T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VERİ BİRLEŞTİRMEYE DAYALI PARÇACIK FİLTRELEME İLE GERÇEK ZAMANLI HAREKET İZLEME

DOKTORA TEZİ

Tuğrul TAŞCI

Enstitü Anabilim Dalı : BİLGİSAYAR VE BİLİŞİM MÜHENDİSLİĞİ

Tez Danışmanı : Doç. Dr. Cemil ÖZ

Kasım 2014

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VERİ BİRLEŞTİRMEYE DAYALI PARÇACIK FİLTRELEME İLE GERÇEK ZAMANLI HAREKET İZLEME

DOKTORA TEZİ

Tuğrul TAŞCI

Enstitü Anabilim Dalı : BİLGİSAYAR VE BİLİŞİM MÜHENDİSLİĞİ

Bu tez 14 / 11 /2014 tarihinde aşağıdaki jüri tarafından oybirliği ile kabul edilmiştir.

Prof. Dr.
A. Coşkun SÖNMEZ
Jüri Başkanı

Prof. Dr.
Orhan TORKUL
Üye

Prof. Dr.
Nedim TUTKUN
Üye

Doç. Dr.
Cemil ÖZ
Üye

Doç. Dr.
İbrahim ÖZÇELİK
Üye

TEŞEKKÜR

Tüm öğrenim hayatım boyunca bana destek veren aileme, doktora çalışmam süresince ilgi, sevgi ve desteğini benden hiçbir zaman esirgemeyen eşime, çalışmamın ilerlemesine her aşamada katkıda bulunan başta danışmanım Doç. Dr. Cemil ÖZ olmak üzere tez izleme jürimdeki hocalarıma teşekkür ederim. Bilişim Fakültesi ve Uzaktan Eğitim Merkezi'nde görev yaptığım dönemlerde ihtiyaç duyduğumda katkı sağlamaktan çekinmeyen çalışma arkadaşlarım ve hocalarıma da teşekkür ederim. Ayrıca, ulusal ve uluslararası boyutlarda yönetimi yaptığı projelerin çeşitli aşamalalarındaki iş geliştirme süreçlerinde verdiği sorumluluklar, akademik konferans, seminer ve diğer çeşitli toplantı organizasyonlarında bulunmam ve değerli katılımcılar huzurunda görüşlerimi ifade etmem için sağladığı imkanlar ile başta özgüven olmak üzere çalışma hayatımda prensip, vizyon ve birikim kazanmamda büyük emeği olan, akademik çalışma hayatımın başından itibaren kendisinden ziyadesiyle istifade ettiğim, değerli hocam Prof. Dr. Orhan TORKUL'a teşekkürü bir borç bilirim.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR LİSTESİ

EKF	: Genişletilmiş Kalman Filtresi
GSF	: Gauss Toplamı Filtresi
IPT	: Matlab İmge İşleme Araç Kutusu
KF	: Kalman Filtresi
M-BT	: Çoklu Top İzleme
MC	: Monte Carlo
MCMC	: Markov Zinciri Monte Carlo
PF	: Parçacık Filtresi
SF-PF	: Veri Birleştirmeye Dayalı Parçacık Filtresi
SMC	: Sıralı Monte Carlo
UKF	: İlerletilmiş Kalman Filtresi
$\& \sim$	: Mantıksal $VE\ DE\check{G}İL$ İşlemi
$\sim =$	: Mantıksal $EŞİT\ DE\check{G}İL$ İşlemi
$\propto$	: Orantılılık operatörü
$\triangleq$	: Kabul operatörü
$\sim$	: Olasılık dağılımından örnekleme
$\otimes$	: Karşılıklı elemanların birebir çarpımı
η	: Normalleştirme katsayısı
ξ	: Kümülatif ortalama karesel hatadaki mutlak değişim
X	: Dinamik sistemin durum değişkeni
Z	: Dinamik sistemin ölçüm değişkeni
X_k	: k zaman adımındaki durum değişkeni
Z_k	: k zaman adımındaki ölçüm değişkeni
$p(X_0)$	: Durum değişkenin 0. zaman adımındaki olasılık dağılımı
$p(X_{0:k})$	: Durum değişkenin birleşik olasılık dağılımı
$p(Z_1)$	: Ölçüm değişkenin 1. zaman adımındaki olasılık dağılımı

$p(Z_{1:k})$: Ölçüm değişkenin birleşik olasılık dağılımı

$p(X_{0:k}|Z_{1:k})$: Durumların ölçümlere bağlı şartlı birleşik olasılık dağılımı

$p(x_k|x_{k-1})$: Geçiş önseli (transition / temporal prior)

$\delta(.)$: Dirac-Delta fonksiyonu

$\widetilde{w}$: Normalize edilmiş parçacık ağırlık vektörü

$L^{(k)}$: k zaman adımındaki uzaklık matrisi,

$Lh^{(k)}$: k zaman adımındaki ortak olabilirlik matrisi

$N_{eff}^{(k)}$: k zaman adımındaki etkin parçacık sayısı

N_{th} : Dejenerasyon katsayısı

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar kelimeler: Olasılıksal kestirim, Parçacık filtresi, Hareket izleme

Günümüz dünyasında ulaşım, iletişim ve eğlence gibi alanlarda insanlara yönelik hizmet veren birçok sistem, aynı şekilde ülkelerin dünya çapında caydırıcı güç olma yolunda sahip oldukları bilimsel araştırma platformları, otomasyon sistemleri ve askeri teknolojiler ile tüm bu sistemleri kuşatan bilişim teknolojileri mevcuttur. Bu teknoloji ve sistemlerin geliştirilme süreçlerinde ortaya çıkan teknik problemlerin, çoğunlukla gerçek-zamanlı ya da buna yakın bir hızda ve makul hata payı ile çözülmesi gereklidir. Aslında tüm bu sistemler yaşayan sistemler olarak kabul edilir. Geliştiricilerin kapsamlı çalışmalar sonucunda ortaya koydukları prensip ve algoritmalar doğrultusunda ürettikleri yazılım programları aracılığıyla bu tür sistemler, işleyiş ortamına bağlı olarak dış dünya ile sürekli etkileşim halinde bulunurlar. Bu tür sistemler, dışardan çok çeşitli sensörler yoluyla toplanan verilerin anlamlı bilgilere dönüştürülerek uygun şekilde kullanıma sokulmasını sağlayan mekanizmalar içerirler.

Değişen ihtiyaç ve koşullar, yüksek yaşam standartları için beklentiler ve insanın sorgulayıcı tabiatı her geçen gün yeni ve daha yetenekli sistem ve hizmetlerin ortaya çıkmasını gerekli kılmaktadır. Bu nedenle, daha iyi donanım ve sensörlerin yanında zeki yazılımlarla donatılmış modern sistemlerin varlığı günümüzde her zamankinden çok daha fazla önem kazanmaktadır. Bu tür sistemlerin işleyiş döngüsünde, sistem geliştiricileri açısından en önemli husus herhangi bir anda sistemin hangi durumda olduğunu açıklayacak ve dolayısıyla gerekli aksiyonun yapılmasına yardımcı olacak bilgiyi sunan bir alt sistemin varlığıdır. Ancak, birçok durumda böyle bir bilgiye herhangi bir kısıtlama olmaksızın ulaşmak mümkün değildir. Bu bağlamda, sistemin durumlarına ilişkin doğrudan ölçülemeyen büyüklük değerleri gürültülü sensörler yardımıyla yapılan ölçümlerden yola çıkılarak elde edilmeye çalışılır. Böyle uygulamalar ise alt düzeyde Kestirim ve Filtreleme alanları kapsamına giren metotlarla yapılmaktadır.

Bu çalışmanın temel konusu olan gerçek-zamanlı hareket izleme de bilimsel araştırma, askeri teknoloji ve eğlence gibi birçok alana yönelik olarak geliştirilen sistemler için çözülmesi gerekli bir alt problemdir. Bu problemin çözümü için son yıllarda araştırmacılar genelde olasılığa dayalı filtreleme özelde ise Parçacık filtreleme yöntemleri üzerine yoğunlaşmıştır. Bu çalışmada da farklı kaynaklardan gelen verilerin birleştirilmesine dayanan bir parçacık filtreleme uygulaması ile RGB-D sensörüne sahip bir aygıtın iç-tanımlı algoritmalarla yürüttüğü gerçek-zamanlı hareket izleme sürecinin belli varsayımlar çerçevesinde iyileştirilmesi hedeflenmiştir.

