevsiz tahmin methodları geliştirildi. Bunların içinde Dağılımlı kalman filtresi(DKF), merkizi fark filtresi (CDF), ayrık farklar filtresi (DDF) gibi filtreler bulunmaktadır. Bu filtreler sigma noktalı filtreler (SPFs) olarak adlandırılır ve simülasyon tabanlı lineer olmayan filtreler grubundadırlar.DKF, ayrık olarak seçilmiş sigma noktaları ile ortalama değer ve kovariansın parametrelerle ifade etme prensibine dayanır.Ve sonraki ortalama değer ve kovarians değerleri herhangi bir lineerleştirme adımından geçmeden linner olmayan fonksiyonda işlenir[6].

4.2 SLAM Problemleri

İnsansız kara araçları eş zamanlı konum belirleme ve haritalamayı yaparken bir takım problemlerle karşılaşırlar. Haritaların doğru çıkarılması ve robotun kendi lokasyonunu doğru kestirebilmesi bu problemlerin aşılmasına bağlıdır. En sık karşılaşılan problemlerden ilki kestirim algoritmalarındaki matematiksel işlemlerin çevredeki algılanıp haritanın içerisine eklenen nesnelerin sayısından kaynaklanan haritanın sürekli büyümesinin getirdiği hesaplama problemi. Eş zamanlı lokalizasyon ve haritalama gerçek zamanlı çalışan bir işlem olduğu için hesaplamalarda harcanacak CPU zamanı çok önemlidir. İkinci problem veri ilişkilendirme problemidir. Araç gezinirken aynı nesneye ait birden fazla ölçüm yapmaktadır. Bu ölçümlerin daha önce gördüğü bir nesneye mi ait olduğu yoksa yeni gördüğü bir nesneye mi ait olduğu çok önemlidir. Eğer bu karar verme mekanizması yanlış çalışacak olursa harita yanlış çıkabiliceği gibi haritada boşu boşuna daha büyümüş olacak ve problem aynı zamanda hesaplama problemine de dönüşmüş olacaktır. İç ve dış sensör hassasiyetleri ve gürültüleride üçüncü problem olarak karşımıza çıkmaktadır. SLAM'ın dördüncü problemi çevredeki nesnelerin geometrilerinin doğru algılanma problemidir.

4.2.1 Hesaplama problemi

Eş zamanlı lokalizasyon ve haritalamanın matematiksel gösterimi, robot pozisyonu ve gözlemlenen işaretçi nesnelerin lokasyonlarını içeren birleşik durumun kestrimini

içerir. Bu durum ayrıcalıklı bir yapıya sahiptir. Çünkü süreç modeli sadece aracın duruşunu etkiler gözlem modeli ise araç ve işaretçi nesne çiftine bir referans oluşturur. SLAM algoritmalarındaki hesaplama karmaşıklığını azaltmak için bu özel yapıdan faydalanmak için bir çok yöntem geliştirilmiştir.

Geliştirilen yöntemler optimal veya korunumulu olmak üzere iki tipe odaklanmıştır. Optimal yöntemlerde gerekli hesaplama miktarını düşürürken aynı zamanda tam formdaki SLAM algoritmlarında ki kestirim ve kovaryansa değerlerini elde etmeye çalışılmıştır. Diğer algoritmalarda ise kestirim ve kovaryans hesaplmamalrında daha çok belirsilikler oluşur. Genellikle bu algoritmalar, daha az doğru sonuç vermesine rağmen hesapmlama karmaşıklığı bakımından daha verimlidir. Bu optimal çözümlerden daha kötü belirsizlik ve kovaryans sonuçları veren algoritmalar tutarsız olarak nitelendirlir ve SLAM problemleriiçin geçersiz çözümler olarak adlandırılırlar.

SLAM deki hesaplama prbolemini çözmek için zaman ve gözlem güncellemeleri yeinden formule edilmiştir. Zaman güncellemesinin hesaplanmasının karmaşıklığını azaltmak için durum arttırımı metodları kullanılabilir. Gözlem güncellemesinin hesaplarında sadeleştirme için ise güncellenmiş denklemlerin kısmi formları kullanılabilir. Bu iki yöntem hesaplama karmaşıklığı azaltılımş optimal bir SLAM kestirimi ile sonuçlanır. SLAM in standart durum uzayı gösteriminin yeniden formullendirilmesi, bilgileri taşıyan matrislerin seyreltilmesini sağlayabilir. Altharitalama yöntemleri haritaların, lokal koordinatsistemli bölgelere ayrılabilmesini sağlamıştır.Böylelikle güncellemeler yanlızca bu lokal bölgelerde yapılabilmektedir. Bu yöntemleri içeren bir çok çalışma yapılmıştır. Bunlardan bazıları Lokal Haritalama Algoritması (Chong ve Kleeman 1999),Decoupled Stokastik Haritalama (Leonard and Feder 2000), CLSF (Williams 2001), Suboptimal SLAM (Guivant ve Nebot 2001), Sparse Weight Filter (Julier 2001), Ayrıştırılmış Genişletilmiş Information Filter (Thrun 2003), Erteleme (Davidson 1998, Knight, Davidson ve Reed 2001), CEKF (Guivant ve Nebot 2001), Harita Ekleme (Tardos 2002).

4.2.1.1 Durum artırımı

Bir k anında, birleşik SLAM durum vektörü $\boldsymbol{x_k} = [\boldsymbol{x_{vk}^T}\ \boldsymbol{m^T}]^T$ olsun. Bu vektör robot pozisyonunu ve işaretçi nesnlerden oluşan haritayı içerir. Araç modeli sadece giriş işaretlerine göre posizyonlamayı değiştirir, harita da bir değşiklik yapmaz.

$$\boldsymbol{x_k} = \boldsymbol{f}(\boldsymbol{x_{k-1}}, \boldsymbol{u_k}) = \begin{bmatrix} \boldsymbol{f_v}(\boldsymbol{x_{vk-1}}, \boldsymbol{u_k}) \\ \boldsymbol{m} \end{bmatrix} \quad \textbf{(4.6)}$$

GKF SLAM algoritmasının en sade halinde kovaryans kestirimi denklem gibi hesaplanır.

$$\boldsymbol{P_{k|k-1}} = \nabla \mathbf{f}_{\boldsymbol{x}} \boldsymbol{P_{k-1|k-1}} \nabla \boldsymbol{f_x^T} + \nabla \boldsymbol{f_u} \boldsymbol{U_k} \nabla \boldsymbol{f_u^T}$$

(4.7)

Burada $\nabla \mathbf{f}_{\boldsymbol{x}}$ ve $\nabla \boldsymbol{f_u}$ jacobian ifadeleridir. Uk control işrateinin kovaryansıdır. Bu metod Jacobian ve kovaryans matrislerinin çarpma işlemlerinden dolayı işaretçi nesene sayısı kadar kübik bir karmaşıklığa sahiptir. Fakat, daha önceden söylediğimiz gibi, araç modeli yanlızca pozisyonlamayı etkiler. Bu yüzden kovaryans kestirimi, işsaretçi nesne sayısana bağlı, lineer bir karmaşıklığa sahip olarak, yeniden yazılabilir.

$$\boldsymbol{P_{k|k-1}} = \begin{bmatrix} \nabla \mathbf{f}_{\boldsymbol{vx}} \boldsymbol{P_{vv}} \nabla \boldsymbol{f_{vx}^T} + \nabla \boldsymbol{f_{vu}} \boldsymbol{U_k} \nabla \boldsymbol{f_{vu}^T} & \nabla \mathbf{f}_{\boldsymbol{vx}} \boldsymbol{P_{vm}} \\ \boldsymbol{P_{vm}} \nabla \boldsymbol{f_{vx}^T} & \boldsymbol{P_{mm}} \end{bmatrix}$$

(4.8)

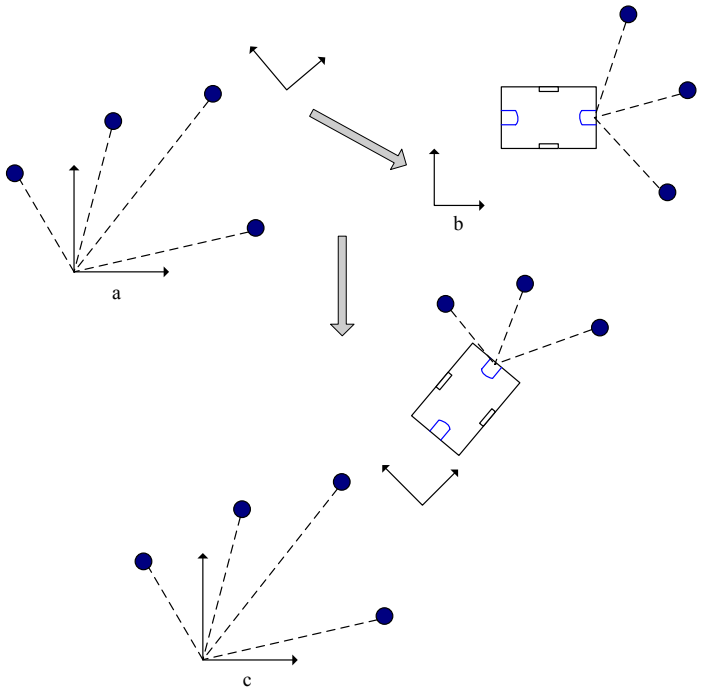
Durum vektörüne yeni bir landmark ekleme işlemi benze şekilde yapılabilir. Yeni landmarklar rbot pozisyonun ve gözlemin bir fonksiyonu olarak tanımlanabilir. Böylelikle artırılmış durumlar sadece, varolan durumların fonksiyonu olur.

4.2.1.2 Parçalanmış durumların güncellenmesi

SLAM in standart halinde, gözlemlerin güncelleştirilmesi adımında, her yeni gözlem yapıldığında bütün araç ve harita durumunlarını günceller. Bu durum GKF güncellemesinde landmark sayısına bağlı olarak ikinci derevedenbir karmaşıklık oluşturur. Bu durumdan kurutlmak için bir çok kısmi güncellme metodu geliştirilmiştir. Bütün metodlar optiaml kestirimler yapabilmektedir. İki temel tipi vardır. CGKF (Sıkıştırılmış Genişletilmiş Kalman Filter) ve CLSF (Constrained Local Submap Filter) . CGKF'de global haritanın lokal bir bölgesinde global koorinat sistemini referans alarak çalışlır. CLSF de ise llokal bölgede kısa zamanlı

bir alt harita oluşturulur. Standart GKF algoritmalarında hesaplama kompleksliği N haritadaki nesnelere ait x ve y pozisyon sayısı olmak üzere ve M = N+3 olmak üzere $O(M^3)$'dür. CGKF algoritmalarında ise N_a aktif durum sayısı olmak üzere hesaplama kompleksliği $O(2N_a^2)$'ye kadar azaltılmaktadır.

Altharita metodlarının bir çok avantajı vardır. İlk olarak, herhangi bir zaman aralığında güncellenmesi gerekn işsaretçi nesne sayısı yalnızca o bölgedeki altharitada sınırlılanmış landamarkların sayısı ile sınırlanır. Böylece gözlem güncellemesi toplam harita boyundan bağımsızdır. İkinci avantaj ise, lokal olarak referanslandırılmış bölgede daha az belirsizlik vardır. Böylece lineerleştirilirken yapılan yaklaşımlar azaltılmış olur. Son olarak altharitalama gurp doğrulama yöntemini kullanıp veri ilişkilendirmeyi güçlendirebilir. Şekil 4.2'de CLSF'ye ait bir harita görülmektedir.



Şekil 4.2 : CLSF. Ön Slam (b) deki local haritaya gore oluşturuluyor ve periyodik olarak (a) 'daki global haritaya kayıtlanarak (c)'deki optimal global kestrimi oluşturmaktadır

4.2.1.3 Seyreltme

GKF-SLAM in standart halinde kestirilen durum vektörünü ve gerçek durum vekörüne bağlı Gaussian olasılık yoğunluğunun ilk iki merkez momentiyle tanımlana

P kovaryans matrisini üretir. Bu gösterim informasyon vektörü y_k ve üretilen kovaryans matrisinin tersi Y_k informasyon matrisi ilede gösterilebilir.

$$Y_k = P_k^{-1} \quad \textbf{(4.9)}$$

$$\hat{y}_k = Y_k \hat{x}_k \quad \textbf{(4.10)}$$

Bu gösterimin avantajı çok büyük haritalarda normalize edilmiş informasyon matrisinin diagonal elemanlarının sıfıra çok yakın olmasıdır. Thrun bu sıfıra yakın elemanları sıfıra setleyip seyreltme işlemi yapan bir teknik sunmuştur. Bunun dışında çeşitli seyreltme teknikleri mevcuttur. Bu teknikler için Bailey ve Durrant-Whyte 'ın SLAM tutorial'ından faydalanılabilir. Bu teknikler matristeki sıfırlı elemanları seyrelttikleri için hesaplamada harcanacak zamanı azaltmış oluyorlar. SLAM de seyreltme işleminin kilit noktası durum artırımının bir seyretltme operasyonu olduğunu kavramaktır.

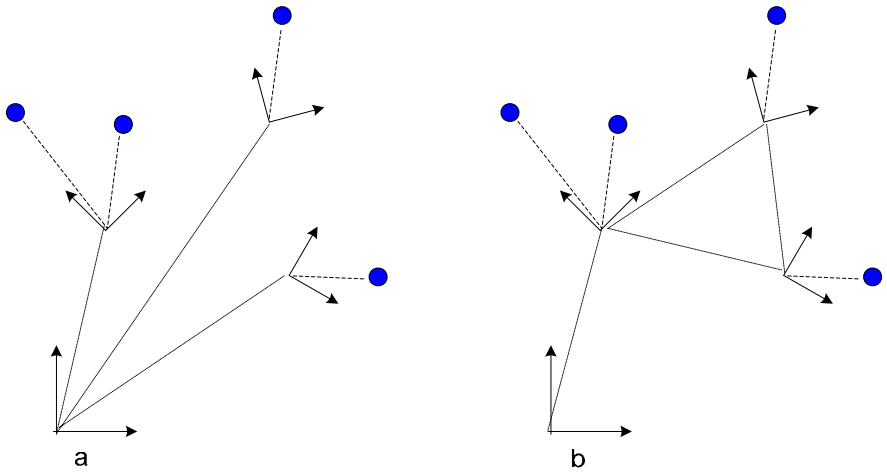
4.2.1.4 Altharitalar

Altharitalama metodları, ölçüm güncellemeleri sırasında işaretçi nesne sayısıyla oluşan ikinci derece hesaplama karmaşıklığını düşüren bir başka yöntemdir. Şekil 4.3'de görüldüğü gibi altharitalamanın iki tekniği vardır.Bunlardan birincisi global altharitalama ikincisi ise relatif altharitalamadır. İki tekniğin ortak noktası altharitalara ilişkin lokal koordinat düzlemleri belirlemeleri ve işaretçi nesne pozisyonlarını bu lokal düzleme göre güncelleştirilmeleridir.

Global altharitalama metodunda kestirimlerin global lokasyondaki yerleri lokal koordinatların ortak bir koordinata göre relatif olarak belirlenmesiyle elde edilmektedir. Relative Işaretçi nesne Represantation (RLR), hierarchical SLAM ve Constant time SLAM (CTS) gibi teknikler bu hesaplamaları yapmak için kullanılmaktadır. Bu yaklaşım ikinci derecdeki hesaplama karmaşıklığını lineer seviye indirmekle birlikte ortak koordinat düzleminin referans alınmasından dolayı büyük pozisyon belirsizliklerinden dolayı oluşan lineerizasyon hatalrında bir iyileştirme yapmamamaktadır.

Relatif altharitalamanın global alt haritalamadan farkı, ortak bir koordinat düzleminin olmamaısıdır. Bunun yerine komşusu olan lokal altharitanın koordinat düzlemine göre relatif olarak kendi lokal düzlemininin yerini belirler. Bütün lokal haritalar yanındaki lokal karitaya ait koordinat düzlemine göre relatif olarak kendi düzlemlerini belirleyerek bir ağ oluştururlar ve böylece global harita oluşturulmuş

olur. Bu yöntemle Global altharitalamada giderilemeyen lineerizasyon hatalarına çözüm getirilmiş oluyor. İlk olarak Chong ve Kleeman tarafından geliştirilen bu metod daha sonra CRSF (Constrained Relatif Submap Filter-Williams), Atlas framework ve NCFM (Network coupled feature maps) gibi çalışmalarda geliştirilmiştir.



Şekil 4.3: Global (a) ve relatif (b) altharitalama örnekleri

4.2.2 Veri İlişkilendirme problemi

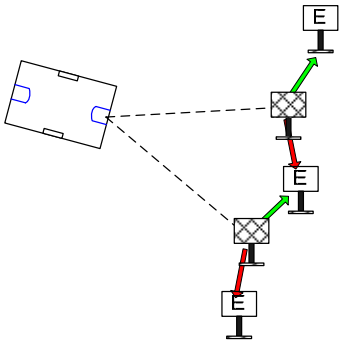
Veri ilişkilendirme, pratik SLAM uygulamalrında kritik bir önem taşır. Mobil robot kapalı çevrim şeklinde bir yrünge izliyorsa aynı yere geldiğini anlayabilmesi gerekmektedir. Eğer bu sağlanamazsa araç yürüngesinde sapmalar meydana gelir. Araç ortamda hareketini sürdürürken, veri ilişkilendirme esnasında, yeni öçlümler haritda bulunan eski işaretçi nesnelerle eşleştirilir. Böylelikle yeni ölçümler haritadaki yerlerini alır. Bu işlemden sonra geri dönüş yoktur. Bu durumdan anlaşılacağı gibi tek bir yanlış eşleştirmenin karita kestiriminin ıraksamasına neden olur. Konumlama algoritmalarının sonuçlarında ise çok büyük hatalar oluşturur. Veri ilişkilendirme problemminin çözümlenmesi için bir çok yöntem geliştirilmiştir.

Şekil 4.4'te bir işaretçi nesnenin atama problemi gözükmektedir.. Yanlış yapılan veri ilişkilendirmenin yanlış yapılması nedeni ile gözlenne işaretçi nesne başka bir nesne ile ilişkilendirmiştir.

4.2.2.1 Grup doğrulama

Hemen hemen bütün SLAM uygulamalarında, istatiksel doğrulama ayırıcı yönetemleri kullanarak veri ilişkilendirme yapılır. Bu yöntem, yanlış yapılan

ilişkilendirmeleri ayırmak için kullanılan hedef izleme metodlarından esinlenilmiştir. İlk yapılan SLAM uygulamalarında ölçümlerle işaretçi nesneler arasındaki veri ilişkilendirme, herbir nesnenin kestirilen konuma yakın olup olmadığı testinin yapılmasıyla gerçekleştirilirdi. Fakat bu yöntem, araç pozisyonundaki belirsizliğin fazla olması halinde güvenilir bir veri ilişkindirme aracı değildir.



Şekil 4.4: Bir Araçta Oluşabilecek Atama Problemi

Grup ayırıcı yöntemimin geliştirilmesiyle daha güvenilir veri ilişkilendirme yapılmıştır. Bu yöntemde bir çok eşleştirme aynı anda yapılmaktadır. Bu tekniğin bilinen iki formu vardır. İlki ağaç araştırmasında dayanan JCBB (Bileşik Uyumlu Dallanma and Bağlanma) ikincisi ise grafik araştırma algoritmalarını içeren CCDA (Combined Constraint Data Association)'dır. Grup ayırıcı eğer ayırıcıya yeterli sınırlandırılırma yapılırsa, ilişkilendirmede ki hatalar önemsiz kalır ve eğer doğru bir işaretçi nesne fiziksel olarak yanlış bir işaretçi nesne için yanlış ilişkilendirme yapılırsa, tutarsızlık seviyesi azaltılmış olacaktır. Daha karmaşık alanlarda bu büyük sorunlar oluşturur. Bunun için daha ileri derece veri ilişkilendirme metodları kullanılır. Bu yöntemlere bir örnek çok hipotezli veri ilişkilendirmedir.

4.2.2.2 Görünüm tanıma

Doğru ve güvenilir bir veri ilişkilendirme yapmak için geometrik kalıplar ve şablonlar tek başınna yeterli olmayabilir. Bu problemi aşmak için yapılan gözlemlerde renki şekil ve yapı olarak veriler arasında ilişkiler yapılarak desteklenir. SLAM probleminde Görünüm Tanıma kapalı bir çevrimde yapılacak veri ilişkilendirmelerini tek başına tahmin edebilecek yada klasik ayırıcılara işaretçi nesneların farklı özellik ve bilgilerini ekleyerek yardımcı olabilecek bir metoddur..

Daha önceden Görünüm tanıma ve Görüntü Benzerlik Metodu, görüntü veritabanları için index oluşturma ve topolojik haritalar içerisindeki yerleri tanıma amaçlı

geliştirilmişlerdir. Çok yakın geçmişte bu teknik üzerindeki Newman'ın kapalı çevrimi algılamadaki çalışmaları iki önemli yenilik getirmiştir. Sıralı görüntüler üzerinde benzerlik ölçümü hesaplanır ve özdeğer tekniği ile benzer ortak modlar kaldırılır. Bu yaklaşım sadece ilginç veya ortak olmayan benzerlikleri dikkate alarak hatalı pozitiflerin meydana gelmesini oldukça azaltır.

4.2.2.3 Çoklu hipotez veri ilişkilendirmesi

Çokhipotezli veri ilişkilendirme tekniği karmaşık alanlarda güvenilir hedef takibi için temel bir tekniktir. Ayrı ayrı veri ilişkilendirme hipotezi için ayrı bir yol kestirimi üreterek, gereğinden fazla yolları dallandırarak, veri ilişkilendirmedeki belirsizliği çözümlemeye çalışır. Oluşturulan yolların sayısı sistemin işleyiş hızına bağlı olarak limitlenir ve düşük-ihtimalli yollar hipotez ağacından budanır.

Özellikle büyük alanlarda güçlü SLAM uygulamaları için Çokhipotezli veri ilişkilendirme önemlidir. Örneğin, kapalı çevrimlerde, araç yapısal olarak benzer özellikler algıladığı çevrede şüphenilen çevrimler için ayrı hipotezler ve çevrim olmadığı hipotezini üretir. En büyük dezavantajı her bir hipotez için ayrı birer harita kesitirmi yaptığından işlemsel yük getirmesidir. Sparsification veya altharitalama methodları kullanılarak kolay işlenebilir bir çözüm üretilebilir. Hızlı SLAM algoritmaları herbir parçacığın kendi haritasını kestirmesinden dolayı birer doğal multihipotez çözümleridir.

4.2.3 Gürültü Problemi

Eş zamanlı lokalizasyon ve haritalamaaski problemlerden biriside gürültülerdir. Kullanılan sensör gürültülerinin yapıları incelenerek bu problem çözülebilir. GKF tabanlı SLAM algoritmalarında gürültülerin Gaussian gürültü olma zorunluluğu vardır. Gaussian olmayan gürültülerde parçacık veya informasyon filtresi gibi farklı kestirim algoritmaları kullanlılmalıdır. Bu sebeple SLAM algoritmasını kullanımına başlamadan önce lazer, enkoder, giroskop gibi algılayıcılara ait ölçümlerden gelen gürültüler incelenmelidir. Bu incelemeler sonucuna göre kestirim algoritmaları seçilmelidir. Sensörlerden bir çok veri alınıp gerçek değerle karşılaştırılıp gürültü tipi belirlenmeli, eğer gürültü gaussian gürültü ise kestirimin daha doğru olması için varyans değerleri saptanmalıdır. Böylece kestirim algoritmalarının daha doğru çalışmaları sağlanmış olur. Kestirim algoritmalarında ölçüm hataları arasındaki korelasyonlarda önemlidir. Ölçümlerdeki hataların biribirine bağımlı olup

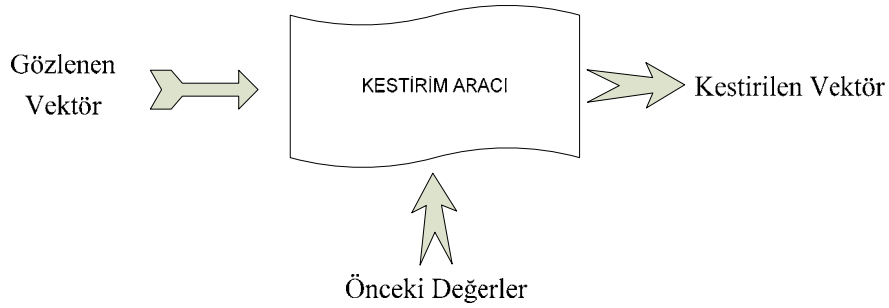
olmadığıda istatiksel olarak belirlenmelidir. Buna bağlı olarak da kovaryans matrisleri oluşturulmaktadır.

4.2.4 Çevre Problemi

Çevredeki nesnelerden gözlenen işaretçi nesnelar bir duvara, bir masaya, açık bir kapıya veya geometrik olmayan bir şekle ait olabilir. Lazer algıladığı sadece mesafe ve açı. Her algılanan nesneye ait mesafe ve açı bilgileri haritada bir yer tutar ve bir nokta ile gösterilirler. Bu noktalar birleştirilerek nesneler oluşturulmaya çalışılır. Yanlış noktalar böylece harita dışına atılabilir. Bu işleme birleştirme ve ayrıştırma işlemi diyoruz. X durum vektörü sadece sayılardan oluşmaktadır. Bu sayıların anlamlı hale gelmesi ve harita oluşturabilmesi için bu gösterimlerin düzgün ve doğru yapılması gerekmektedir. Ayrıca çevrenin dinamik olması da ayrı problemi beraberinde getirmektedir. Gerçek dünya statik değildir. Oysa biz uygulamalarımızda hep statik işaretçi nesneları algılamaya çalışıp onların haritalarını oluşturmaya ve güncellemeye çalışmaktayız. Park edilmiş araçlar, masalar, sandalyeler gibi nesneler sürekli yer değiştirirler. Bu tip nesneler algılanıp ayrıştırılarak statik bir işaretçi nesne gibi haritaya eklenmezler. Bunun çözümünü daha üst katmanlarda da yapılabilir. Haritalar bir kez oluşturulduktan sonra güncellenmeye devam edilebilir ve periyodik olarak haritalar arasında görüntü karşılaştırılmaları yapılarak farklılıklar tesbit edilip harita daha yalın hale getirilebilir.

4.3 GKF ile SLAM

Eş zamanlı haritalama ve lokalizasyon algoritmalarının ilk uygulamalarında Genişletilmiş Kalman Fitresi (GKF) kullanılmıştır. GKF-SLAM en iyi bilinen ve en iyi geliştirilen metodlardan birisidir. GKF tabanlı SLAM'a alternatif olarak Parçacık filtre tabanlı SLAM ve informasyon filtre tabanlı SLAM'lar da ortaya atılmışlardır. Genişletilmiş Kalman filteresinin temel hali Kalman filtresidir. Kalman filtresinin kullanılabilmesi için araç ve gözlem modellerinin lineer olması gerekmektedir. Gerçek hayatta bu modeller lineer değildir. Bunun için lineer olmayan modeller için Kalman Filtresinin kullanabilirliğini araştırılmış ve non-lineer modelleri lineer hale getirerek Kalman Filtresinin non-lineer modeller içinde kestirim metodu olarak kullanılabileceğini gösterilmiştir. Aşağıdaki şekilde kestirim araçlarının çalışmasına ait bir şekil görülmektedir [4].



Şekil 4.5: Kestirim Aracı Yapısı

GKF tabanlı SLAM tabanında araç modelini denklem deki tanımlamak vardır [11].

$$P(x_k|x_{k-1}, u_k) \Leftrightarrow x_k = f(x_{k-1}, u_k) + w_k \quad \textbf{(4.11)}$$

Burada f araç kinematiğini içeren modeldir. w_k ise Gaussian bir gürültüdür Gözlem modeli ise;

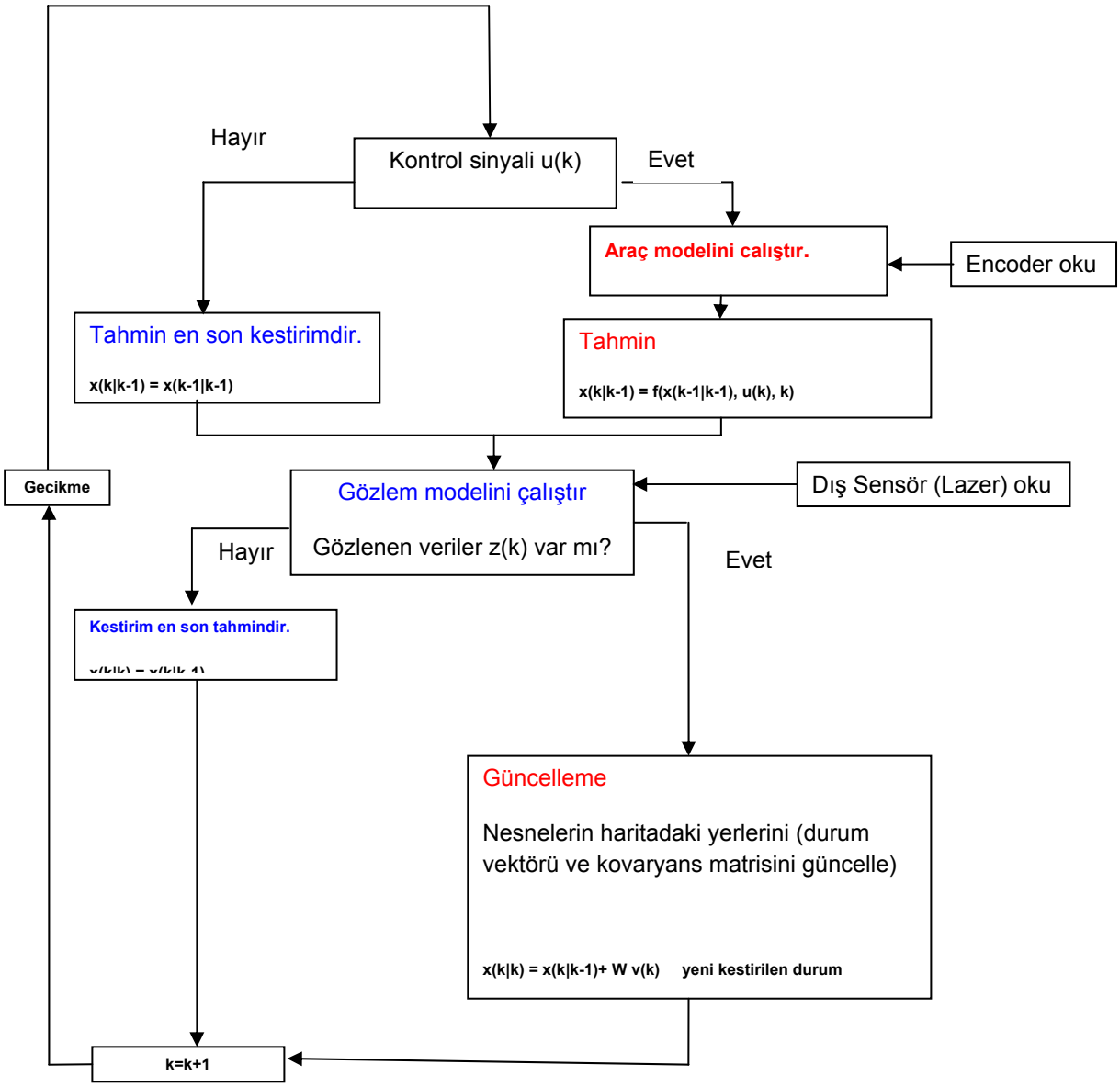
$$P(z_k|x_k, m) \Leftrightarrow z_k = h(x_k, m) + v_k \quad \textbf{(4.12)}$$

şeklindedir. Burada h gözlem modelini içeren fonksiyondur. v_k ise yine Gaussian bir gözlem gürültüsüdür. Bu tanımlamalaradan sonra yukarıda bahsettiğimiz temel genişletilmiş kalman filteri metodu kullanılarak ortalama değer ve kovaryans hesabı yapılır.Daha sonrada zaman güncelleştirme ve gözlem güncelleştirme adamıları uygulanır ve algoritma tamamlanır. GKF tabanlı SLAM uygulamasının akış diyagramı Şekil 4.4'te verilmiştir [4]. Algoritma ilk olarak araca kontrol işaretinin uygulanması ile başlar. Eğer kontrol işareti yoksa en son kestirim değeri tahmin olarak atanır ve lazerden gözlem verilerinin gelip gelmediğine bakılır. Eğer lazerden veri geliyorsa bu verilere göre araç ve çevrede bulunan işaretçi nesneların pozisyonları güncellenir. Araç için pozisyon bilgisi x,y ve aracın yönünü ifade eden θ değerleridir. Işaretçi nesnelar için pozisyon bilgisi ise x ve y değerleridir. Kontrol işareti ve aracın bir önceki kestirilen pozisyon değeri, aracın dinamik modeline uygulanarak aracın yeni pozisyonu tahmin edilir. Dinamik model nonlineer bir fonksiyondur. O yüzden kestirim aracı olarak Genişletilmiş Kalman Filtresi kullanılmıştır. Aracın dinamik modeli için girişler aracın bir önceki kestirilmiş pozisyonu, uygulanan kontrol işareti ve sistem gürültüsüdür. Bu dinamik modelden aracın ve işaretçi nesneların tahmini yeni pozisyonları x(k|k-1) elde edilir. Kovaryans matrisi ise aracın ve gözlemlerin belirsizlik elipsini vermektedir. Bu

belirsizlik elipsi ne kadar küçük olursa araç ve işaretçi nesne pozisyonlarıda o kadar gerçeğe yakın elde edilirler. Bu elipsleri küçültmek için elde edilen ölçüm verileri ile kovaryans matrisini de güncellememiz gerekmektedir. Güncellenen kovaryans matrisine P(k|k) kestirilmiş kovaryans denir. P(k|k-1) kovaryans matrisi tahmininde bir önceki kestirilen kovaryans matris değeri P(k-1|k-1) ve aracın nonlineer modellerinin lineerleştirilmesiyle elde edilen ∇F_X ve ∇F_v jacobian değerleri kullanılacaktır. Buradaki gürültünün kovaryans matrisi Q önemlidir. Sistemden gelen gürültü encoderin hassasiyeti ve gürültüsüne bağlıdır. Enkoder ne kadar hassas ve ne kadar az gürültüyle ölçüm yapıyorsa tahminlerimiz o kadar iyi olacaktır. Peki bu Q hata kovaryans matrisi nasıl elde edilecek. Araç x pozisyonunda hareket ettilerek encoderden gelen değer ile gerçek ölçüm arasında standart sapma hesaplanır. Bu işlem mümkünse birçok kere tekrarlanarak gaussian bir grafik elde edilir. Bu grafiğin standart sapması σ_x^2'i verir. σ_y^2 , σ_θ^2 değerleride aynı şekilde elde edilerek gürültü kovaryans matrisi Q elde edilir. ∇F_X ve ∇F_v değerleri aracın kinematik modelinde elde edilirler. Uygulama kısmında bu değerler ayrıntılı olarak verilmiştir. Kovaryans matrisi tahmininden sonra elde edilen tahmin durum vektörün gözlem modeline uygulanması ile gözlem tahmini z(k|k-1) de hesaplanır. Bu hesaplamalarla durum ve gözlem vektörü, kovaryans matrisi tahmin adımı tamamlanmış olur. Lazerden gelen yeni gözlem verileri varsa güncelleme adımına geçilir. Eğer lazerden herhangi bir veri gelmiyorsa en son tahmin değerleri kestirim değeri olarak kullanılırlar. Lazerden gelen yeni gözlem verileri ile durum vektörü ve kovaryans matrisleri güncellenerek kestirilirler. Kestirme işlemi aşağıdaki algoritmanın güncelleme kutucuğunda gösterilmiştir. Güncelleme adımı tamamlandıktan sonra algoritmanın bir çalışma zamanı tamamlanır ve tekrar başa dönülür.