REAL-TIME MOTION TRACKING USING SENSOR-FUSION BASED PARTICLE FILTER

SUMMARY

Keywords: Probabilistic estimation, Particle filter, Motion tracking

There exist so many systems in today's world, developed for serving people in the fields such as transportation, communication or entertainment and also in scientific research platforms, automation systems and military technologies acquired by governments in the way of being a globally deterrent power. Technical problems emerging within the development process of such systems are required to be solved mostly in real-time or near real-time with acceptable accuracy rates. Indeed, these systems are deemed as living systems such that they permanently interact with the outside world depending on their operating environment through the software programs produced in line with the principles and algorithms revealed by the architectures as a result of comprehensive studies. Such systems include kind of mechanisms ensuring data collected by various sensors from the outside to be conveniently put into practice.

Changing needs and conditions, expectations for higher living standards and exploration-driven nature of human being imply novel and more capable systems and services to be introduced with each passing day. Hence, modern systems equipped with better hardware and sensors at the same time supported by smarter software gain currency today more than ever. The most significant requirement from the point of architectures for such type of systems is to have a kind of sub-system allowing acquiring information at any time associated with the states of the system in order to take the appropriate action. However, in most situations it is quite difficult to reach this information without any restrictions. Within this context, the requested information about the system states which is not directly obtainable is attained via the available sensor measurements, of course, in a distorted form owing to the noise. Such applications undoubtedly refers to the methods of well-positioned research field namely "Estimation Theory" and "Filtering".

Real-time motion tracking which is the main focus of this dissertation is obviously a sub-problem required to be solved for several systems developed within the scope of fields such as scientific research, military technologies and entertainment. Solution of this problem has recently lead to researchers directing to the field of probabilistic filtering in general and particle filtering in particular. In this study, it is aimed to enhance the real-time motion tracking process of a device having a RGB-D sensor by its built-in algorithms pursuant to specific assumptions using a sensor-fusion based particle filtering application.

BÖLÜM 1. GİRİŞ

Günümüzde, ulaşım, iletişim ve eğlence gibi alanlarda günlük hayatın gereksinimlerinin karşılamasında; süregiden bilimsel araştırma girişimleriyle evrenin, insanın veya diğer canlıların varoluşlarıyla ilgili bilinmeyenlerin keşfedilmesinde ve uluslararası ekonomik, siyasal ya da askeri rekabette ülkelerin öne çıkmasında önemli etkisi olan birçok teknolojik sistem ve hizmet mevcuttur. Manyetik Rezonans (MR) görüntüleme yoluyla hastalık teşhisi, karmaşık yapıya sahip uçaklarla seyahat, oyun konsollarındaki harekete duyarlı oyunlarla eğlence deneyimi, insansız kara, hava ve deniz araçlarıyla tehlikeli bölgelerde, okyanusların derinliklerinde ya da diğer gezegenlerde araştırma gibi çok sayıda uygulama bunlara örnek olarak verilebilir [1]. Yaşamın her alanında faaliyette olan bu tür sistemlerin geliştirilmesi ve uygulamaya alınması, arka planda hedef/hareket izleme, konumlandırma ve navigasyon, işitsel/görsel sinyal işleme, tanıma ve anlamlandırma ve doğa olaylarının kestirimi gibi çok sayıda teknik problemin (mekanik, optik, akustik, haptik, çevresel gibi) çeşitli tipteki sensörlerle elde edilen verilerden kullanılabilir bilgi çıkarılmasına dayanan çözümlerini gerekli kılar [2]. Söz konusu teknik problemler için getirilen çözümlerin, gerçek-zamanlı ve kabul edilebilir bir doğrulukta olması da çoğu durumda bir zorunluluktur. Örneğin, Avrupa Uzay Ajansının koordinatörlüğünde gerçekleştirilen Rosetta projesinde [3], on yıllık uzay yolculuğunun sonunda Philae adlı insansız uzay aracı, kısa bir zaman önce hareket eden kuyruklu yıldıza iniş yapmıştır. Böyle bir uygulamada, insansız uzay aracının hedefindeki kuyruklu yıldızın uzaydaki hareketini gerçek-zamanlı olarak takip edip yörüngesini buna göre belirlemesi kaçınılmazdır.

Değişen ihtiyaç ve koşullar, yüksek yaşam standartları için beklentiler ve tabii ki insanın sorgulayıcı tabiatı her geçen gün yeni ve daha yetenekli sistem ve hizmetlerin ortaya çıkmasını gerekli kılmaktadır. Bu nedenle, daha iyi donanım ve sensörlerin yanında zeki yazılımlarla donatılmış modern sistemlerin varlığı günümüzde her zamankinden çok daha fazla önem kazanmaktadır. Bu tür sistemlerin işleyiş

döngüsünde, sistem tasarımcıları/geliştiricileri açısından en önemli husus herhangi bir anda sistemin hangi durumda olduğunu açıklayacak ve dolayısıyla gerekli aksiyonun yapılmasına yardımcı olacak bilgiyi sunan bir mekanizmanın varlığıdır [4]. Ancak, çoğu durumda bu tür sistemlerin çoğunlukla lineer olmayan ve düzgün bir gürültü içermeyen yapılarından dolayı böyle bir bilgiye herhangi bir kısıtlama olmaksızın ulaşmak mümkün değildir [5]. Bu bağlamda, sistemin durumlarına ilişkin bilgi ancak gürültülü sensörler yardımıyla yapılan ölçümlerden yola çıkılarak elde edilir. Bu gürültü, hem dinamik sistem işleyiş süreci içinde hem de ölçümlerde mevcuttur [6]. Böyle bir durumda, sistemin durumlarına ilişkin bilgi veren parametrelerin değerlerini kestirebilmek için bir temel oluşturmak amacıyla sistem dinamik işleyiş sürecinin simülasyonundan ve sensörlerle yapılan ölçümlerden elde edilen verilerden gürültünün çıkarılması gereksinimi ortaya çıkar.

İnsansız kara, deniz ve hava araçları, sürücüsüz otomobil, etkileşimli bilgisayar oyunları, askeri savunma ve saldırı araçları gibi çok sayıda sistem için ortak bir problem olan [1] gerçek-zamanlı hareket izleme, düzlemde ya da uzayda zamana bağlı olarak konum değiştiren bir ya da daha fazla cismin hareket yörüngesinin makul bir gecikme süresi içinde belirlenmesi ve gerekli aksiyonun devreye alınması süreci olarak tanımlanabilir [7]. Gerçek-zamanlı hareket izleme, ele alınan özel problemin niteliğine bağlı olarak, başta çevresel, akustik ve optik sensörler olmak üzere farklı tipteki sensörler aracılığıyla elde edilen verilerden anlamlı bilginin çıkarılması esasına dayanır. Örneğin, güdümlü bir füze sisteminde hedefin konumunu belirlemede çevresel bir ısı sensörü kullanılabilirken [8], okyanus dibinde araştırma yapan robotik bir araçta yön tayini için ultrasonik bir sensör tercih edilebilir [9]. Bu çalışmada ise görsel imgeleri dijital işaretlere dönüştüren optik sensörler (kamera) aracılığıyla insan hareketinin gerçek-zamanlı olarak izlenmesi üzerinde yoğunlaşılmıştır.

Optik sensörlerin hareket izlemede kullanımı, literatürde bilgisayar görmesine dayalı hareket izleme [10] adıyla bilinmektedir ve geçtiğimiz yirmi yıllık dönemde çok sayıda çalışmaya konu olmuş önemli bir araştırma alanıdır [11],[13]. Bilgisayar görmesine dayalı hareket izleme alanında yapılan çalışmalar, veri elde etme yolları bakımından aktif ve pasif sistemler olmak üzere iki ana kategoride toplanabilir [14].

Aktif sistemler, izlenen insan deneğin / deneklerin özel donanımlı kapalı ortamlarda bulunmasını ve özel renkli kıyafetler ile işaretçiler giymesini gerektirir [15],[16],[17]. Bu tür sistemler, pahalı ve karmaşık olmalarına rağmen, insan deneğin hareketini kısıtlamaları nedeniyle genellikle özel film çekimi ve yüksek hassasiyet gerektiren hareket analizi vb. sistemlerde kullanılır [18],[19],[20]. Pasif sistemler, aktif sistemlere göre daha düşük doğruluk derecesine sahip olmakla birlikte, işaretçi kullanımı gerektirmezler ve çoğunlukla yazılım tabanlı işleyen yapıları dolayısıyla mekân kısıtlaması olmadan serbestçe kullanılabilme olanağı sunarlar [10],[21],[22]. Aktif ve pasif sistemlerin her ikisi de belirli hareket izleme uygulamaları için daha uygun olsa da, günümüz dünyasında pasif sistemlerin kullanımına gereksinim duyan uygulamaların sayısı ve çeşitliliği aktif sistemlere göre çok daha fazladır.

Gözetleme, video sıkıştırma, animasyon ve kontrol, sanal gerçeklik (karakter animasyonu, bilgisayar oyunları), gelişmiş insan-makine ara-yüzü (işaret dili, mimik tanıma), içerik tabanlı görüntü depolama ve geri getirme ve analiz ve sınıflandırma (klinik çalışmalar, spor eğitimi) vb. uygulamalarda, insan hareketinin gerçek-zamanlı izlenmesi, çözülmesi gerekli bir problemdir [24]. Bu uygulamaların birçoğu mekân kısıtlaması altında yapıldığında amacına hizmet etmeyeceği için işaretçi tabanlı profesyonel ve ticari izleme sistemleri yerine bilgisayar görmesi tabanlı pasif izleme sistemleri ile çözümü daha fazla ilgi görmektedir.

Pasif hareketi izleme sistemleri, model tabanlı ve görünüş tabanlı olmak üzere iki kategori altında toplanmıştır [14]. Model tabanlı sistemlerde insan vücudunun anatomik ve eklemlerin kinematik özelliklerinin dikkate alındığı iki ya da üç boyutlu modellerin kullanımı söz konusudur. Görüntülerden elde edilen bilgiler bu modellere uygulanmak suretiyle çözüm elde edilir. Görünüş tabanlı sistemlerde ise izleme bir modele ihtiyaç duymadan daha alt düzeydeki görüntü özniteliklerinden faydalanılarak yapılır.

Bilgisayar görmesine dayalı pasif sistemlerle insan hareketinin gerçek-zamanlı olarak izlenmesinde kullanılan model ve görünüş tabanlı yöntemler, problemin modellenmesi açısından farklılık gösterse de, temeldeki hedef çok geniş bir çözüm uzayında uygun bir çözüm bulmaktır. Bu bağlamda, literatürdeki uygulamaların tamamına yakınında

problem, sahne, denek ve ortam ile ilişkili bir dizi varsayım kullanılarak çözülmeye çalışılır. Yapılan çalışmalarda sıklıkla kullanılan varsayımlar Tablo 1'de verilmiştir.

Tablo 1.1. Hareket İzlemede Kullanılan Varsayımlar

Varsayım	
Harekete İlişkin	**Görünüme İlişkin**
Denek çalışma alanı içinde kalmalıdır.	Ortamda sabit ışık olmalıdır.
Kamera hareketi yok ya da sabit olmalıdır.	Sahne durağan arka plana sahip olmalıdır.
Aynı anda sahnede yalnızca tek bir denek olmalıdır.	Sahne tekdüze arka plana sahip olmalıdır.
Denek sürekli olarak kameraya bakmalıdır.	Kamera parametreleri bilinmelidir.
Çakışma olmamalıdır.	Özel donanım kullanılmalıdır.
Denek yavaş ve sürekli hareket etmelidir.	Deneğin ilk pozisyonu bilinmelidir.
Deneğin hareket biçimi bilinmelidir.	Denek bilinmelidir.
Denek düz bir zeminde hareket etmelidir.	Deneğin üzerinde işaretçi olmalıdır.
	Denek özel renkli kıyafet giymelidir.
	Denek kendisini saran kıyafet giymelidir.

Bilgisayar görmesine dayalı pasif sistemler ile insan hareketinin izlenmesi ile ilgili olarak 1980 yılından itibaren yapılmış çalışmalar [11], [14] ve [23] deki incelemelerde detaylı olarak ele alınmıştır. Bu çalışmalarda, literatürde görünüş tabanlı hareket izlemede kullanılmış olan yöntemler dayandıkları temel esaslara göre Tablo 2'deki gibi kategorize edilmiştir. Tablo 2'deki kategorilerden herhangi birine dâhil olan bir yöntem farklı çalışmalarda tek başına kullanılabildiği gibi, aynı çalışmada farklı kategorilere ait yöntemlerin birlikte kullanılması da söz konusudur.

Tablo 1.2. Hareket İzlemede Kullanılan Yöntemler

Kategori	İşleyiş Esası
Optik Akış	Ardışık imge dizilerindeki farktan yola çıkılarak hareketin yörüngesinin belirlenmesi
Siluet Çıkarımı	İzlenen deneğin arka plandan ayrılması
Öznitelik Çıkarımı	İmgedeki kenar, ana hat, ya da bölge gibi özniteliklerin çıkarımı
Çoklu-kamera Kullanımı	Sahneyi farklı açılardan gösteren imgelerinin birleştirilmesi
Veri tabanı Karşılaştırması	Denek hareketlerinin daha önce kaydedilmiş olan hareket veri tabanı ile karşılaştırılması
Olasılıksal Çıkarım	Sahne arka planına ait renk, bölge, kenar ile deneğe ait antropometrik ölçü ve hareket değişimi gibi özniteliklerin olasılıksal çıkarımı

İnsan hareketinin izlenmesi probleminde çözüm uzayının oldukça geniş olmasının yanı sıra birçok uygulama için geçerli olan gerçek-zaman kısıtlaması araştırmacıları son yıllarda olasılığa dayalı çıkarım temelinde işleyen yöntemlere yöneltmiştir. Olasılığa dayalı çıkarım asıl olarak zamanla değişen dinamik sistemler için önceki zaman adımlarında (sistemin durumları arasındaki geçiş ilişkisi ve ölçümlerden yararlanılarak) elde edilen bilgilerin mevcut zaman adımında kullanılması yoluyla çözüm uzayının daraltılmasını sağlayan bir işleyişe sahiptir. Olasılığa dayalı çıkarım yöntemleri, literatürde Bayesçi filtreleme yöntemleri olarak bilinmektedir. Bayesçi filtreleme yöntemleri, özellikle son birkaç yılda insan hareketinin gerçek-zamanlı olarak izlenmesi probleminin çözümünde yoğun olarak kullanılmaktadır. Bayesçi filtreleme yöntemleri içerdikleri (imgelerden kullanılabilir bilgi çıkarmayı sağlayan algoritma olarak tanımlanan) gözlem modeli ile birbirlerinden ayrılmaktadır. Bu bağlamda, mevcut yöntemlerden daha iyi bir gözlem modeli içerecek yeni yöntemlere her zaman için ihtiyaç olduğu söylenebilir.

1.1. Araştırmada İzlenen Yol ve Katkılar

Bu çalışmada, kestirim teorisi ve filtreleme araştırma alanlarının tarihsel geçmişi önemli kilometre taşlarıyla birlikte, Bayesçi bir bakış açısıyla ele alınmıştır. Bayesçi kestirimin temelleri ile iyi bilinen Bayesçi optimal ve optimal-altı yöntemler, tarihi gelişimleri, avantaj ve dezavantajları ve uygulamalarıyla birlikte sunulmuştur. Bayesçi

kestirimin son yıllarda en çok tercih edilen yöntemlerinden birisi olan Parçacık filtreleme ise bu çalışmanın ana konusudur. Parçacık filtreleme, teorisinden başlamak üzere denklemlerinin elde edilişi ve bu yöntemle herhangi bir problemin nasıl modellenebileceği detaylı bir şekilde ele alınarak bu konuda yapılan çalışmalar kategorik olarak incelenmiştir. Ayrıca, bu tez çalışması çerçevesinde geliştirilen veri birleştirmeye dayalı parçacık filtreleme ile gerçek-zamanlı insan hareketi izleme yöntemi bütün yönleri ve benzer çalışmalara göre sağladığı katkılarla birlikte ortaya konulmuştur. Son olarak da Parçacık Filtreleme ile veri birleştirmeye dayalı insan hareketi izleme probleminin çözümü verilerek elde edilen sonuçlar değerlendirilmiştir.

Aslında bugün Bayesçi kestirim ile ilgili önemli sayıda ders notu, öğretici makale, inceleme, araştırma makalesi ve kitap mevcuttur. Bu çalışmanın, yapılmış çalışmalardan üstün olduğu iddia edilmemekle birlikte, söz konusu çalışmaların birçoğunda görülen eksikliklerin bu çalışmadaki bakış açısıyla kapatılması hedeflenmiştir. İlgili çalışmaların önemli bir kısmı mühendislik camiasına yabancı olan birtakım özel problemlerin çözümüne yönelik olarak farklı terminoloji ve olasılık teorisine ait çeşitli karmaşık çıkarımlar içeren çalışmalar olup fizikçi ya da istatistikçiler tarafından yapılmıştır. Çalışmaların bir kısmında parçacık filtresinin elde edilişi ile ilgili olarak verilen denklemler birbirinden kopuk şekildedir ve arka plandaki sürecin ardışık parçalarını birleştirmeyi zorlaştırmaktadır. Diğer bazı çalışmalarda, sayısal ve ilerlemeye sevk eden örneklerin mevcut olmayışı, konunun anlaşılmasını çok güçleştirmektedir. Bu çalışmada ise, konuya yeni başlayan araştırmacıların da faydalanabileceği şekilde adım adım ilerleme yaklaşımı benimsenmiştir. Konunun felsefi temelindeki basit Bayes kuralından parçacık filtrelemeye gelene kadar aradaki tüm aşamalar, adım adım açıklanmış ve matematiksel gösterimleri, uyumlu bir notasyon kullanılarak verilmiştir. Bu yaklaşım ile araştırmacıların oldukça soyut kavramlar içeren olasılık tabanlı kestirim konusunu daha etkin şekilde kavraması için bir temel oluşturulmuştur.