GKF-SLAM probleminde, haritanın ıraksaması, monoton olarak harita kovaryans matirisinin determinantının ve işaretçi nesne çiftlerinin alt matrislerinin ıraksamasıyla ilişkilidir. İşaretçi nesnelerin ayrı ayrı varyansaları, robot paozisoynu ve gözlemlerin ilk belirsizlikler tarafından belirlenen alt sınıra yakınsar.

Gözlem güncelleme adımı, her gözlem yapıldığında, işaretçi nesnlerin ve birleşik kovaryans matrislerinin güncellenmesini içerir. Buda hesaplama karmaşıklığının işaretçi nesne sayısına bağıl olarak ikinic dereceden artmasına neden olur. Bu problemler yukarıda bahsettiğimiz yöntemlerle çözülmüştür.



Şekil 4.6: GKF Akış Diyagramı

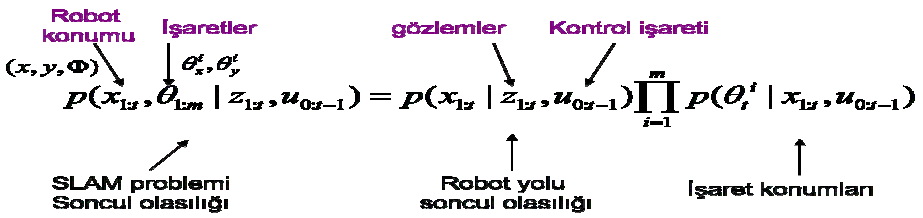
4.4 Fast SLAM

Çalışma mantıklarına göre sınıfladığımızda kestirim algoritmaların en poüplerleri GKF, maksümum benzerlik teknikleri ve paçaık filtreleridir. GKF 'nin öne çıkan güçlü tarafının robotun konumundan ve işaretçi nesnelerin yerinden, robotun kendi konumunun koşullu olasılığını çok iyi bir şekilde tahmin edebilmesi olduğundan, ancak zayıf tarafının, algılayıcı gürültülerinin gauss tipinde olması çok ve işaretçi nesne sayısını 'm' olarak alırsak, algılayıcının gözlemlerini yenileyip tek bir 'yeni' işaretçi nesne bulmasında bile kendi kovariyans matrisinin yeniden oluşturulması için işaretçi nesne sayısına bağlı ikinci derece bir hesaplama yükü getirir. Maksimum Benzerlik Algortimalarında ise sensör okumalarına bakılıp gerçeğe en yakın

haritaoluşturulur. ve ilişki kümeleri yaratarak robotun yeri tehmin edilir. Gutmann bu algoritmayı artımsal bir biçimde kullanarak robotun gezdiği döngüleri doğru bir biçimde tahmin etmeyi başarmıştır. Hahnel bu algoritmada veri ilişkilendirme kullanmıştır. Lakin, bu algoritma ile kullanılan veri ilişkilendirme ağaçları gerçek zamanlı çalışmalara izin vermemektedir. Ancak çoklu hipotezlerdeki benzerliklerin bulunmasında parçacık filtrelerinde yer alan maksimum benzerlik algoritması kullanılmıştır.

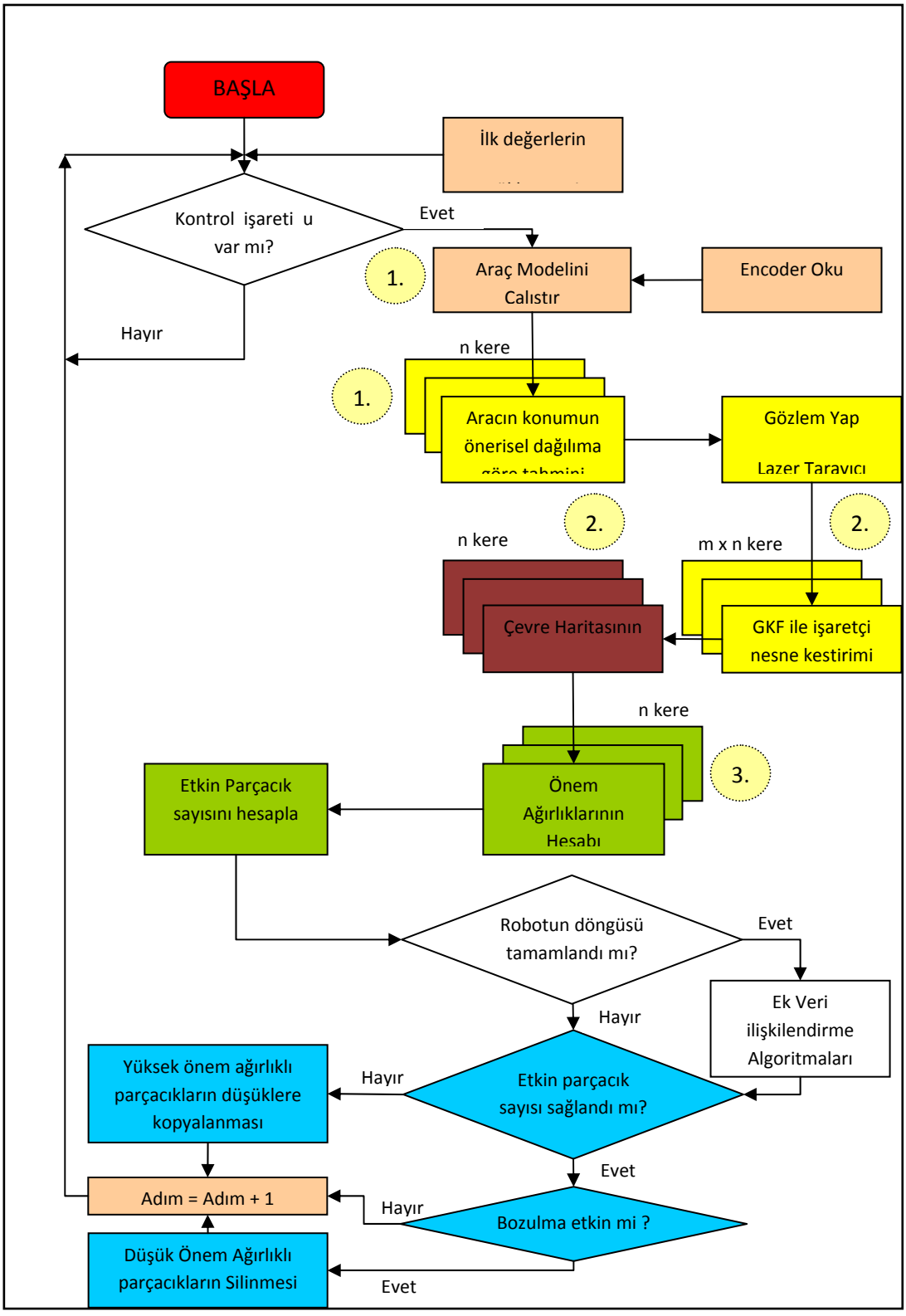
FastSLAM algoritması ise ardışıl olasılıksal SLAM'da mantıksal bir atlama yaratmıştır. Çünkü, doksanlı yıllarda kullanılan GKF'nin tek hipotezden oluşan lineer gauss tipi algoritmalarının tersine, FastSLAM ardışıl Monte Carlo tekniklerini kullanır ve çok hipotezli yapıya dönüştürmüştür. Bu yapı gauss tipi olmayan süreçleri ve dağılımlarını içermektedir. Ayrıca çok hipotezli olması algoritmayı kendiliğinden veri eşleştirmesi yapan bir model kılmıştır. GKF gibi parçacık filtreleri de Bayes kuralı temellidir. SLAM problemini çözmek için etkili bir yol olan Rao-Blackwized parçacıkfiltresi önerilmiştir. Bu çalışma, Montemerlo tarafından farklı işaretçi nesne haritaları kullanılarak genişletilmiştir.Şu anda da popüler olarak üzerinde geliştirmeler yapılmaktadır.

Görüldüğü üzere RBPF filtreleri hem GKF, hem maksimum benzerlik algoritması, hem de Monte Carlo Lokalizasyonu(MCL) algoritmalarını içermektedir. Parçacık filtresinin tanımını yapmak gerekirse, bu filtre parçacıkların dağılımlarını örnekleyen bir ardışıl Monte Carlo tekniğidir. Gauss dağılımlı olmayan ve doğrusal olmayan süreçlerin tahmin edilmesinde kullanılır. Kalman filtresindeki amaç çevredeki işaretlerin yerini tahmin etmektir,bu tahminlere bağlı olaraksa mobil robotun yerini tahmin etmektir. Oysa parçacık filtresinin amacı, mobil robotun geçeceği olası yolları tahmin etmektir. Ortam büyüklüğüne uygun parçacık sayısı ile optimal Bayes kestirimine yaklaşılır.



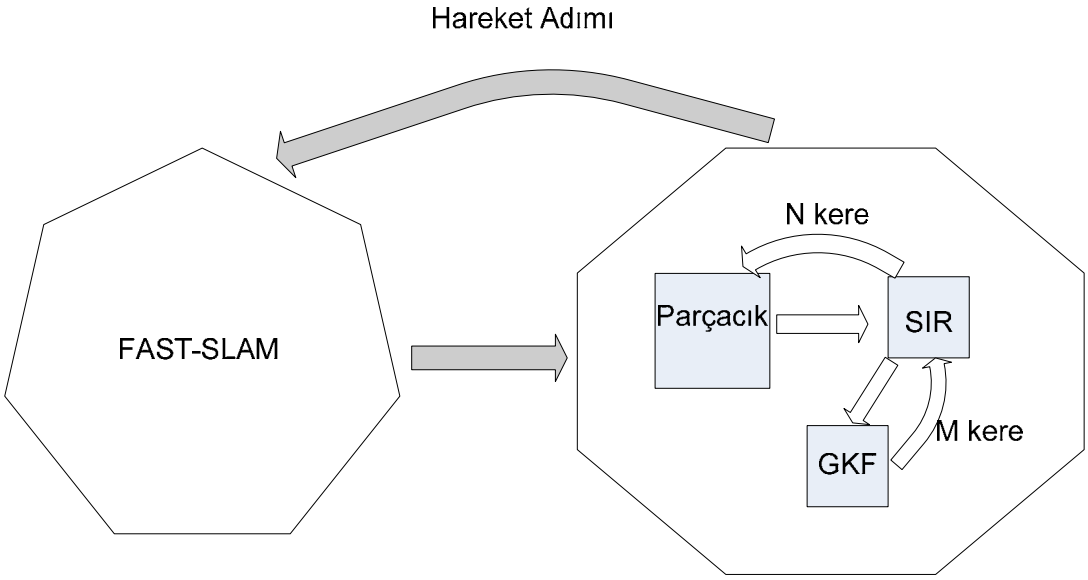
Şekil 4.7 : SLAM Probleminin FastSLAM Hali

RBPF filtresinin adımları akış şeması şekil 4.9'de verilmiştir [5].



Şekil 4.8: RBPF Akış Şeması.

FastSLAM algoritmasının daha rahat anlaşılması için şekil 4.8 oluşturulmuştur [5].



Şekil 4.9 : FastSLAM şematik gösterimi

5. DKF TABANLI SLAM

Bu bölümde Dağılımlı Kalman Filtrei tabanlı SLAM algoritması anlatılacaktır. DKF tabanlı SLAM in çekirdeiğini yukarıda bahsettiğimiz Dağılımlı Dönüşüm oluşturur. Dağılımlı dönüşümde ise Sigma noktaları esas alınır. Bu noktalar sayesinde, rastgele değişkenin bir önceki olasılık dağılımının ortalaması ve kovaryansı tam olarak elde edilir. y=f(X) sistemini göze alalım[12].

X in olasılık dağılımı $\hat{x}$ ortalamasına ve P kovaryansına sahip olsun. İlk olarak sigma noktalarımızı aşağıdaki yöntemlerle hesaplıyoruz. Sigma noktalarının sayısı 2N +1 tanedir ve bunların ilki olasılık dağılımının ortalama değeridir. Diğer noktalar ortalamaya ölçümlendirilmiş kovaryans matrisinin karaköklerinin eklenip çıkarıtlmasıyla oluşturulur. Bu yüzden oluşturulan bu sigma noktaları ortalama etrafında simetriktir.

$$\begin{aligned} \mathcal{X}_0 &= \bar{x} && i = 0 \\ \mathcal{X}_i &= \bar{x} + \left(\sqrt{(n+\lambda)P_{xx}}\right)_i && i = 1, \dots, n \\ \mathcal{X}_i &= \bar{x} - \left(\sqrt{(n+\lambda)P_{xx}}\right)_{i-n} && i = n+1, \dots, 2n \end{aligned} \quad \textbf{(5.1)}$$

Burada lambda , sigma noktalarının ortalama etrafında dağılımını kontrol eden ölçüm faktörüdür.Kovaryans matrisinin karakökünü elde etmek için Cholosesky faktörizasyonu kullanılabilir. Sigma noktaları matrisi şu şekilde oluşturulabilir;

$$\mathcal{X}(k) = [\mathcal{X}_0(k), \mathcal{X}_i(k)\mathcal{X}_{i+N}(k)] \qquad i = 1, \dots, N \quad \textbf{(5.2)}$$

Dönüştürülmüş sigma noktaları nonlinear fonsiyondan geçirilirek bulunur.

$$y_i = h(\mathcal{X}_i) \qquad i = 0, \dots \dots, 2n \quad \textbf{(5.3)}$$

Son olarak bir sonraki dağılımın ortalması ve kovaryansı aşağıdaki gibi elde edilir.

$$\bar{y} = \sum_{i=0}^{2N} W_i^{(m)} Y_i \quad \textbf{(5.4)}$$

$$P_{y=} \sum_{i=0}^{2N} W_i^{(c)} (Y_i - \bar{y})(Y_i - \bar{y})^T \quad \textbf{(5.5)}$$

$$P_{xy=} \sum_{i=0}^{2N} W_i^{(c)} (X_i - \bar{x})(Y_i - \bar{y})^T \quad \textbf{(5.6)}$$

Bu denklemler Taylor seirisinde ikinci dereceye kadar hasastır. $W_i^{(m)}$ ve $W_i^{(c)}$ skaler pozitif ağırlıklardır. m ve c indisleri ağrılıkların, olasılık dağılımının ortalaması veya kovaryansına aitliklerinin belirtir. Burada görüdüğümüz gibi herhangi bir Jacobian veya türev hasabı yoktur[13].