Bunun yanında, çalışmada, biri örnek diğeri ise çalışmanın temel konusu olan gerçek-zamanlı hareket izleme olmak üzere iki problem, kullanılan başlangıç ve gürültü parametrelerinin sayısal değerleri verilerek parçacık filtreleme algoritmasıyla çözülmüş ve bulgular değerlendirilmiştir. Böylece, bu çalışmada yapılan uygulamalar

Bayesçi kestirim alanında çalışan araştırmacılara deneyimlerini geliştirmede başvuru kaynağı olarak kullanma imkânı sağlanmıştır.

Bu çalışmada, sunulan diğer temel bir katkı da; günümüz bilişim dünyasının öncü ve en büyük şirketlerinden birisi olan Microsoft'un, özellikle gerçek zamanlı oyun deneyimleri için tasarlamış olduğu ve insan iskelet sistemini bir model ile gerçek-zamanlı olarak izleyen Kinect cihazının özellikle çakışma durumlarında kendi iç-tanımlı algoritmalarıyla ürettiği hatalı sonuçlara iyileştirme getirmesidir. Çalışmada kullanılan parçacık filtreleme yönteminin gözlem modelinde, Kinect cihazından alınan RGB ve derinlik imgeleri çeşitli imge işleme teknikleriyle analiz edilerek her ikisinden de yeni bilgiler çıkarılmak suretiyle çakışma durumları için cihazın kendi algoritmalarıyla bulduğu sonuçlardan daha iyi sonuçlar elde edildiği gösterilmiştir.

1.2. Tezin Organizasyonu

Karmaşık bir arka planda hareket eden insan deneğin el avuç-içi merkezinin olasılıksal örneklemeye dayalı parçacık filtreleme yöntemi kullanılarak gerçek zamanda tespit edilip takip edilmesinin hedeflendiği bu tez çalışmasında, öncelikli olarak dinamik sistem tanımı ve özellikleri ele alınarak durum uzayı gösterimi ile dinamik sistemlerin matematiksel modeli açıklanacaktır. Daha sonra, dinamik sistemlerin durumlarına ilişkin kestirim yapabilmeyi sağlayan kestirim teorisinin geçmişi, gelişimi ve özellikleri ile temel uygulamalarından birisi olan filtrelemeden bahsedilecektir.

Bu çalışmanın temel odak noktası olasılığa dayalı kestirim olduğundan, daha sonra Bayesçi kestirimle ilgili notasyon açıklamalarıyla birlikte verilerek; tanımı, özellikleri, geçmişi, gelişimi ve kullanım alanları açıklanacaktır. Çalışmada kullanılan Parçacık filtrelemenin temeli olan öz-yinelemeli Bayesçi kestirimin elde edilişi adım adım anlatılarak yaygın şekilde kullanılan optimal Kalman filtresi (KF) ve Genişletilmiş Kalman filtresi (EKF), İlerletilmiş Kalman filtresi (UKF), Gauss toplamı filtresi (GSF), Izgara tabanlı filtreler (GBF), ve Markov Zinciri Monte Carlo (MCMC) gibi optimal altı Bayesçi yöntemlerin özellikleri, kullanım alanları ele alınacaktır. Son olarak ise, bu tez çalışmasında temel yöntem olarak kullanılan Parçacık filtrelemenin diğer Bayesçi kestirim yöntemlerine göre güçlü yönleri izah edilecektir.

Üçüncü bölümde Parçacık filtrelemenin özellikleri detaylı olarak ele alınarak, performans artırımı sağlayan yeniden örnekleme yöntemlerinde bahsedilecektir. Bu bölümde ayrıca Parçacık filtresi kullanılarak bir problemin nasıl formülize edilebileceği izah edilerek, tekdüze bir arka planda serbestçe hareket eden farklı boyut ve renklere sahip üç adet dairenin (top) sahne üzerindeki yörüngelerinin izlenmesi örnek problemi parçacık filtreleme kullanılarak çözülecek ve sonuçları yorumlarıyla birlikle açıklanacaktır.

Dördüncü bölümde, bu çalışmada ele alınan insan hareketinin gerçek-zamanlı izlenmesi problemiyle ilgili tanım, veri akışı, ortam, kurulum, başlangıç koşulları verilecek, kullanılan yeniden örnekleme aşamasını içeren parçacık filtreleme algoritması (Sequential Importance Sampling Resampling – PF-SIR) adım adım izah edilecektir. Parçacık filtresinin bu çalışmada ele alınan probleme uygulaması sürecinde gerekli olan dinamik ve gözlem modelleri verilerek bu modeller detaylı olarak incelenecektir. Ayrıca, gözlem modelinde uygulanan veri birleştirme sürecinin işleyişi anlatılarak veri birleştirmenin sağladığı performans kazanımı değerlendirilecektir.

Beşinci bölümde, elde edilen bulgular verilerek performans değerlendirmesi yapılacaktır. Son bölümde, bu tez çalışmasında uygulanan yöntemin sonuçları paylaşılarak, farklı alanlardaki gerçek-dünya problemlerine nasıl uyarlanabileceği üzerinde durulacak ve gelecek çalışmaları ile ilgili bir perspektif sunulacaktır.

BÖLÜM 2. DİNAMİK SİSTEMLERDE BAYESÇİ KESTİRİM VE FİLTRELEME

2.1. Dinamik Sistemler

Dinamik sistem kavramını anlayabilmek için öncelikle teknik bir perspektiften sistem tanımını yapmak gerekir. Sistem birbiriyle karşılıklı ilişki içerisinde olan bileşenlere sahip bütünsel bir yapıdır. Eğer bir sistemin davranışları zaman içinde değişiyorsa bu durumda bu sistem dinamik olarak nitelendirilir. Dinamik bir sistem, ardışık durumları arasındaki ilişki bakımından lineer ya da non-lineer olabilir. Dinamik bir sistemin davranışları, ardışık durumları arasındaki geçiş ilişkisine bağlı olarak matematiksel modeller yoluyla gözlemlenir. Çoğunlukla, daha doğru ve genel sonuçlar bulma gereksiniminden dolayı bazı parametrelerin değerlerinde diğerlerine göre meydana gelen oldukça küçük değişimleri incelemek gerekir. Bu bağlamda, zamanda sürekli dinamik sistemler genellikle sistemin durum vektörlerinin hız ve ivmelerindeki değişimlere bakılarak gözlemlenir. Böyle durumlarda, incelenen dinamik sistemin matematiksel modelini elde etmek için sırasıyla hız ve ivme değişimlerine karşılık gelecek şekilde birinci ve ikinci dereceden türevler kullanılır. Bazı spesifik dinamik sistemler için ise sistemin durumunu belli zaman adımlarında bilmek gerekir. Bu tür sistemler için, sistemin bir sonraki zaman adımı mevcut zaman adımının bir fonksiyonu olarak modellenir ve modellerde diferansiyel denklemler yerine fark denklemleri kullanılır.

Gerçek dünyadaki hemen hemen tüm sistemler, dinamik sistemler olarak varlığını sürdürür, lineer olmayan durum geçiş fonksiyonlarına sahiptir. Ayrıca parametreleriyle ilgili olarak Gauss dağılımına uymayan gürültü gibi yüksek dereceli belirsizlik içerirler. Dinamik sistemlerle ilgili durum kestirimi yapmak için öncelikle bunların matematiksel olarak gösterilmesi gerekir. Durum uzayı modelleri bu amaçla

kullanılan en yangın gösterim biçimidir [12]. Durum uzayı gösteriminde dinamik ve gözlem modeli olmak üzere iki model mevcuttur:

Dinamik model, istenen sistem parametrelerinin kestirimini içeren durum vektörünün zamandaki evrimini tanımlar:

$$x_k = f_{k-1}(x_{k-1}, v_{k-1}), k > 0 \quad \textbf{(2.1)}$$

Burada, x_k kestirilecek durum vektörü; k zaman adımı; f_{k-1} lineer olmayan bir fonksiyon; ve v_{k-1} sistem ya da proses gürültüsü olarak tanımlanır.

Gözlem modeli durum vektörü ve elde edilen ölçümler (gözlemler) arasındaki ilişkiyi tanımlamak için kullanılır.

$$z_k = h_k(x_k, w_k), k > 0 \quad \textbf{(2.2)}$$

Burada, z_k, k zaman adımında elde edilen ölçüm vektörü; h_k ölçüm fonksiyonu ve w_k da ölçüm gürültüsü olarak adlandırılır.

2.2. Kestirim Teorisi ve Filtreleme

Kestirim teorisi, deneysel bulgu ve istatistiksel analizlere dayalı olarak sistem parametrelerinin kestirimi ile uğraşan bir araştırma alanıdır. Kestirim teorisi yöntemleri, çoğu zaman doğrudan ölçülemeyen ancak gürültülü ölçümler vasıtasıyla ulaşılabilen bilinmeyen bir büyüklüğün (örneğin bir geminin radar merkezine uzaklığı) değerini mümkün olan en yüksek doğrulukta kestirmeye çalışırlar [4],[25]. Kestirim teorisinin işaret işleme, kalite kontrol, telekomünikasyon, yazılım mühendisliği, proje yönetimi ve ağ çarpışma denetimi gibi çok sayıda alanda uygulamaları mevcuttur [26]. Temelini Kestirim teorisinden alan yöntemlerdeki temel amaç, her zaman için ölçümleri kullanarak sistem parametrelerinin değerini tahmin edecek optimal (mümkün olan tüm yararlı bilginin çıkarılıp kullanıldığı) ve gerçeklenebilir bir kestirimci oluşturmaktır [27]. Araştırmacılar, kestirim ve filtrelemeyi genellikle aynı anlamda ele almışlardır [4],[28]. Filtreleme farklı araştırma alanları için farklı tanımlara sahip olmakla birlikte, genel olarak işaretlerin istenen ve istenmeyen

kısımlarının ayrılması süreci olarak tanımlanır [28]. Böylece işaretin istenen kısmı, kendi başına, istenen amaca yönelik olarak kullanılabilir hale gelir. Filtreleme bu anlamda, kestirim teorisinin bir uygulaması olarak düşünülebilir. Filtreleme yöntemleri, sistemin ardışık durumları arasındaki ilişki ve ölçümlerden faydalanarak belli bir aksiyon için temel oluşturmak amacıyla sistemin davranışlarını ortaya çıkarmayı hedeflerler. Gerçek-dünya sistemlerinin durumları genellikle gizlidir ve kullanışlı bilgi ancak istenmeyen gürültüyle bozulmuş gözlemlerden elde edilebilir. Bu bağlamda, filtreleme birçok gerçek-dünya sistemi için temel uygulamalardan birisi olarak düşünülebilir.

Kestirim, teknik bir araştırma alanı olarak bilimsel literatüre ilk kez gezegen hareketlerinin kestirimi üzerine yaptığı çalışmalarla K.F. Gauss tarafından taşınmıştır. Kestirim teorisinin referans yöntemi olarak kabul edilen "En Küçük Kareler (Least Squares – LS)" yöntemi ilk kez 17. yüzyılın sonunda Gauss tarafından ve sonrasında da 18. yüzyılın başında Legendre tarafından birbirinden bağımsız olarak geliştirilmiştir [25]. Kestirim teorisindeki ikinci kilometre taşı, 1912 yılında R.A. Fisher tarafından sunulan "En Çok Olabilirlik (Maksimum Likelihood – MLE)" yöntemidir. Fisher, aynı zamanda "parameter", "likelihood", "Bayesian" gibi çok önemli kavramları da ilk kez kullanan araştırmacıdır [29]. MLE yöntemi, istatistikteki birçok çıkarıma dayalı yöntemin temeli olarak kabul edilir [30]. Kestirim teorisi alanındaki diğer önemli bir dönüm noktası da, 1941 yılında Kolmogorov ve arkasından 1942 yılında Wiener tarafından birbirlerinden bağımsız olarak geliştirilen " Doğrusal Minimum Ortalama Karesel Hata Kestirimi (Lineer Lineer Minimum Mean Square Estimation)" yöntemidir. Bu yöntemin önemi, aslında, kestirim teorisindeki en büyük gelişme olarak sayılabilecek Kalman Filtresi için bir temel oluşturmasından kaynaklanmaktadır [25].

1960 yılında R.E.Kalman tarafından geliştirilen Kalman filtresinin kökeni aslında en küçük kareler yöntemine dayanmaktadır. Bununla birlikte, Kalman filtresi yönteminin zamanla değişen sistemlerin gerek sürekli gerek ayrık zamanlı gözlemlerle kestirimine uygun olması, integral denklemleri yerine fark ya da diferansiyel denklemlerin kullanımına imkân veren durum uzayı gösteriminin ortaya çıkmasıyla problem formulasyonlarının değişmesi ve tabii ki yöntemin rekürsif algoritmaların etkin şekilde

kullanılabilmesine olanak sunması Kalman filtresinin çok çeşitli dallardaki araştırmacılar arasında çok popüler hale getirmiştir. Bunun yanında, Kalman filtresi denklemlerinin dijital bilgisayarla gerçeklemeye oldukça uygun olması ve gerçek-zamanlı kestirime imkân tanıması en azından takip eden 30 yıl içinde filtreleme alanında çok büyük gelişmelerin yaşanmasını beraberinde getirmiştir [25]. Kalman filtresinin geliştirilmesinden sonra, kestirim teorisi uygulamaları uydu navigasyonu, gezegen yörünge kestirimi, kontrol teorisi, video izleme sistemleri, geodezi araştırmaları, neuro-science, makine öğrenmesi, telekominikasyon, işaret işleme, ekonomi, finans, politik bilimler, yöneylem araştırması ve bio-mühendislik gibi çok geniş ve farklı alanlarda 1990'lı yılların başlarında kadar artan bir şekilde kendilerini göstermişlerdir [31]. Kalman filtresinin detaylarına ilişkin teori, denklemler ve uygulamalar [32], [33] ve [34] de bulunabilir.

Kalman filtresi, aslında ilk olarak ortalama karesel hatayı minimize eden bir yöntem olarak geliştirilmiştir. Bununla birlikte, daha sonraki yıllarda bu yöntem, bu çalışmanın da temel yapısını oluşturan Bayesçi kestirimin tanım ve yöntemleriyle tekrar gerçekleştirilmiştir [4].

2.3. Dinamik Sistemlerin Olasılıksal Gösterimi

Dinamik sistemlerin olasılıksal gösterimi, özellikle Bayesçi yöntemlerin kullanıldığı durumlarda tercih edilen bir yoldur. Böyle durumlarda Bayesçi yöntemler, sistem dinamiklerinin stokastik olarak modellenmesi ve istenen parametrelerin istatistiksel özelliklerinin çıkarılması yoluyla çözümün yaklaşık değerinin bulunmasını hedeflerler. Bunun için, olasılığa dayalı modeller kurulurken konuşma sinyali, dijital bilgisayar verileri, ya da gürültü gibi dinamik süreç ve belirsizlikler stokastik süreçler olarak varsayılır. Stokastik bir süreç, olasılık kurallarına göre istatistiksel bir olayın zaman içindeki evrimini açıklamak için kullanılan ve k zaman adımıyla indekslenerek $\{x_k\}$ olarak gösterilen bir rassal değişken dizisi olarak tanımlanır.

Aslında, stokastik süreçlerle çalışılırken oldukça fazla sayıda rassal değişken ile uğraşılır. Rassal değişkenlerin sayısı sonlu olabileceği gibi gözlem altındaki stokastik

sürecin parametre uzayına bağlı olarak sonsuz da olabilir. Bir rassal değişkenin olasılık dağılımı, onu tanımlayan en temel özniteliktir. Bir rassal sürecin olasılıksal karakteristiği, o süreci oluşturan rassal değişkenlerin birleşik olasılık dağılımından elde edilir. Bundan dolayı, bilgisayarlı çözümlere olarak sunan stokastik süreçler olarak modellenen dinamik sistemler için rassal değişkenlere ait birleşik olasılık dağılımı ile ortalama, kovaryans, ve korelasyon gibi istatistiksel parametreler üzerinde çalışılır. Bu bağlamda, durumu stokastik bir Markov sürecinin karakteristiğini taşıdığı kabul edilen dinamik bir sistemin sonsal olasılık yoğunluk fonksiyonu denklem (2.3) ile ifade edilir:

$$p(x_{0:k}) \sim p(x_0) \times \prod_{i=1}^{k} p(x_i|x_{i-1}) \tag{2.3}$$

Burada, $p(x_0)$, 0. zaman adımındaki ilksel olasılık dağılımını ve $\sim$ operatörü de bağımsız ve özdeş olarak dağılmış (independent and identically distributed) rassal değişkenlerin bir diziden örneklenmesi (sampling) işlemini karakterize etmektedir. Dinamik modelin oluşturulmasına benzer şekilde, $p(z_k|x_k)$ gözlem benzerliği (observation likelihood) olacak şekilde gözlem modeli de denklem (2.4) 'deki gibi ifade edilir.

$$p(z_{1:k}|x_{0:k}) \sim \prod_{i=1}^{k} p(z_i|x_i) \tag{2.4}$$

Böyle bir gösterim yoluyla, dinamik sistemin davranışları incelenir ve durum geçiş ve gözlem denklemleri koşullu olasılıklar şeklinde modellenerek istenen parametrelerin değerleri olasılıksal olarak kestirilir.