SLAM için yenir artırılmış durum vektörü oluşturalım. Bu vektör orjinal durum vektörü X(k) ve işlem gürültüsü V(k) yı içersin. Yeni kovaryans matrisimiz ise P(k) ve işlem gürültüsü kovaryansında oluşan diyagonal bir matris olsun :

$$X_k^a = [X_k \; V_k]^T \quad \textbf{(5.7)}$$

$$P_k^a = \begin{bmatrix} P_k & \\ & Q_k \end{bmatrix} \quad \textbf{(5.8)}$$

Daha sonra k anında sigma noktaları hesaplanır;

$$\mathcal{X}^{a+} = [\hat{X}_k^a \quad \hat{X}_k^a + \sqrt{(n+\lambda)P_k^a} \quad \hat{X}_k^a - \sqrt{(n+\lambda)P_k^a}] \quad \textbf{(5.9)}$$

Bu sigma noktaları modelden geçirilip dönüşrütülmüş sigma noklatalrı hesaplanır;

$$\mathcal{X}_i^{x-}(k+1) = f(\mathcal{X}_i^{x+}(k), u(k), \mathcal{X}_i^{V+}(k)) \quad \textbf{(5.10)}$$

Burada $\mathcal{X}_i^{x+}(k)$ orjinal durum vektörü için oluşturulmuş sigma noktaları kümesidir. $\mathcal{X}_i^{V+}(k)$ ise işlem gürültüsü için oluşturulan sigma noktalarıdır. Tahmin edilen durum kestirimi ve ilgili kovaryans matrisi aşağıdaki gibi hesaplanabilir;

$$\hat{x}_{k+1}^- = \sum_{i=0}^{2(n+q)} W_i^{(m)} \mathcal{X}_{i,k+1} \quad \textbf{(5.11)}$$

$$P_{k+1}^- = \sum_{i=0}^{2(n+q)} W_i^{(c)} \{\mathcal{X}_{i,k+1} - \hat{x}_{k+1}^-\}\{\mathcal{X}_{i,k+1} - \hat{x}_{k+1}^-\}^T \quad \textbf{(5.12)}$$

Dönüştürülmüş sigma noktaları h(.) ölçüm modelinden geçirilir ve tahmin edilen gözlem örnekleri elde edilir.

$$Z_i^-(k+1) = h(\mathcal{X}_i(k+1)) \quad \textbf{(5.13)}$$

Daha sonra öngörülen ölçüm ve ilgili kovaryans matrisi hesaplanır.

$$\hat{z}_{k+1}^- = \sum_{i=0}^{2N} w_i^m Z_i^-(k+1) \quad \textbf{(5.14)}$$

Doğru veri ilişkilendirmenin yapılması koşuluyla, ölçüm hatası ve kovaryans matrisi P_{ZZ}^{-} ve durum ve ölçüm arasındaki çapraz kovaryans matrisi P_{XZ}^{-} aşağıdaki gibi hesaplanır.

$$v_{k+1} = z_{k+1} - \hat{z}_{k+1}^{-} \quad \textbf{(5.15)}$$

$$P_{ZZ,k+1}^{-} = R + \sum_{i=0}^{2N} W_i^{(c)} \{Z_{i,k+1} - \hat{z}_{k+1}^{-}\}\{Z_{i,k+1} - \hat{z}_{k+1}^{-}\}^T \quad \textbf{(5.16)}$$

$$P_{XZ,k+1}^{-} = \sum_{i=0}^{2N} W_i^{(c)} \{\mathcal{X}_{i,k+1} - \hat{x}_{k+1}^{-}\}\{Z_{i,k+1} - \hat{z}_{k+1}^{-}\}^T \quad \textbf{(5.17)}$$

Gözlem gürültüsü toplamsal olmalıdır. Güncellenmiş durum kestirmi ve ilgili kovaryans matrisi aşağıdaki gibi hesaplanabilr.

$$\hat{X}_{k+1}^{+} = \hat{X}_{k+1}^{-} + K_{k+1}.v_{k+1} \quad \textbf{(5.18)}$$

$$P_{k+1}^{+} = P_{k+1}^{-} - K_{k+1}.P_{ZZ,k+1}^{-}.K_{k+1}^{T} \quad \textbf{(5.19)}$$

Burada kalman kazancı;

$$K_{k+1} = P_{XZ,k+1}^{-}.[P_{ZZ,k+1}^{-}]^{-1} \quad \textbf{(5.20)}$$

Şeklindedir.

Yeni nesne z_{k+1} algılandığında artımsal işlemler aşağıdaki gibi sürdürülebilir;

$$X_{k+1}^{AUG} = [X_{k+1}^{-} \; z_{k+1}]^T \quad \textbf{(5.21)}$$

$$P_{k+1}^{AUG} = \begin{bmatrix} P_{k+1}^{-} & \\ & R_{k+1} \end{bmatrix} \quad \textbf{(5.22)}$$

$$\mathcal{X}_{i,k+1}^{AUG+} = g_i(\mathcal{X}_{i,k+1}^{AUG}) \quad \textbf{(5.23)}$$

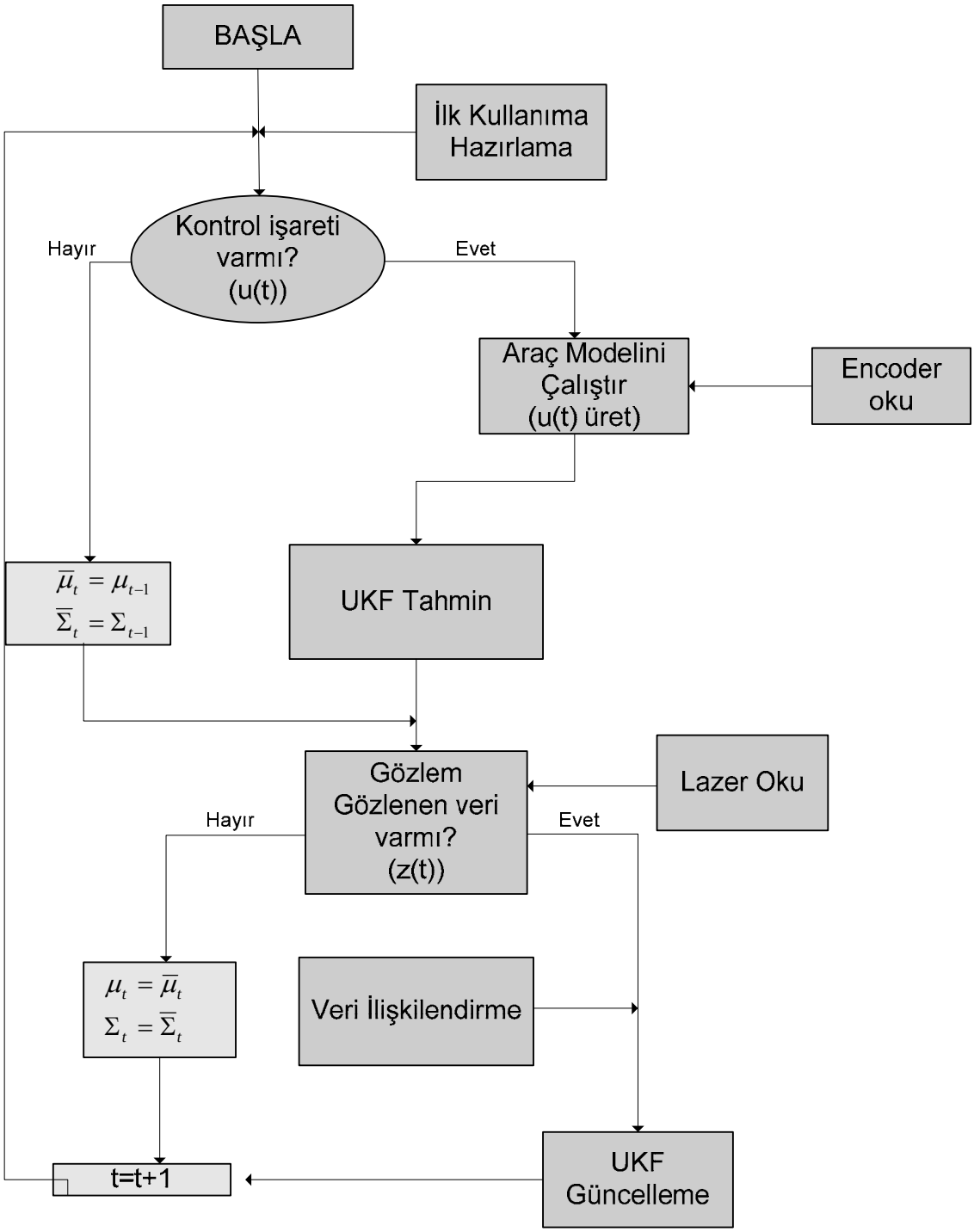
$$\hat{x}_{k+1}^{+} = \sum_{i=0}^{2N} W_i^{(m)} \mathcal{X}_{i,k+1}^{AUG+} \quad \textbf{(5.24)}$$

$$P_{k+1}^{+} = \sum_{i=0}^{2N} W_i^{(c)} \{\mathcal{X}_{i,k+1}^{AUG+} - \hat{x}_{k+1}^{-}\}\{\mathcal{X}_{i,k+1}^{AUG+} - \hat{x}_{k+1}^{-}\}^T \quad \textbf{(5.25)}$$

Burada Rk+1 gözlem gürültüsünün kovaryansıdır.

DKF tabanlı SLAM uygulamasının akış diyagramı Şekil 3.2'de verilmiştir. Algoritma araca kontrol işaretinin uygulanması ile başlar. Kontrol işareti yoksa son kestirim değeri tahmin olarak atanır. Daha sonra lazerden gözlem yapılıp yapılmadığına bakılır. Lazerden veri geliyorsa bu verilere göre araç ve çevrede bulunan işaretçi nesneların pozisyonları güncellenir. Araç için pozisyon bilgisi x,y ve aracın yönünü ifade eden θ değerleridir. Işaretçi nesnelar için pozisyon bilgisi ise x

ve y değerleridir. Kontrol işareti ve aracın bir önceki kestirilen pozisyon değeri, aracın dinamik modeline uygulanarak aracın yeni pozisyonu tahmin edilir. Aracın dinamik model nonlineer bir fonksiyondur. O yüzden kestirim aracı olarak Dağılımlı Kalman Filtresi kullanılmıştır. Aracın dinamik modeli için girişler aracın bir önceki kestirilmiş pozisyonu, uygulanan kontrol işareti ve sistem gürültüsüdür. Bu dinamik modelden aracın ve işaretçi nesneların tahmini yeni pozisyonları **x(k|k-1)** elde edilir. Kovaryans matrisi ise aracın ve gözlemlerin belirsizlik elipsini vermektedir. Bu belirsizlik elipsi ne kadar küçük olursa araç ve işaretçi nesne pozisyonlarıda o kadar gerçeğe yakın elde edilirler. Bu elipsleri küçültmek için elde edilen ölçüm verileri ile kovaryans matrisini de güncellememiz gerekmektedir. Güncellenen kovaryans matrisine **P(k|k)** kestirilmiş kovaryans denir. **P(k|k-1)** kovaryans matrisi tahmininde bir önceki kestirilen kovaryans matris değeri **P(k+1|k)** dır . Bu kovaryans değerleri hesaplanırken yukarıdada bahsettiğimiz gibi sigma noktaları (X) ve ağırlıklar (W) kullanılır. Buradaki gürültünün kovaryans matrisi **Q** önemlidir. Sistemden gelen gürültü encoderin hassasiyeti ve gürültüsüne bağlıdır. Enkoder ne kadar hassas ve ne kadar az gürültüyle ölçüm yapıyorsa tahminlerimiz o kadar iyi olacaktır. Peki bu Q hata kovaryans matrisi nasıl elde edilecek. Araç x pozisyonunda hareket ettilerek encoderden gelen değer ile gerçek ölçüm arasında standart sapma hesaplanır. Bu işlem mümkünse birçok kere tekrarlanarak gaussian bir grafik elde edilir. Bu grafiğin standart sapması σ_x^2'i verir. σ_y^2 , σ_θ^2 değerleride aynı şekilde elde edilerek gürültü kovaryans matrisi Q elde edilir. Kovaryans matrisi tahmininden sonra elde edilen tahmin durum vektörün gözlem modeline uygulanması ile gözlem tahmini **z(k|k-1)** de hesaplanır. Bu hesaplamalarla durum ve gözlem vektörü, kovaryans matrisi tahmin adımı tamamlanmış olur. Bir sonraki adımda lazerden gelen yeni gözlem verileri varsa güncellemeye geçilir. Lazerden herhangi bir veri gelmiyorsa en son tahmin değerleri kestirim değeri olarak kullanılırlar. Lazerden gelen yeni gözlem verileri ile durum vektörü ve kovaryans matrisleri güncellenerek kestirilirler. Kestirme işlemi yukarıda bahsettiğimiz gibi durum ve gözlem arasında kovaryans ve kalman kazancının hesaplanmasıyla yapılır. Güncelleme adımı tamamlandıktan sonra algoritmanın bir çalışma zamanı tamamlanır ve tekrar başa dönülür.



Şekil 5.1: DKF Akış Diagramı

6. UYGULAMA

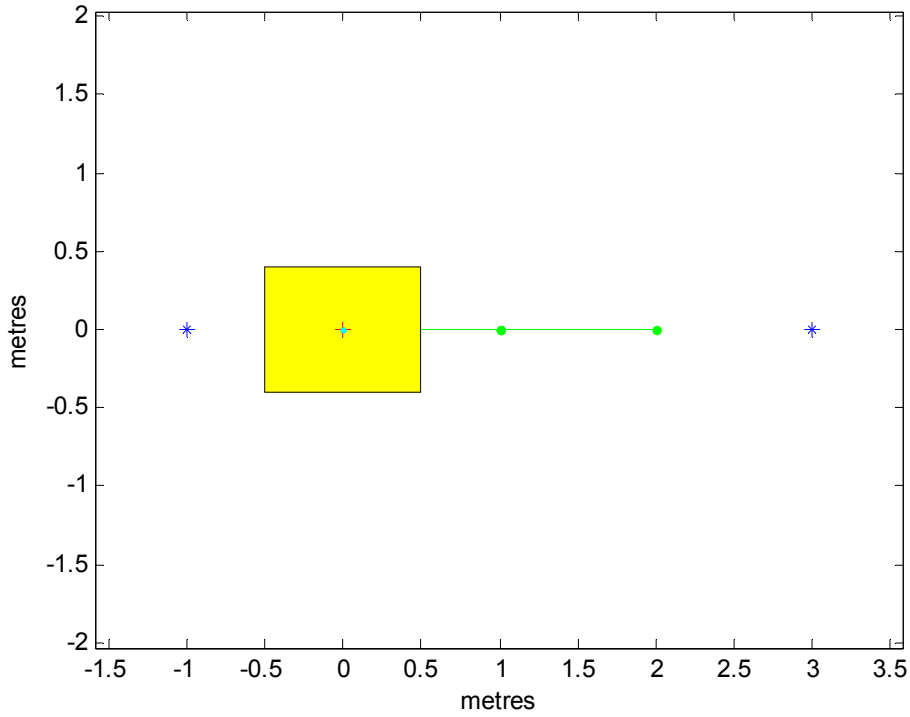
Yaptığımız teorik çalışmaları bilgisayar ve gerçek oratamda test etmek amacıyla bir takım seneryolar ve deney düzenekleri kurduk. Bu çalışmalar aynı zaman labaratuvarımızda yürütülen TUBİTAK projesinde kullanılmıştır.

Labaratuvarımızda daha önce yaptığımız çalışmalarımızda da kullandığımız MATLab programını kullandık. Önceki çalışmalarımızda MATLab programına hazırlannan SLAM agoritmalarını gerçek uygulamada kullanma aşamasında bir takım sorunlarla karşılaşmıştık. Yazdığımız algoritma kodlarını işlemcilere gömerken bir takım sorunlar çıktı. Programda kullanılan bazı fonksiyonların işlemcilere gömülü olarak çalışamadığını farkettik. Bu yüzden algoritmamalarımız MATLab Simulink in Embeded Toolbox eklentisini kullanarak oluşturduk. Bu eklentide kullanılan tüm fonksiyonlar sorunsuz olarak gömülü ortamda çalışmaktadır.