2.4. Bayesçi Kestirim

Bayesçi çıkarım, Bayes kuralı ve olasılık teorisine ait bir dizi diğer yöntemi kullanan genel bir çıkarım methodolojisidir [26]. Bayesçi çıkarımın temelinde 18. Yüzyıl matematikçi ve teoloji uzmanı olan Thomas Bayes tarafından geliştirilen ve kendi adıyla anılan Bayes kuralıdır. Aslında, Bayes teoreminin bugün bilinen genel hali 1814

yılında P.S. Laplace tarafından elde edilmiştir. Laplace günümüzde, Bayesçi çıkarımı geliştirmesi ve olasılığı farklı alanlara uygulaması üzerine yaptığı çalışmalar dolayısıyla olasılık alanındaki öncü araştırmacılar arasında yer alır. Bayesçi yaklaşımın altındaki temel felsefe, bir olay hakkındaki mevcut inancın elde edilen yeni deliller ışığında değişebilmesinin mümkün olduğu fikridir. Bu felsefeden yola çıkıldığında, zamanla değişen dinamik sistemlerin durum kestirimi probleminin Bayesçi çıkarım tarafından ele alınabileceği çok rahatlıkla anlaşılabilir. Bunun yanında, lineer ve Gauss gürültülü senaryolar için Bayesçi çıkarım optimal sonuçlar da üretebilir. Bu bağlamda, literatürde Optimal Bayesçi Filtreleme olarak da anılmaktadır [25],[26]. Optimal Bayesçi filtrelemede, durum uzayı gösterimi dinamik ve gözlem modellerinin Bayesçi yaklaşıma çok daha uygun olan olasılıksal biçimleriyle değiştirilir. Amaç, sistem üzerinde yapılan gözlemlerden elde edilen tüm verilerden çıkarılan bilgiyle birlikte sistem durumunun sonsal olasılık yoğunluk fonksiyonunu (posterior probability density function) oluşturmaktır. Sonsal olasılık yoğunluk fonksiyonu, sistemin durumuna ait tüm istatistiksel bilgiyi taşıması dolayısıyla, optimal bir kestirime imkan vermesi nedeniyle kestirim probleminin çözümü olarak düşünülür [35]. Birçok kestirim problemi için yeni bir gözlem yapıldıktan hemen sonra bir kestirim yapmak gereklidir. Bu durumda, elde edilen verinin ardışık olarak işlenmesine olanak tanıyan rekürsif bir filtre uygun çözüm olacaktır. Optimal Bayesçi filtreleme, literatürde aynı zamanda "Öz-yinelemeli Bayesçi Kestirim (Recursive Bayesian Estimation – RBE)" olarak da adlandırılır. Bu öz-yinelemeli filtreleme yaklaşımında iki temel adım mevcuttur: Tahmin (prediction) ve güncelleme (update). Sistem işleyiş gürültüsünü de içerek dinamik model durumun sonsal olasılık yoğunluk fonksiyonu kestirebilmek için tahmin adımında kullanılırken, güncelleme adımında en son elde edilen gürültülü gözlem, gözlem modeli ile birlikte yapılan tahminin değerinin iyileştirilmesinde kullanılır. Öz-yinelemeli Bayesçi kestirimde, temelinde Bayes kuralı işleyen bu iki adım, birtakım istenen koşullar oluşuncaya kadar birbiri ardından tekrar tekrar uygulanır.

2.4.1. Öz-yinelemeli Bayesçi kestirim

Sıralı Monte Carlo (Sequential Monte Carlo – SMC) olarak da bilinen Parçacık filtresi, sonsal olasılık yoğunluk fonksiyonunun yaklaşık değerini hesaplamak amacıyla öz-

yinelemeli şekilde önem örneklemesinin gelişmiş bir şeklini uygulayan yöntemler için kullanılan genel bir addır. Parçacık filtresi, öz-yinelemeli Bayesçi kestirim kullanmak suretiyle dinamik sistemlerinin durumlarını analiz etmek için bir temel oluşturur.

Ayrık zamanlı dinamik bir sistemin parametrelerini makul bir doğrulukla kestirebilmek için sistem dinamikleri ile ilgili edinilmiş her bilginin olasılıksal modele ekenmesi gereklidir. Böyle sistemlerde, sistem devamlı olarak gelişim/değişim gösterse de ölçümler belli zaman adımlarında yapılır. Yeni bir ölçüm yapmadan, mevcut adımdaki ölçüm sonuçları sistemin bir sonraki durumuna ilişkin parametrelerin kestirimi için gereklidir. Bu tip bir kullanım için öz-yinelemeli bir mekanizmanın kullanılması gerektiği açıktır. Böyle bir mekanizma, Markov zinciri varsayımlarıyla destekli şekilde Bayes Kuralı temel alınarak kurulabilir.

Sonlu durumlu ayrık zamanlı dinamik bir sistem, zamanda durağan bir dağılım gösteren ve şu anki durumu yalnızca bir önceki durumuna bağlı olan bir Markov süreci şeklinde modellenir. Markov sürecinin temel iki varsayımı aşağıda sırasıyla (2.5) ve (2.6) da verilen denklemlerde gösterilmiştir.

Sistemin şu anki durumu, bir önceki zaman adımındaki durumu hariç geçmişteki tüm durumları ve bu zaman adımlarında elde edilmiş ölçümlerden bağımsızdır. Bu varsayım denklem (2.5) ile ifade edilir.

$$p(x_k|x_{0:k-1}, z_{1:k-1}) = p(x_k|x_{k-1}) \quad \textbf{(2.5)}$$

Burada, $x_k \in \mathbb{R}^n$, k zaman adımındaki durum vektörünü, $z_k \in \mathbb{R}^m$ de ölçüm vektörünü göstermektedir.

Şu anki zaman adımında elde edilmiş ölçüm bilgisi, sistemin şu anki durumu hariç geçmiş tüm zaman adımları ve bu zaman adımlarında elde edilmiş ölçümlerden bağımsızdır. Bu varsayım denklem (2.6) ile ifade edilir.

$$p(z_k|z_{1:k-1}, x_{0:k}) = p(z_k|x_k) \quad \textbf{(2.6)}$$

Öz-yinelemeli Bayesçi kestirimde, ayrık zamanlı dinamik sistemin durumunu temsil eden rassal değişkenlerin birleşik dağılımı, her adımda Bayes Teoremi, Olasılık teorisinin zincir kuralı ve yukarda sözü edilen Markov süreci varsayımları kullanılarak ifade edilir. İlk olarak, rassal değişkenlerin birleşik dağılımı Bayes kuralı kullanılarak durum geçiş olasılıkları cinsinden denklem (2.7) deki gibi yazılır.

$$p(A|B) = \frac{p(B|A) \times p(A)}{p(B)} \rightarrow p(x_{0:k}|z_{1:k}) = \frac{p(z_{1:k}|x_{0:k}) \times p(x_{0:k})}{p(z_{1:k})} \qquad \textbf{(2.7)}$$

Burada, $p(x_{0:k}|z_{1:k})$, k zaman adımına kadar, sistemin tüm durumlarının eldeki tüm ölçümlere göre koşullu birleşik olasılık dağılımını gösterir. $p(z_{1:k}|x_{0:k})$ ise yine k zaman adımına kadar ölçümlerin durumlara göre koşullu olasılığını (ölçümlerin sistemin durumuna ne kadar uygun olduğunu) ifade eder. (Denklem 2.8)

$$p(z_{1:k}|x_{0:k}) = p(z_k, z_{1:k-1}|x_{0:k}) \qquad \textbf{(2.8)}$$

Olasılık zincir kuralı kullanılarak denklem (2.8) oluşturulabilir. Denklem (2.8) de $p(z_{1:k}|x_{0:k})$ ifadesinin eşiti olan $p(z_k, z_{1:k-1}|x_{0:k})$ ifadesi, denklem (2.7) de $p(z_{1:k}|x_{0:k})$ yerine konularak denklem (2.9) elde edilir.