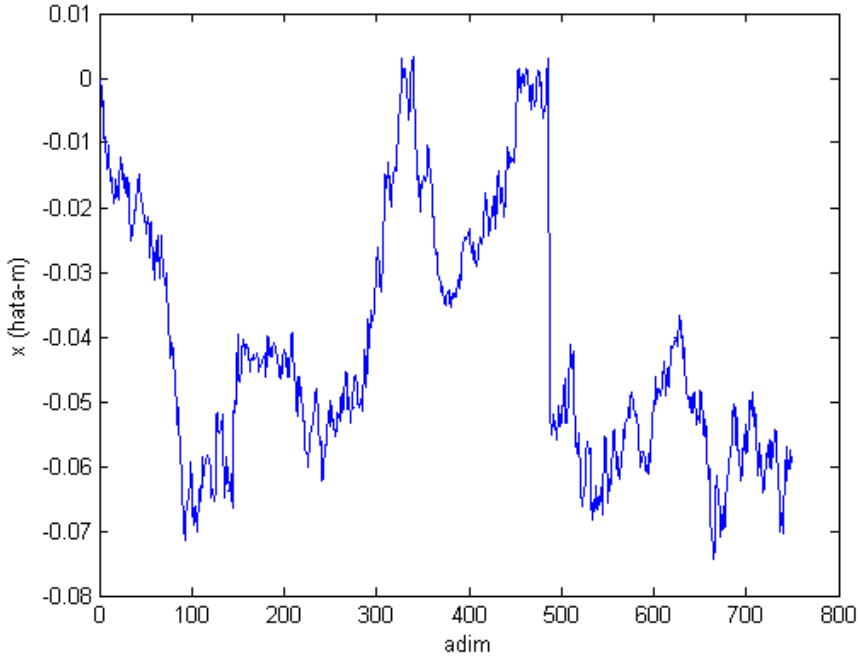
6.1 Yörünge Seneryoları

İlk uygulamamızda aracımız 2 işaretçi nesne arasında doğrusal bir yol planlanmıştır. Bu durumda araç başlangıç noktasından sonra 1 metre yol aldıktan sonra yörüngenin ortasına gelip lazer taraması yapmaktadır. Yörüngenin sonuna gelindiğinde araç 180 derece dönüş yaparak yeniden tarama yapar ve tekrar yörünge ortasına gelir. Benzer dönüş hareketini başlangıç noktasındada yaptıktan sonra bir tur daha atıp çevrim tammalanır. Bu seneryo Şekil 6.1 de MATLab ortamında gösterilmiştir.

Bu uygulamamızda lazer mesafe ölçüm sensörlerimiz her ölçümde sadece bir tane işaretçi nesne görmektedir. Aracın durum vektörü içindeki x,y ve θ değerleri için hata değerleri parametrik olarak verilmiştir.

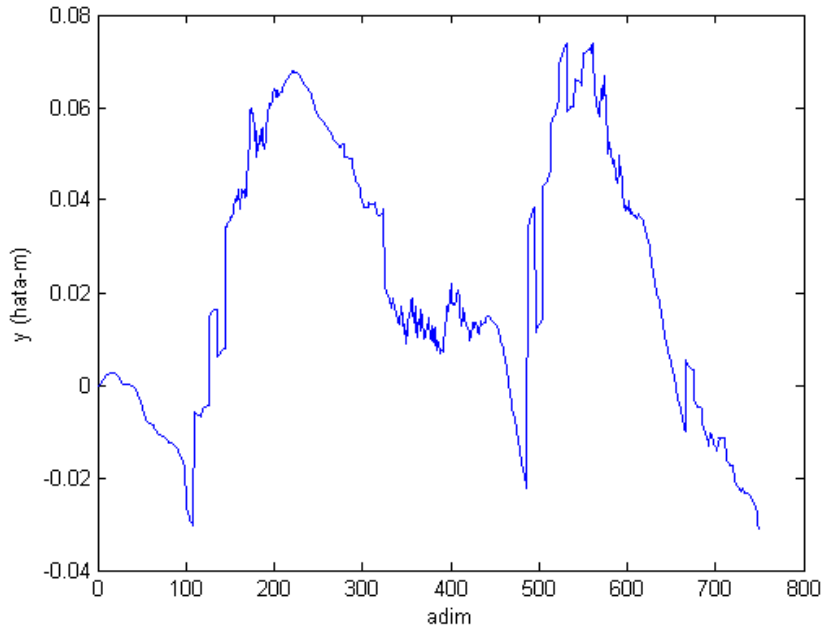


Şekil 6.1: Uygulama 1

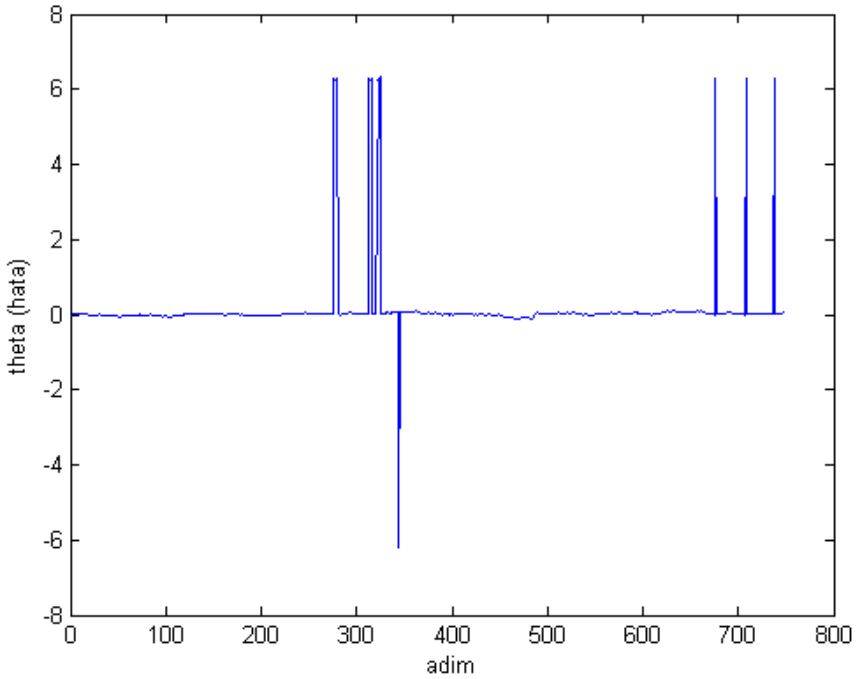


(a)

Şekil 6.2: Uygulama 1 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata (c)



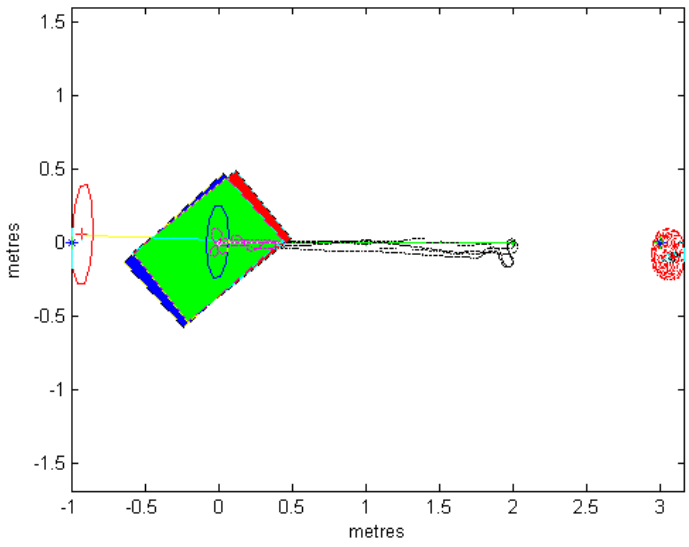
(b)



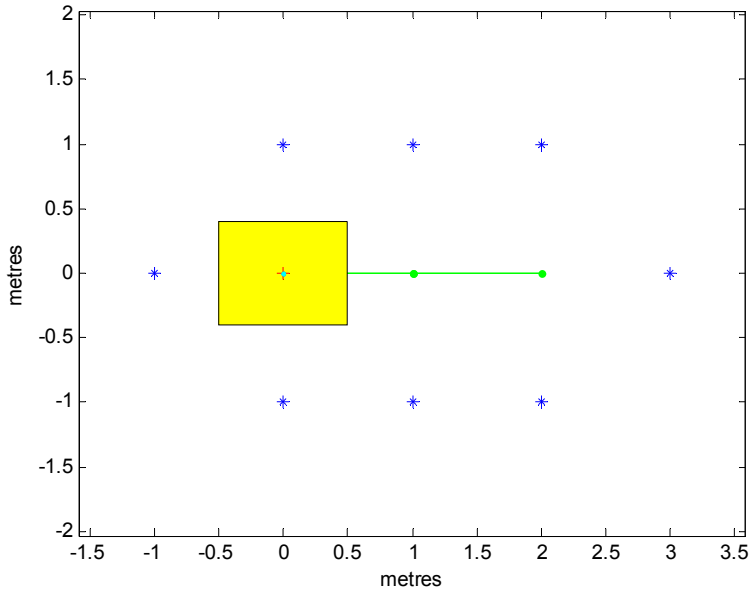
(c)

Şekil 6.2: (devam) Uygulama 1 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata (c)

İkinci uygulamamızda araç 8 işaretçi nesne arasında yine doğrusal bir yörünge izlemiştir. Aracın hareketi uygulama birdeki ile aynıdır. Ancak bu uygulamada daha fazla işaretçi nesne kullanılmıştır. Uygulama seneryasu şekil 6.4 de gözükmektedir. DKF-SLAM sonucunda elde edilen araç lokasyon hataları aşağıda gösterilmiştir

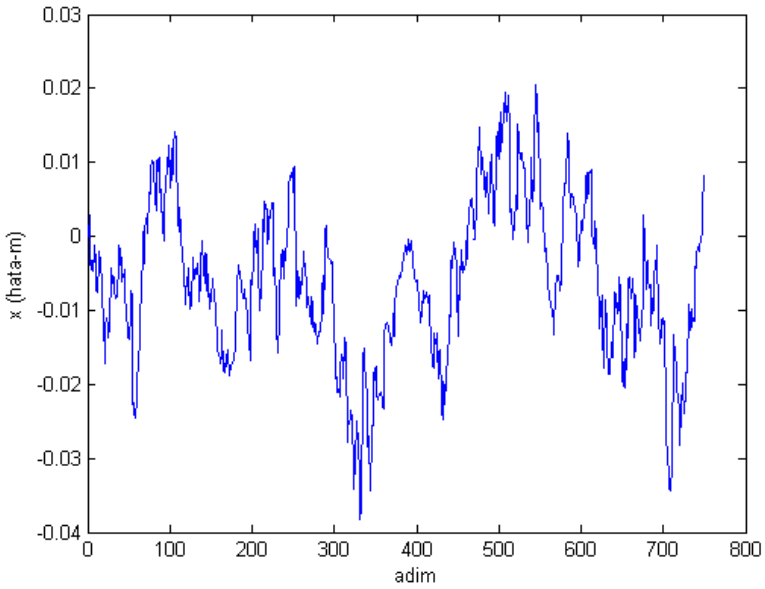


Şekil 6.3: İşaretçi nesneler için belirsizlik elipsleri

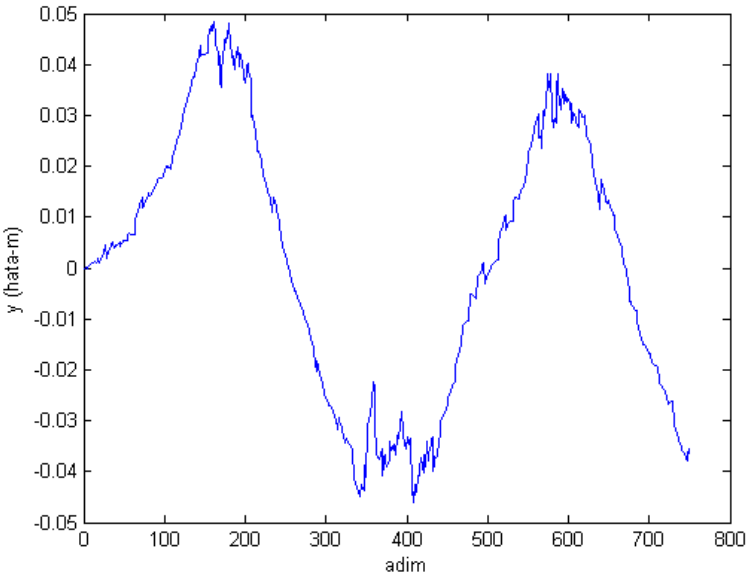


Şekil 6.4: Uygulama 2

İşaretçi nesneler için oluşturulan hata belirsizlikelipsleri şekil 6.6 de gözükmektedir. Bu uygulamada kullanılan her işaretçi nesnenin kestriim değeri ile bulunan lokasyonu bir belirsizlik elipsoidi ile tanımlanır ve bu elipsoid her turda daha da küçülereke belirsizlik limitini daraltır.

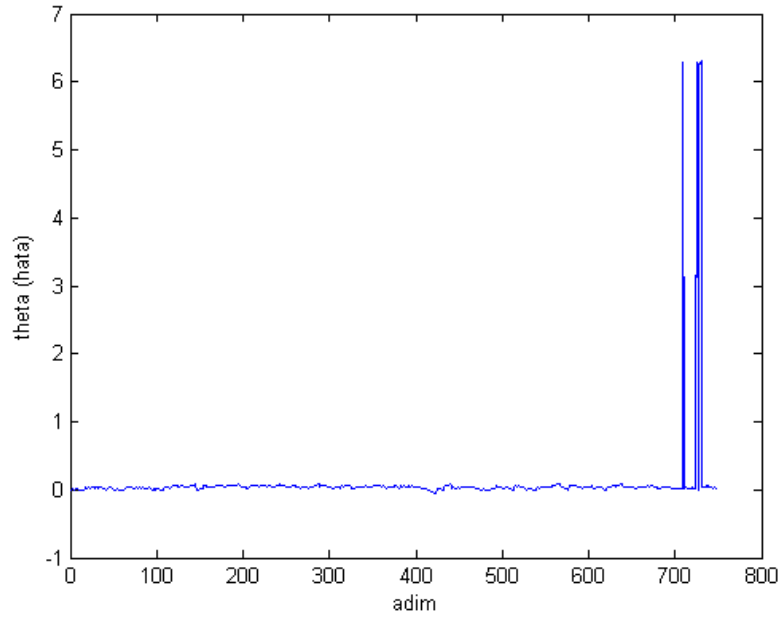


(a)



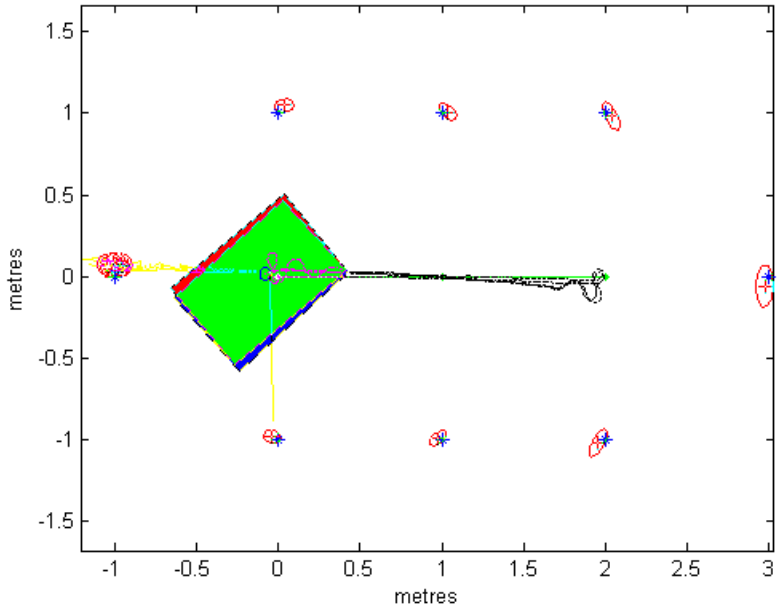
(b)

Şekil 6.5: Uygulama 2 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata (c)



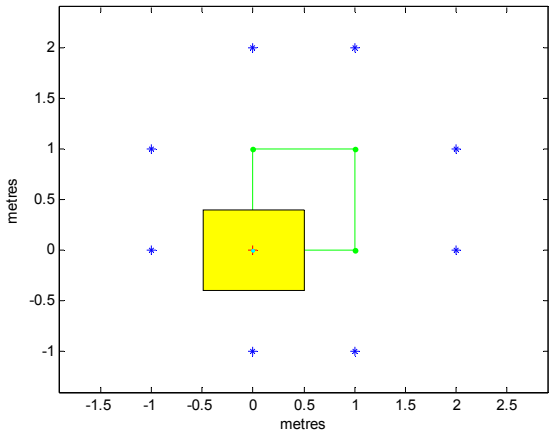
(c)

Şekil 6.5: (devam) Uygulama 2 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata (c)



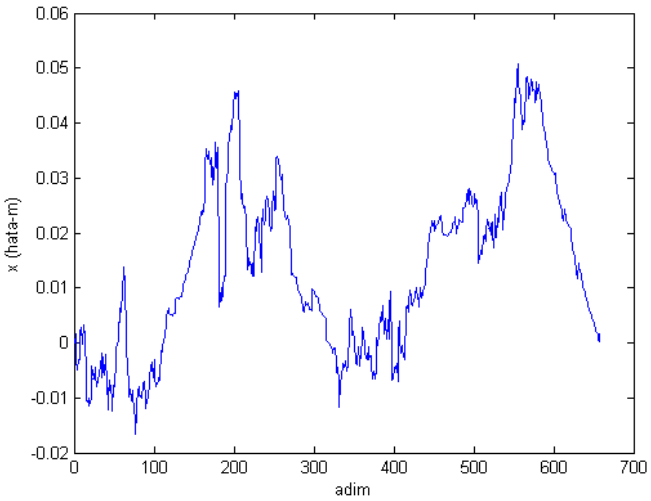
Şekil 6.6: Uygulama 2 işaretçi nesneler için belirsizlik elipsleri

Üçüncü uygulamamızda aracımız bir kenarı 1 metre olan kare şeklinde kapalı bir yörünge izlemektedir. Kerenin köşlerinde 1 metre uzaklıkta birbirine dik olacak şekilde 6 adet işaretçi nesne yerleştirilmiştir. Bu uygulamanın seneryosunu şekil 6. 7 da görülmektedir.