$$p(x_{0:k}|z_{1:k}) = \frac{p(z_k, z_{1:k-1}|x_{0:k}) \times p(x_{0:k})}{p(z_{1:k})} \qquad \textbf{(2.9)}$$

$$p(z_k, z_{1:k-1}|x_{0:k}) = p(z_k|z_{1:k-1}, x_{0:k}) \times p(z_{1:k-1}|x_{0:k}) \qquad \textbf{(2.10)}$$

Denklem (2.10) da, yine denklem (2.8) 'in elde edilişine benzer şekilde olasılık zincir kuralı kullanılmak suretiyle elde edilir. Denklem (2.10) 'daki $p(z_k, z_{1:k-1}|x_{0:k})$ ifadesinin eşiti denklem (2.9)'da yerine konularak denklem (2.11) elde edilir.

$$p(x_{0:k}|z_{1:k}) = \frac{p(z_k|z_{1:k-1}, x_{0:k}) \times p(z_{1:k-1}|x_{0:k}) \times p(x_{0:k})}{p(z_{1:k})} \qquad \textbf{(2.11)}$$

$$p(z_{1:k-1}|x_{0:k}) = \frac{p(x_{0:k}|z_{1:k-1}) \times p(z_{1:k-1})}{p(x_{0:k})} \qquad \textbf{(2.12)}$$

Bayes kuralı kullanılarak denklem (2.12)'deki eşitlik kolayca yazılabilir. Denklem (2.12)'deki $p(z_{1:k-1}|x_{0:k})$ ifadesinin karşılığı denklem (2.11)'deki yerine konularak denklem (2.13) elde edilir.

$$p(x_{0:k}|z_{1:k}) = \frac{p(z_k|z_{1:k-1}, x_{0:k}) \times p(x_{0:k}|z_{1:k-1}) \times p(z_{1:k-1}) \times p(x_{0:k})}{p(z_{1:k}) \times p(x_{0:k})} \quad \textbf{(2.13)}$$

Denklem (2.13) üzerinde gerekli sadeleştirmeler yapılarak denklem (2.14) aşağıdaki gibi elde edilir.

$$p(x_{0:k}|z_{1:k}) = \frac{p(z_k|z_{1:k-1}, x_{0:k}) \times p(x_{0:k}|z_{1:k-1}) \times p(z_{1:k-1})}{p(z_{1:k})} \quad \textbf{(2.14)}$$

$$p(z_{1:k}) = p(z_k|z_{1:k-1}) \times p(z_{1:k-1}) \quad \textbf{(2.15)}$$

Denklem (2.15) olasılık zincir kuralı bir kez daha kullanılarak yazılabilir. $p(z_{1:k})$ ifadesinin denklem (2.15) 'deki eşiti, denklem (2.14)'de yerine konulduğunda denklem (2.16) elde edilir.

$$p(x_{0:k}|z_{1:k}) = \frac{p(z_k|z_{1:k-1}, x_{0:k}) \times p(x_{0:k}|z_{1:k-1}) \times p(z_{1:k-1})}{p(z_k|z_{1:k-1}) \times p(z_{1:k-1})} \quad \textbf{(2.16)}$$

Denklem (2.16) üzerinde gerekli sadeleştirmeler yapılarak denklem (2.17) aşağıdaki gibi elde edilir.

$$p(x_{0:k}|z_{1:k}) = \frac{p(z_k|z_{1:k-1}, x_{0:k}) \times p(x_{0:k}|z_{1:k-1})}{p(z_k|z_{1:k-1})} \quad \textbf{(2.17)}$$

Bu aşamada, denklem (2.6) ile verilen 2. Markov süreci varsayımı durum geçiş olasılıkları cinsinden denklem (2.18) deki gibi yazılır.

$$p(x_{0:k}|z_{1:k}) = \frac{p(z_k|x_k) \times p(x_{0:k}|z_{1:k-1})}{p(z_k|z_{1:k-1})} \quad \textbf{(2.18)}$$

$$p(x_{0:k}|z_{1:k-1}) = p(x_k|x_{0:k-1}, z_{1:k-1}) \times p(x_{0:k-1}|z_{1:k-1}) \quad \textbf{(2.19)}$$

Denklem (2.19) olasılık zincir kuralı kullanılarak oluşturulabilir. Denklem (2.18) 'deki $p(x_{0:k}|z_{1:k-1})$ ifadesi, denklem (2.19)'daki karşılığı ile değiştirilerek denklem (2.20), aşağıdaki gibi elde edilir.

$$p(x_{0:k}|z_{1:k}) = \frac{p(z_k|x_k) \times p(x_k|x_{0:k-1}, z_{1:k-1}) \times p(x_{0:k-1}|z_{1:k-1})}{p(z_k|z_{1:k-1})} \quad \textbf{(2.20)}$$

$$p(x_{0:k}|z_{1:k}) = \frac{p(z_k|x_k) \times p(x_k|x_{k-1}) \times p(x_{0:k-1}|z_{1:k-1})}{p(z_k|z_{1:k-1})} \quad \textbf{(2.21)}$$

Denklem (2.21), denklem (2.5) ile verilen ilk Markov süreci varsayımı kullanılarak oluşturulur.

Bayes kuralına göre, denklem (2.22)'de verilen (kanıt-evidence olarak tanımlanan) $p(z)$ ifadesi aslında, pay kısmında verilen ifadenin marjinal olasılık dağılımıdır ve $p(x|z)$ sonsal olasılık dağılımının hesaplanmasında herhangi bir etkisi yoktur. Bu nedenle sabit bir katsayı olarak düşünülebilir.

$$p(x|z) = \frac{p(z|x) \times p(x)}{p(z)}, \qquad p(z) = \int p(z|x) \times p(x)dx \quad \textbf{(2.22)}$$

$$p(z_k|z_{1:k-1}) = \int p(z_k|x_k) \times p(x_{0:k}|z_{1:k-1})dx_k \quad \textbf{(2.23)}$$

$p(z_k|z_{1:k-1})$ ifadesinin denklem (2.23)'deki karşılığı, denklem (2.21)'de yerine konularak denklem (2.24) elde edilir.

$$\eta = \frac{1}{p(z_k|z_{1:k-1})} = \frac{1}{\int p(z_k|x_k) \times p(x_{0:k}|z_{1:k-1})dx_k} \quad \textbf{(2.24)}$$

Normalleştirme katsayısı olarak tanımlanan η, denklem (2.21)'de yerine konularak denklem (2.25) elde edilir.

$$p(x_{0:k}|z_{1:k}) = \eta \times p(z_k|x_k) \times p(x_k|x_{k-1}) \times p(x_{0:k-1}|z_{1:k-1}) \quad \textbf{(2.25)}$$

η bir katsayı olarak düşünüldüğü için, denklem (2.25) üzerinde bir düzenleme yapılabilir. Burada, $\propto$ operatörü, oransallık ifade edecek şekilde denkleme yerleştirilerek, η eşitlikten çıkarılır ve denklem (2.26) aşağıdaki gibi oluşturulur.

$$p(x_{0:k}|z_{1:k}) \propto p(z_k|x_k) \times p(x_k|x_{k-1}) \times p(x_{0:k-1}|z_{1:k-1}) \quad \textbf{(2.26)}$$

Burada, $p(x_{0:k}|z_{1:k})$ sistemin sonsal olasılık yoğunluk fonksiyonu, $p(z_k|x_k)$ ifadesi k zaman adımında, sistemin durumuna göre ölçümün olabilirliğini, $p(x_k|x_{k-1})$ ifadesi, geçiş önseli (transition / temporal prior), $p(x_{0:k-1}|z_{1:k-1})$ ifadesi de bir önceki zaman adımında sistemin sonsal olasılık yoğunluk fonksiyonu olarak ifade edilir.

Denklem (2.26) da verilen bu formulasyon ile, herhangi bir zaman adımında sistemin sonsal olasılık yoğunluk fonksiyonu, bir önceki zaman adımındaki sonsal olasılık yoğunluğu ile ilişkili olarak elde edilebilir hale gelir. Bu ilişki öz-yinelemeli Bayesçi kestirimin iki aşaması ile ifade edilir: tahmin ve güncelleme. Tahmin adımında $p(x_k|x_{k-1})$ dağılımı, mevcut zaman adımında ölçüm almadan tahmin edilir. $p(z_k|x_k)$ ifadesinin değeri mevcut zaman adımındaki ölçüm olabilirliği ile hesaplanarak önceki zaman adımına ait sonsal olasılık yoğunluk fonksiyonu $p(x_{0:k-1}|z_{1:k-1})$ ifadesinin değeri güncelleme aşamasında güncellenir.