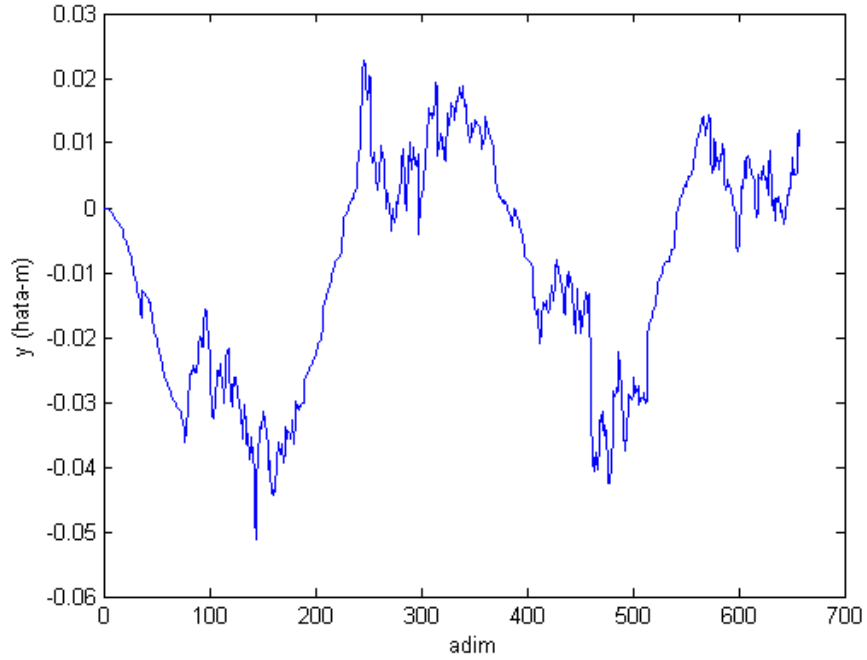
Şekil 6.7: Uygulama 3

Yapılan deney sonuçlarında araç lokasyonunda oluşan hata değerleri aşağıda gösterilmiştir. İşaretçi nesneler için oluşturulan ve her bir işaretçi nesnenin hata belirsizliği aşaüıdaa gösterilmiştir. Bu uygulamada kullanılan her işaretçi nesnenin kestriim değeri ile bulunan lokasyonu bir belirsizlik elipsoidi ile tanımlanır.

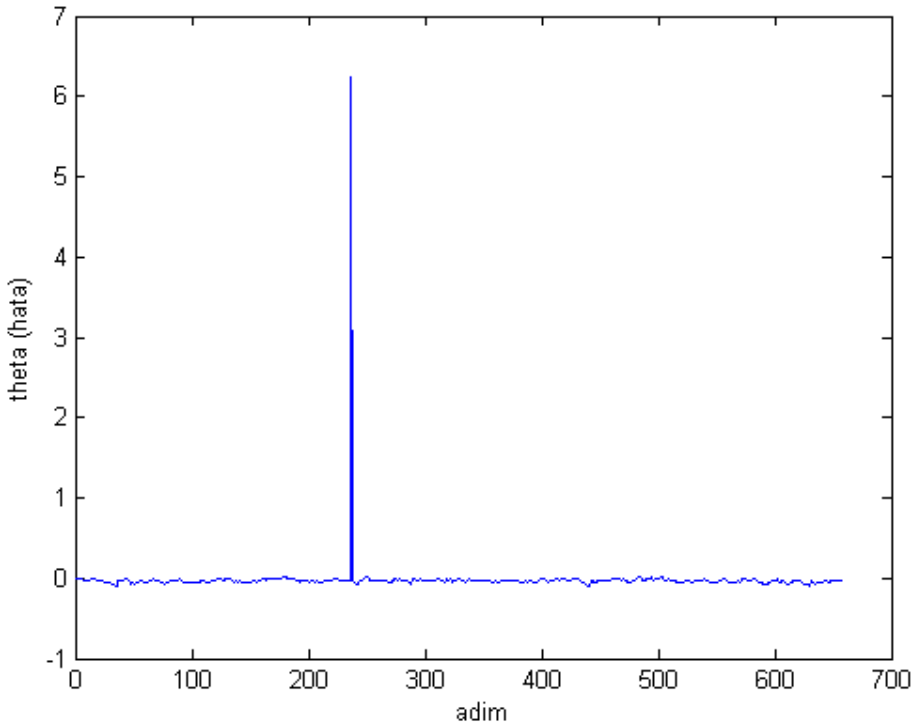


(a)

Şekil 6.8: Uygulama 1 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata (c)

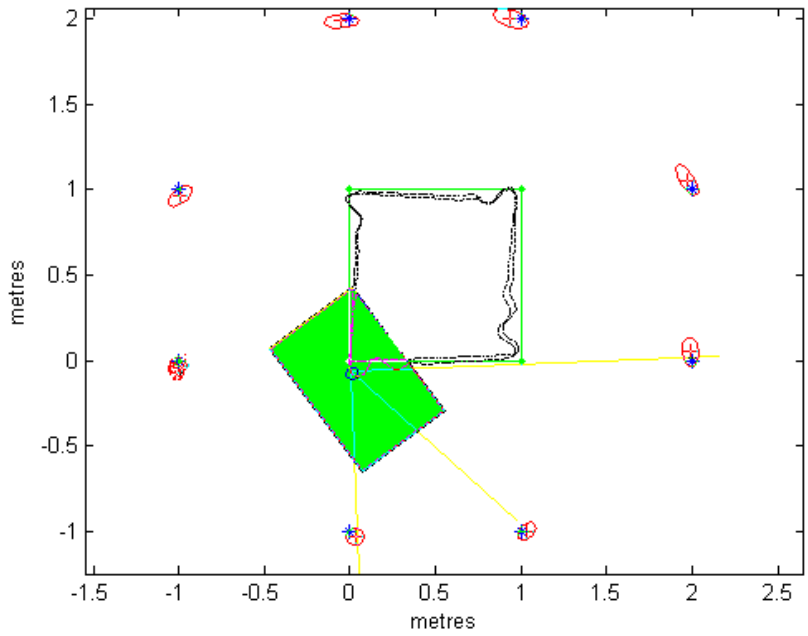


(b)



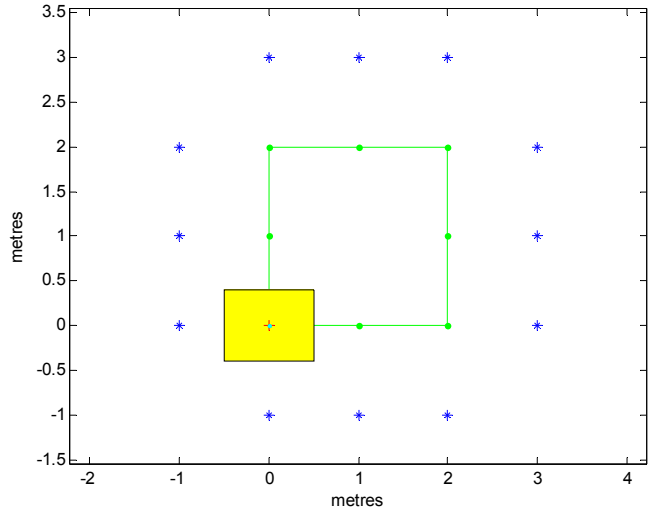
(c)

Şekil 6.8: (devam) Uygulama 1 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata (c)



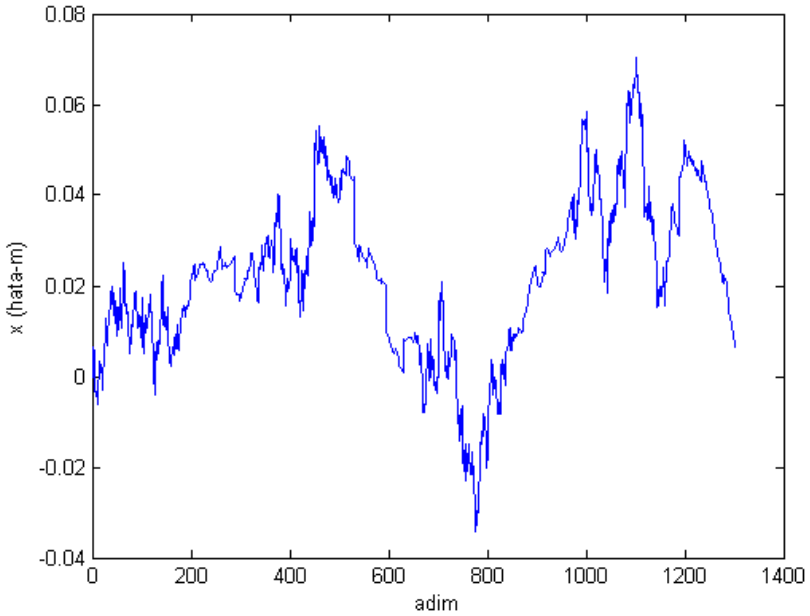
Şekil 6.9: Uygulama 3 işaretçi nesneler için belirsizlik elipsleri

Dördüncü uygulamamızda üçüncü uygulamamıza benze şekilde kare bir yöünge izlenmiştir. Farklı olarak karenin bir kenarı 2 metre alınmıştır. İşaretçi nesnelerimiz yürüngeden birer metre uzaklığa yerleştirilmiştir. Bu yugulammaızda toplam 12 adet işaretçi nesne kullanılmıştır. Seneryomuz şekil 6.10 te gözükmektedir.

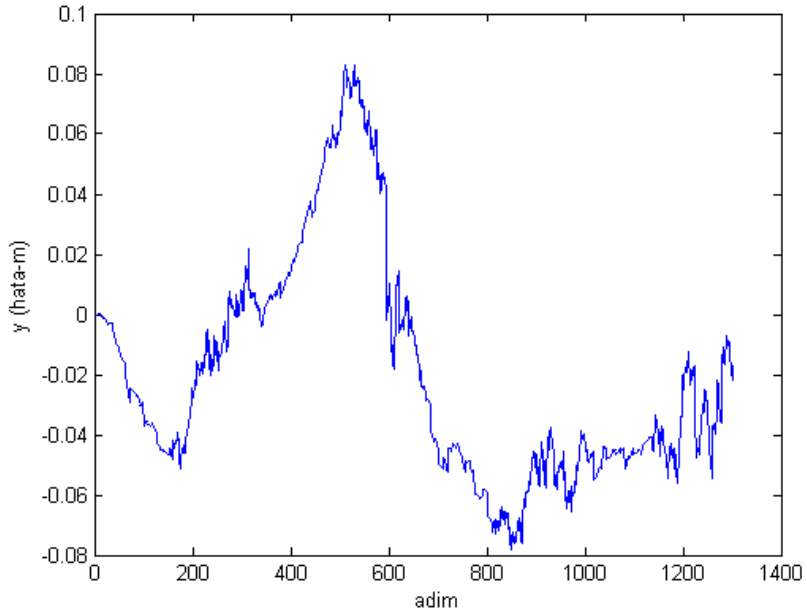


Şekil 6.10: Uygulama 4

Yapılan çalışmalar sonucunda araç lokasyonundaki oluşan hata değerleri aşağıda verilmiştir. İşaretçi nesneler için oluşturulan belirsizlik elipsoidlerini şekil de görülmektedir.

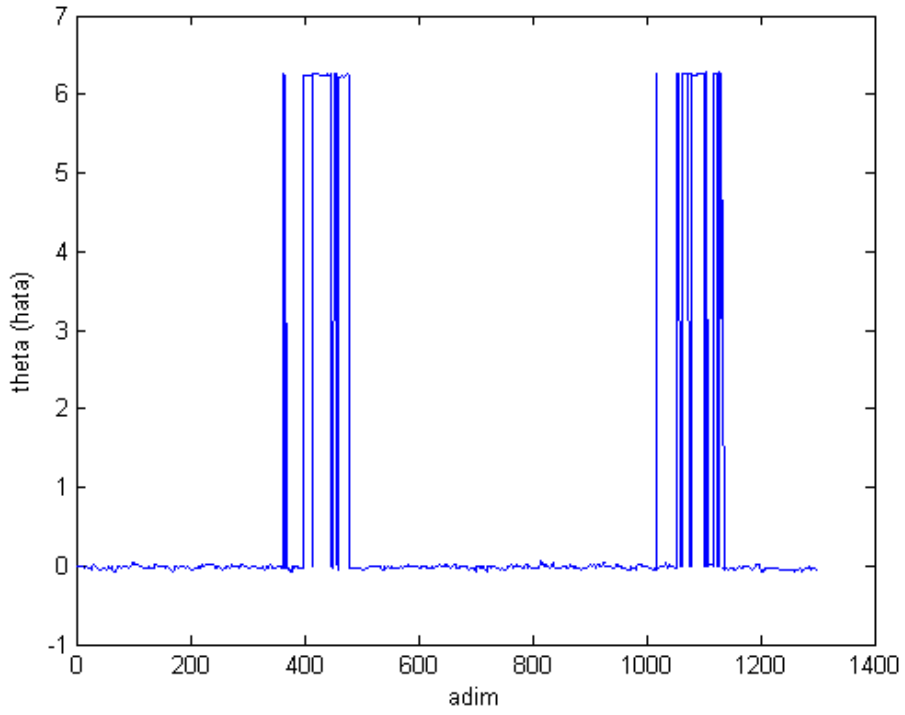


(a)



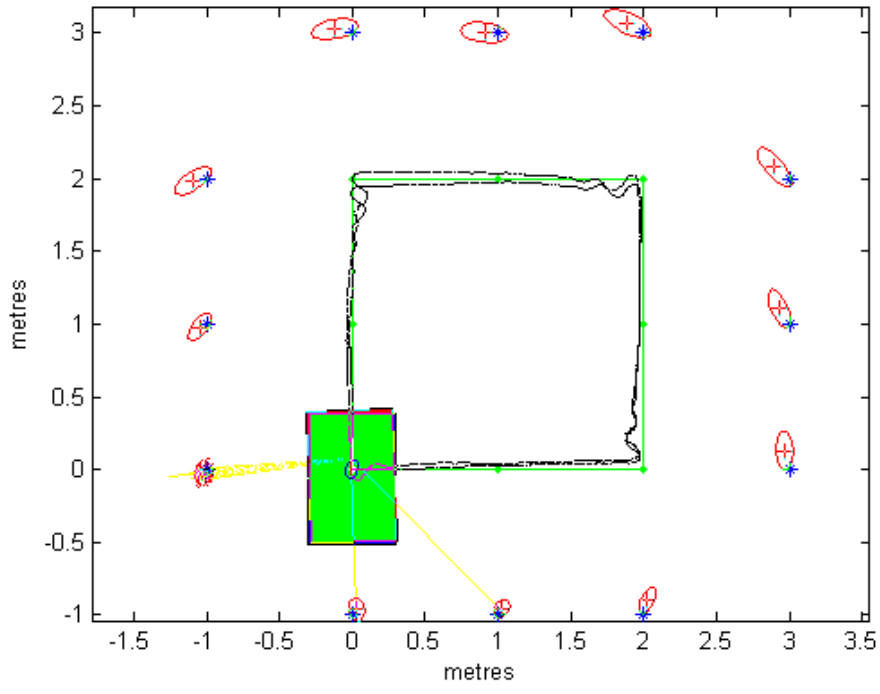
(b)

Şekil 6.11: Uygulama 4 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata (c)



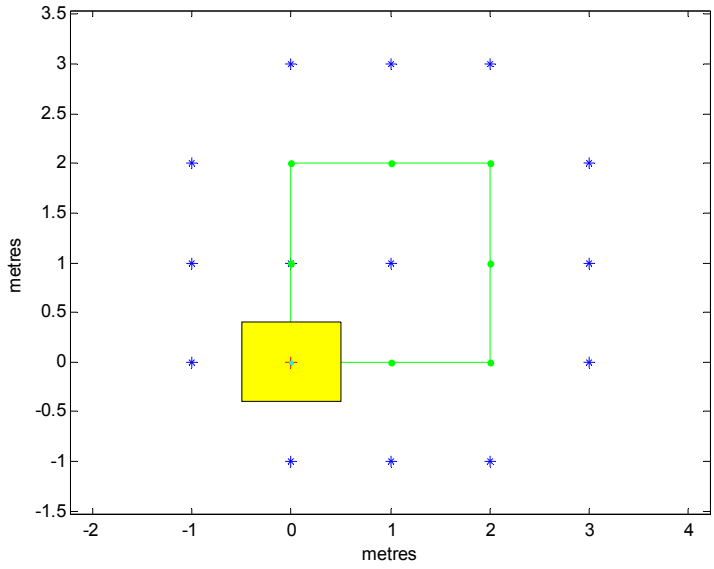
(c)

Şekil 6.11: (devam) Uygulama 4 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata (c)

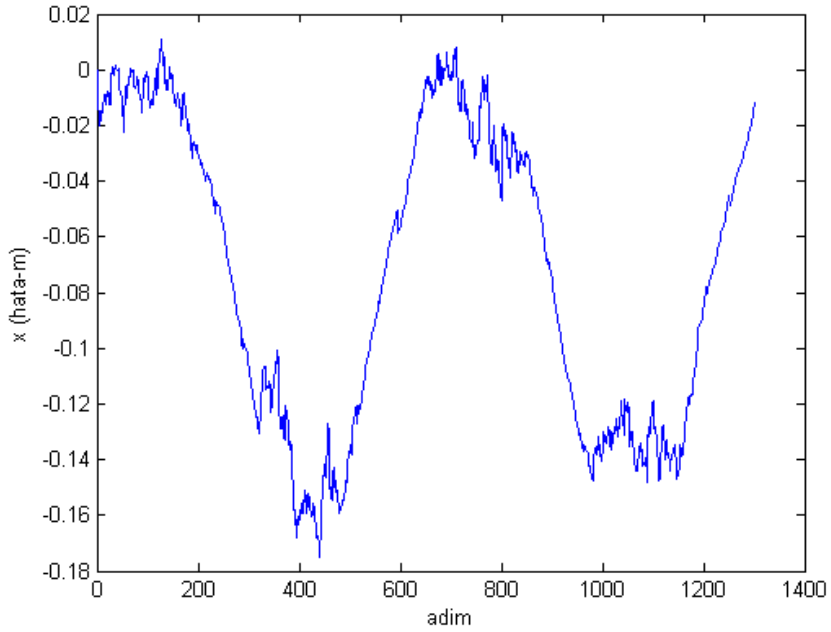


Şekil 6.12: Uygulama 4 işaretçi nesneler için belirsizlik elipsleri

Beşinci uygulamamzıda bi önceki uygulamaya ek olarak yörüngenin içine bir işaretçi nesne konulmuştur.Uygulamamızın senryosunun şekil 6.13 da görebiliriz. Yapılan çalışmalar sonucunda araç lokasyonunda oluşan hatalar şekil 6.14 da gözükmektedir.

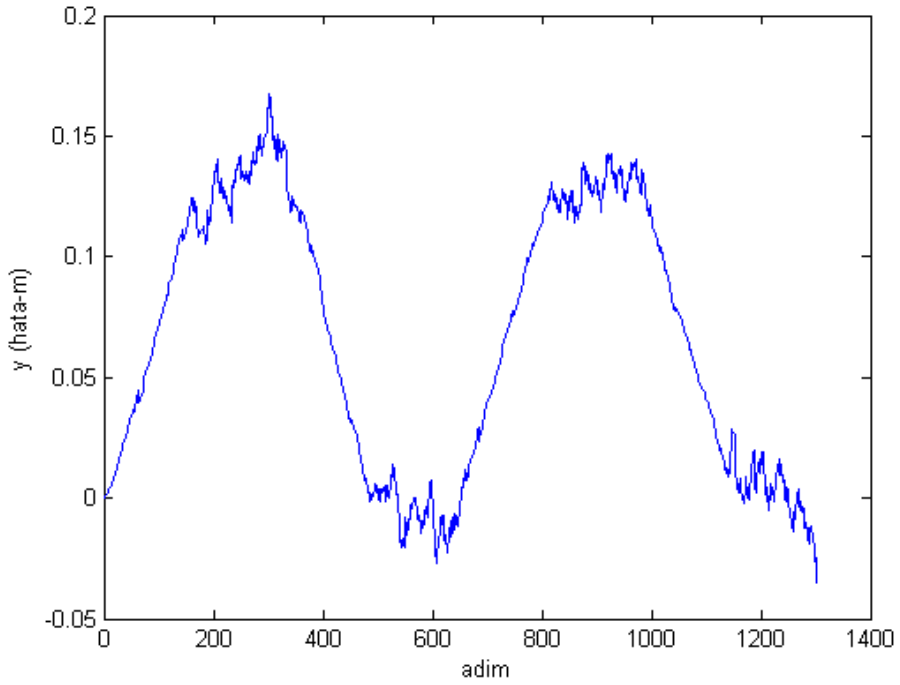


Şekil 6.13: Uygulama 5

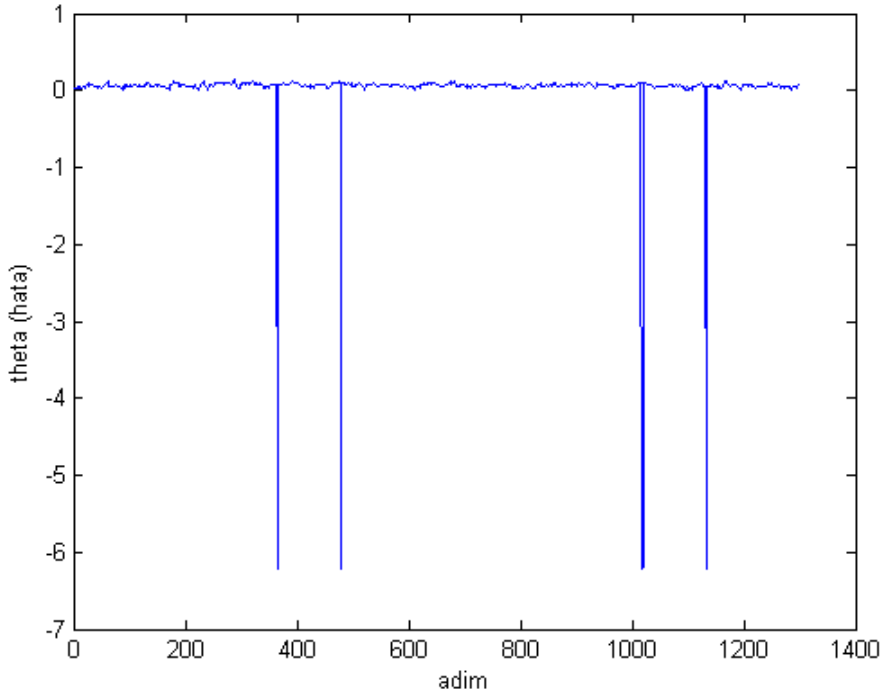


(a)

Şekil 6.14: Uygulama 5 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata (c)



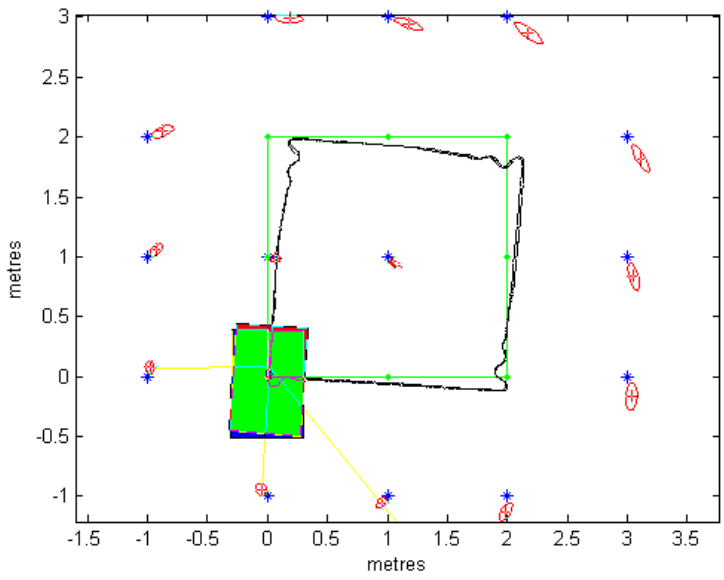
(b)



(c)

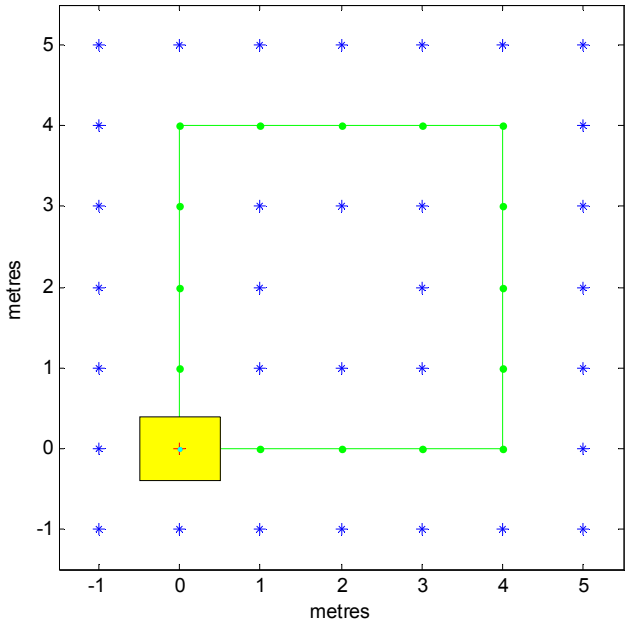
Şekil 6.14: (devam) Uygulama 5 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata (c)

İşaretçi nesneler için oluşturulan belirsizlik elipsoidleri şekil 6.15 de verilmiştir.



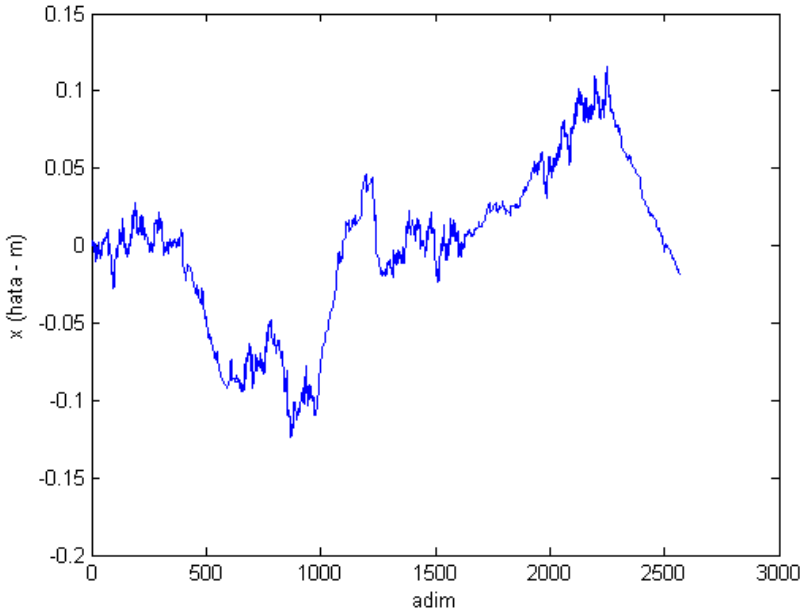
Şekil 6.15: Uygulama 5 işaretçi nesneler için belirsizlik elipsleri

Son uygulamamaızda yörüngemizin içinde ve dışında bulunan işaretçi nesneler birlikte algılanmaya çalışılmıştır. Burada kare yörüngemiziz bir kenarının uzunluğu 4 metre olarak ayarlanmıştır. Ve yörüngemizin içinde ve dışında simetrik olarak yerleştrilimiş 32 adet işaretçi nesne kullanılmıştır(şekil 6.16)

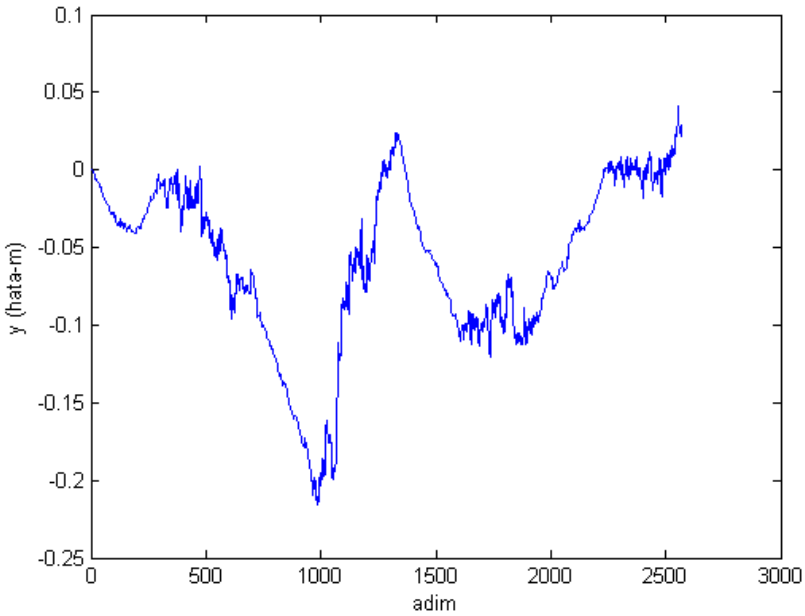


Şekil 6.16: Uygulama 6

Yapılan çalışmalar sonucunda aracın konumunda oluşan hata değerlerini şekil 19 daa görmekteyiz. İşaretçi nesneler için oluşturulan belirsizlik elipsleri ise şekil 6. 20 de görülmektedir.

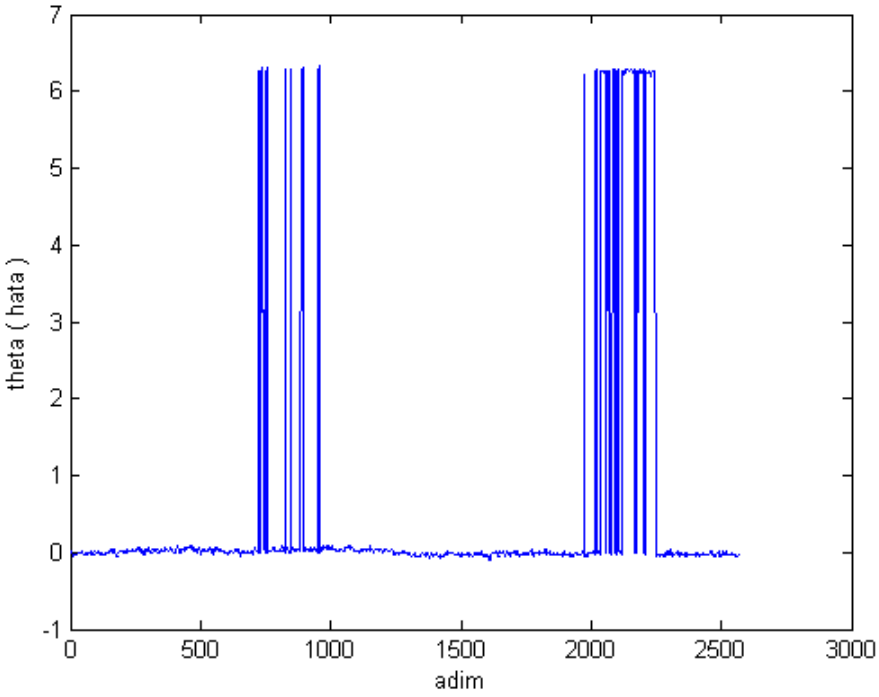


(a)



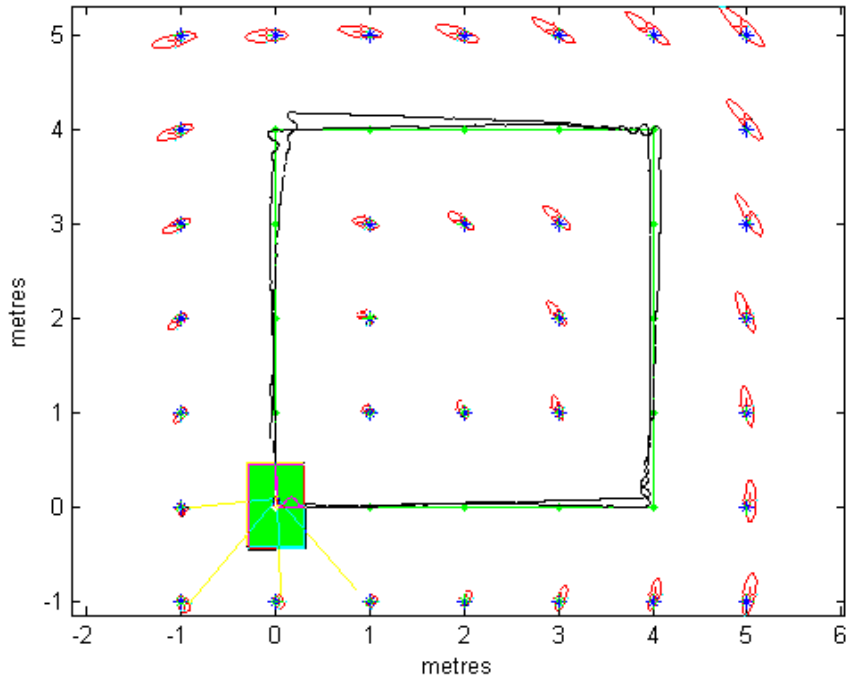
(b)

Şekil 6.17: Uygulama 6 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata (c)



(c)

Şekil 6.17: (devam) Uygulama 6 x konumundaki hata (a), y konumundaki hata (b) ve theta konumundaki hata (c)



Şekil 6.18: Uygulama 5 işaretçi nesneler için belirsizlik elipsleri

7. SONUÇ

1980 lerden bu yana, kestirim algoritmalarına ihtiyac duyulan mühendislik problemlerinde, genişletilmiş kalman filtresi standar bir metod olmuştur. Fakat, son zamanlarda, yeni ve daha hassas, lineer olmayan filtreler, genişletilmiş kalman filtresine alternatif olarak önerilmiştir. Bir çok yeni algoritmada olduğu gibi, yeni filtreleme algoritmaları çok yaygın olarak bilinmiyordu ve kullanım alanları sınırlıydı. Bu çalışmada sigma noktalı kalman filreleri ailesinden olan dağılımlı kalman filtresinin gelşimi incellenmiştir.Ve testlere tabi tutulmuştur.

İlk olarak, analitiksel bir nonlineer yaklaşım algoritması olan dağılımlı kalman filteri incelenmiştir. Dağılımlı kalman filtresi örnekleme tabanlı bir filtredir ve nonlineer bir dönüşüm metodu olan dağılımılı dönüşüme dayanır.Bu metodda olasılık dağılımının ortalama değeri ve kovaryansı, örneklenmiş sigma noktaları kümeleri kullanarak paremetrelerle etkili bir şekilde ifade edilir.

Yapılan çalışmalar sonucunda dağıllımlı kalman filtresinin eş zamanlı haritalama ve lokalizasyon probleminde kullanıma uygun bir metod olduğu gözlenmiştir. Elde ettiğimiz verilerin hatalarına göz attığımızda SLAM problemi için uygun olduğunu görülmüştür. Daha sonra yapılacak çalışmalarda MATlab simulink ortamında hazırlanan sigma noktalarının hesaplanması kısmında daha fazla işaretçi nesne olması durumunda meydan gelecek karmaşıklığın üstesinden gelmek için bir algoritma geliştirilebilir. Ayrıca bu çalışmada kullanılan dağılımlı kalman filtresini parçacık filtreleindeki genişletilmiş kalman filtresi yerine kullanarak daha etkili ve doğru kestirim algoritmaları yapılabilir.

KAYNAKLAR

[1] Bailey, T. ve Durrant-Whyte, H.F., 2006. Simultaneous Localization and Mapping (SLAM): Part I, *IEEE Robotics and Automation Magazine,* **June 2006.**

[2] Uhlmann, J.K., 1994. Simultaneous Map Building and Localization for Real Time Applications, transfer thesis.

[3] Kurt, Z., 2007. Eş Zamanlı Konum Belirleme ve Haritalamaya Yönelik Akıllı Algoritmaların Geliştirilmesi, *Yüksek Lisans Tezi,* Y.T.Ü. Fen Bilimleri Enstitüsü, İstanbul.

[4] Kavak, D., 2007. İnsansız Kara Araçları Navigasyonunda Genişletilmiş Kalman (GKF) Ve Sıkıştırılmış Kalman Filtre (CGKF) Tabanlı Slam Yöntemlerinin Geliştirilmesi Ve Karşılaştırılması. İTÜ Kontrol Kumanda Sistemleri Yüksek Lisans Tezi.

[5] Kuleli, A., 2009 . Mobil Robotlarda Parçacık Fıltresi Kullanarak Eş Zamanlı Lokalizasyon Ve Haritalama. İTÜ Kontrol Mühendisliği Yüksek Lisans Tezi.

[6] Julier, S.J., Uhlmann, K.J., Unscented Filtering and Nonlinear Estimation. *IEEE*. 2004.

[7] Lee, D-J., 2005**.** Nonlinear Bayesian Filtering with Applications to Estimation and Navigation, Texas AM University. PhD Thesis.

[8] Brunke, S.S. 2001. Nonlinear Filtering and System Identification Algorithms for Autonomous Systems. University of WashingtonPhD Thesis.

[9] Julier, S.J., Uhlmann, K.J., Reduced Sigma Point Filters for the Propagation of Means and Covariances Through Nonlinear Transformations. *IEEE ACC* 2002.

[10] Bailey, T. ve Durrant-Whyte, H.F., 2006. Simultaneous Localization and Mapping (SLAM): Part II, *IEEE Robotics and Automation Magazine,* **September 2006,** 108-117.

[11] Riberio, I.M., 2004. Kalman and Extended Kalman Filters: Concept, Derivation and Properties.

[12] Shojaie, K., Shahri, M.A. Iterated Unscented SLAM Algorithm for Navigation of an Autonomous Mobile Robot. *IEEE/RSJ,* September 2008.

[13] Wan, E.A., Van Der Merwe, R. 2000.The Unscented Kalman Filter for Nonlinear Estimation, *IEEE Adaptive Systems for Signal Processing, Communications and Control Symposium,* **October 2000.**

[14] Van Der Merwe, R 2004. SigmaPoint Kalman Filters for Probabilistic Inference in Dynamic State Space Models. OGI School of Science and Engineering. PhD Thesis.

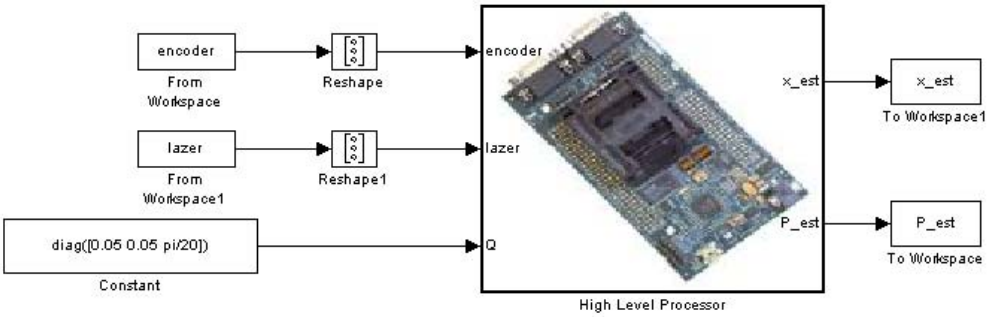
[15] Julier, S.J., Uhlmann, K.J., The Scaled Unscented Transformation. *IEEE American Control Conference.* **May 2002**.

[16] Martinez-Cantin, R., Castellanos, A.J. 2005. Unscented SLAM for Large-Scale Outdoor Environments. *IEEE/RSJ International Conference.*

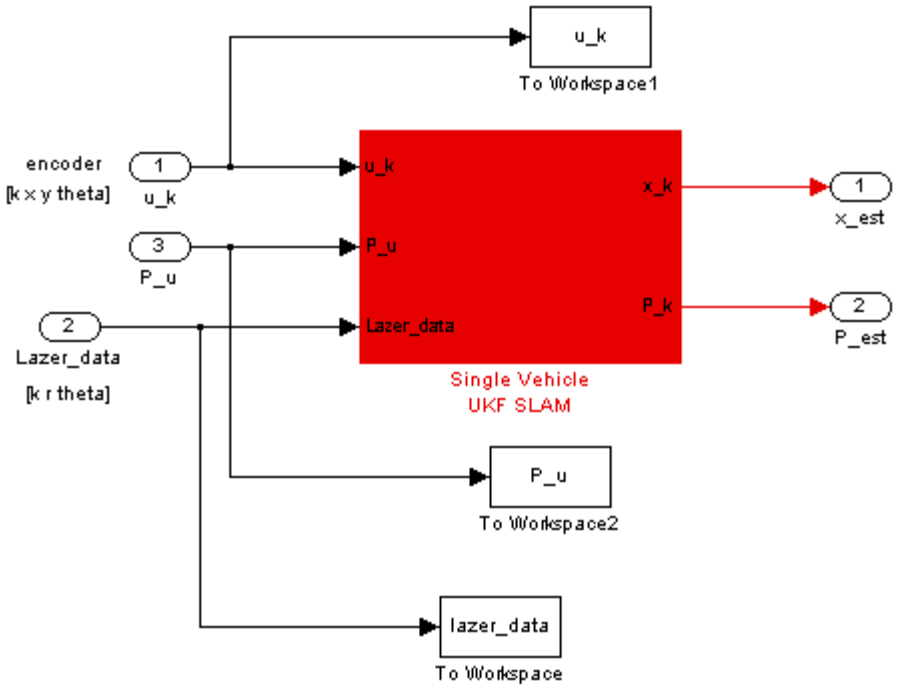
EKLER

EK A.1: DKF SLAM simulink programı

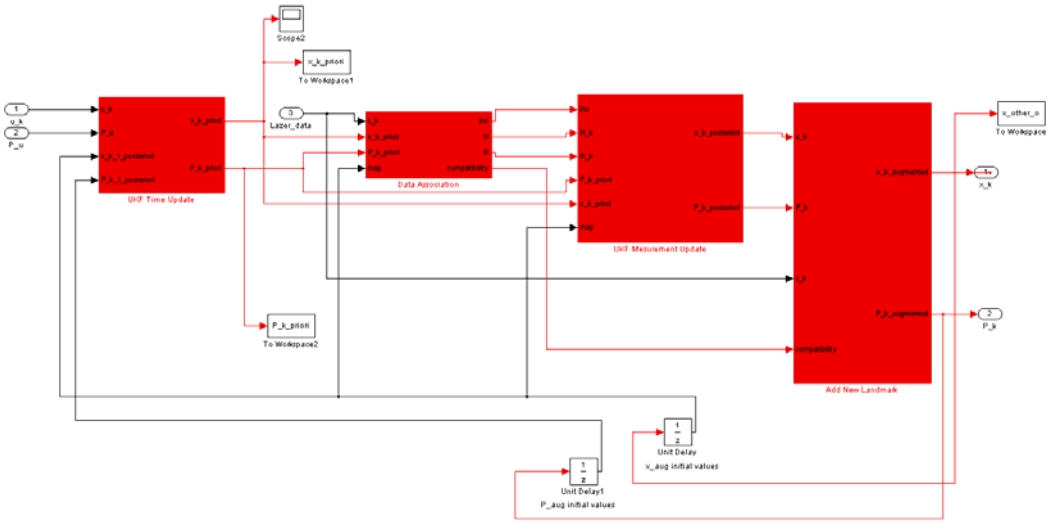
EK A.1



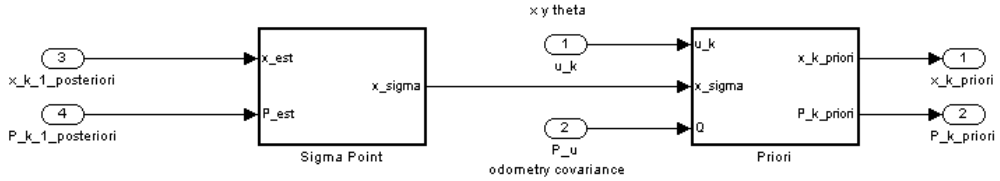
Şekil A.1: DKF SLAM Simulink Şeması Ana Görünüm



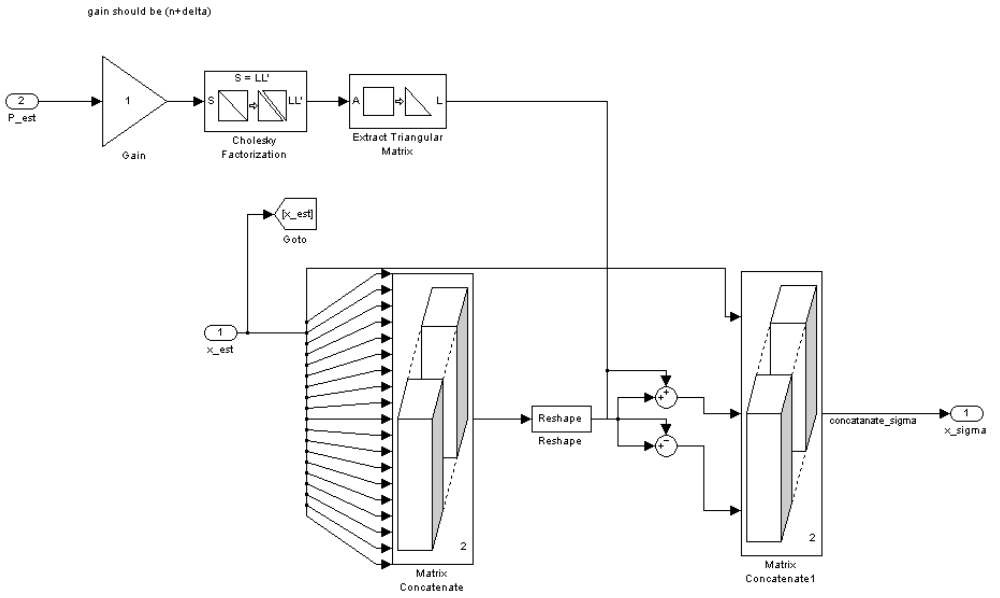
Şekil A.2: Tek Araç için DKF-SLAM Simulink Bloğu Ana Görünüm



Şekil A.3: DKF-SLAM Simulink Bloğu

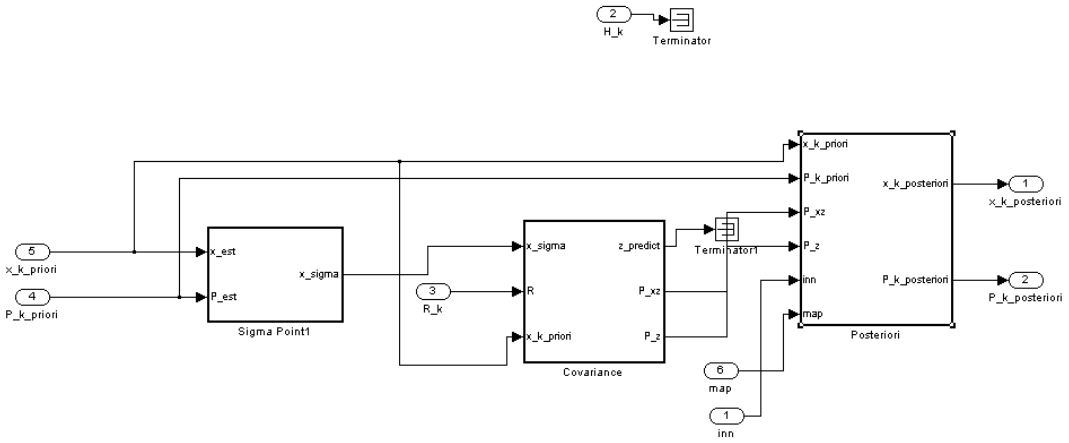


(a)

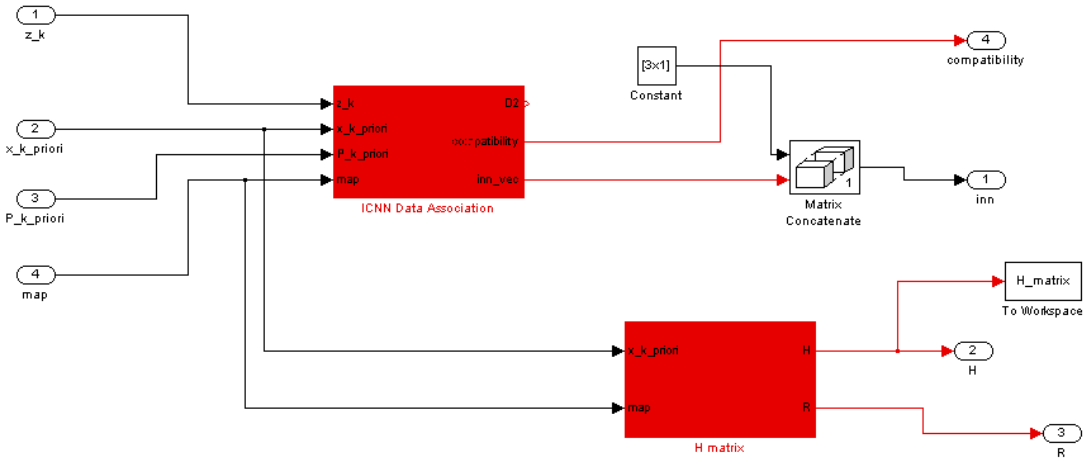


(b)

Şekil A.4: DKF SLAM de zaman güncellemesi adımında (a) sigma noktaları hesaplama (b)



Şekil A.5: DKF SLAM Ölçüm Güncellemesi Adımı



Şekil A.6: Veri İlişkilendirme Bloğu

ÖZGEÇMİŞ

Ad Soyad: Kadir Paslıoğlu

Doğum Yeri ve Tarihi: Pazar – 16-08-1984

Adres: Yayla Mh. Şinasi Dural Cd. Adabük Sk. Adabük Sit. B-Blok D:10 Tuzla/İstanbul

Lisans Üniversite: İstanbul Teknik Üniversitesi –Fizik Mühendisliği

Yayın Listesi:

- Kurşun A, Uluocak V., Bostan O, Paslıoğlu K., Temeltaş H. Pnömatik Eyleyici Tırmanan Mobil Robot Tasarım ve Gerçeklemesi. TOK. 2009