2.4.2. Optimal-altı Bayesçi kestirim

Optimal Bayesçi filtreleme aslında dinamik ve ölçüm modelleri Gauss gürültüsü içeren ve lineer olan Kalman filtresinin genel bir hali olarak düşünülebilir [40]. Bununla birlikte, gerçek-dünya sistemlerinin birçoğu lineer değildir ve Gauss dağılımına uymayan gürültü içerir [5],[36]. Bu tip sistemler için kapalı form ya da analitik bir çözüm mümkün değildir [37],[38],[39],[40]. Bu durumdan kaynaklanmak üzere, çeşitli optimal-altı Bayesçi yaklaşım tabanlı filtreleme yöntemleri zamanla ortaya çıkmıştır. Bu yöntemlere ait bir sınıflandırma [39] ‘da verilmiştir. Sistem dinamik sürecinin Taylor serisine açılma yoluyla lineerleştirilerek daha sonra bilinen Kalman çözümümün uygulanmasına dayanan Genişletilmiş kalman filtresi (EKF), bu yöntemlerden ilki ve en çok bilinenidir [5],[41]. EKF çok farklı alanların problemlerinin çözülmesi amacıyla uzun yıllar kullanılmış olmasına rağmen, lineer

olmayan yüksek dereceli durum geçiş ilişkisine sahip sistemler için verdiği kötü sonuçlar dolayısıyla zamanla yerini başka yöntemlere bırakmıştır [41],[42]. Gauss toplamı filtresi (GSF), lokal lineerleştirmeye dayanan EKF'ye benzer diğer bir yöntemdir. Aslında GSF bir dizi EKF'nin paralel çalıştırılmasına dayalı bir yöntemdir [43]. GSF yönteminin altında yatan düşünce, istenen sonsal yoğunluğa Gauss yoğunluk fonksiyonlarının ağırlıklı bir ortalaması yoluyla ulaşmaktır [43]. GSF'nin performansı karışım için belirlenen bileşenlerin sayısına ve ölçümlere dayanan ağırlıklarına bağlıdır. GSF ile elde edilen kestirimlerin EKF'ye göre daha yansız olduğu görülmüştür [36]. GSF'nin kullanımı sonsal olasılık yoğunluğunun çok modlu olduğu durumlarda oldukça mantıklıdır. Diğer taraftan, bu yöntemin dezavantajı, Gauss karışımının ağırlıklarını sabit tutarken belirsizliği lineer olmayan sistem boyunca yaymak ve güncellemeyi sadece yeni bir ölçüm yapıldığında uygulamaktır [36]. İlerletilmiş Kalman filtreleme (UKF) , EKF ve GSF'ye göre nispeten daha yeni bir optimal-altı filtreleme yöntemidir. UKF, ortalama etrafında seçilen birkaç örnek noktanın lineer olmayan sistem boyunca yayılmasına dayanan istatistiksel bir lineerleştirme tekniğidir. Bu yöntemle, EKF ve GSF'ye göre hesaplama yükü daha az olan ve daha doğru sonuçlar üreten çözümler elde edildiği görülmüştür [36],[44].

Lineer olmayan Bayesçi yaklaşık filtreleme yöntemlerinden bir diğer grupta, ızgara tabanlı yöntemlerdir. Bu yöntemler, çok boyutlu integrallerin dolayısıyla da sonsal yoğunluğun değerini elde etmek amacıyla sayısal integrasyon kullanırlar [36]. Bu yöntemlerin temel açmazı, yüksek boyutlu bir durum uzayı varlığında hesaplama yükünün olağan dışı artmasıdır. Sonuç olarak, günümüzdeki lineer olmayan sıradan bir sistemin sahip olduğu yüksek boyutluluğa dair ızgara tabanlı yöntemlerin başarılı bir çözüm getirmesi pek mümkün değildir [37],[42].

Bayesçi yaklaşıma dayalı lineer olmayan durum kestirimi yöntem gruplarından bir diğeri de örnekleme tabanlı yöntemlerdir. Bu yöntemler aslında, öz-yinelemeli Bayes filtrelemenin Monte Carlo (MC) simülasyon tekniği kullanımıyla yapılan uygulamalarıdır [37].

MC simülasyon tekniği, kökenini 1770'lerde rulet masasına dayanan basit rassal sayı üretiminden alır. Bununla birlikte, 2.dünya savaşından sonra takip eden yıllar

içerisinde sırasıyla fizik, istatistik ve mühendislik alanındaki araştırmacıların ilgisini toplamıştır [37]. MC örnekleme yöntemini baz alan yöntemlerin temel prensibi, elde edilmek istenen sonsal yoğunluğu bir dizi ağırlıklı rassal örnekle temsil etmektir [45]. Kullanılan örnek sayısı arttıkça böyle bir gösterimin hedefteki sonsal olasılık dağılım fonksiyonuna dolayısıyla da optimal Bayesçi kestirime daha yakın bir tahmin üreteceği açıktır. MC yöntemlerinin avantaj ve dezavantajları mevcuttur. Diğer yöntemlere göre temel avantajları durum uzayının boyutundan bağımsız olarak yaklaşıklık hatası varyansının azaltılmasıdır. Diğer taraftan MC yöntemleri, iki temel dezavantaja sahiptir: yüksek boyutlu olasılık dağılımının örneklenmesinin zorluğu ve örneklemenin mümkün olması durumunda artan hesaplama karmaşıklığı [26]. Tahmin edilebileceği üzere, ikinci dezavantaj sürekli bir şekilde gelişen (işlemci, bellek vb.) hesaplama kaynakları ile aşılabilir bir problemdir. Diğer taraftan, MC yöntemleriyle örneklemenin zorluğu problemi, iki kategoride farklı yöntemlerin geliştirilmesine vesile olmuştur. Bunlar MCMC ve SMC yöntem gruplarıdır.

Metropolis-Hastings ve Gibbs gibi MCMC yöntemleri durağan dağılımı hedef dağılıma eşdeğer olan bir hedef dağılımından doğrudan örneklemeye dayanır [46]. MCMC yöntemlerinin detaylı bir açıklaması ve uygulamalarını içeren çalışma [47]'de bulunabilir. SMC yöntemleri literatürde farklı şekillerde isimlendirilmiştir: bootstrap filtreleme, condensation algoritması, parçacık filtreleme, "survival of the fittest" [37]. Parçacık filtreleme, bu çalışmada SMC için tercih edilen adlandırma olmasının yanında, bu çalışmanın da temel odak noktasıdır. Parçacık filtreleme, örneklenmesi zor olan hedef dağılım yerine, örneklenmesi daha elverişli olan başka bir dağılımın örneklenmesi fikrine dayanır [45]. Önerilen dağılım, önem fonksiyonu olarak da adlandırılır ve eldeki örneklerden hedef dağılıma ulaşmaya imkan veren bir dağılımdır. Böylelikle, hedef dağılıma nispeten önem derecelerine (ağırlıklarına) sahip olan örnekler elde etmek mümkün olur. Algoritmanın sonraki adımları, büyük ölçüde önem örneklemesiyle elde edilen örneklere bağlı olduğundan, önem fonksiyonunun seçimi hayati öneme sahiptir [37]. İyi seçilememesi durumunda başarısız çözümler ile karşılaşmak neredeyse kesindir.

Parçacık filtreleme ilk olarak 1993 yılında Gordon, Salmond ve Smith'in yaptığı çalışmayla literatüre sunulmuştur [48]. O zamandan bu yana, işaret işleme, robotik ve

bilgisayar görmesi alanlarında başarılı örnekler sunulmuştur [42],[49]. Parçacık filtresi [50], [51] ve [52] 'deki çalışmalarda verilen birkaç tane farklı uygulamaya sahip olmasına rağmen, öz-yinelemeli Bayesçi kestirim ve olasılığa dayalı örnekleme tabanlı kestirimin daha kolay şekilde kavranmasını sağlamak amacıyla, bu çalışmada temel Parçacık filtreleme yöntemi esas alınmıştır.

Parçacık filtrelemede, genellikle çözümü hedeflenen problem hakkında varsayım yapılmaz [42]. Problem lineer olmayan durum geçişine ve Gauss dağılımına uymayan proses ve ölçüm gürültülerine sahip dinamik bir sistemle ilişkili olabilir. Bununla birlikte, göz önüne alınması gerekli iki temel husus bulunur: (1) durum uzayının yüksek boyutluluğu, (2) sistemin dinamik ve gözlem modelleri [38],[46]. İlk husus, günümüzün hesaplama kaynaklarıyla büyük ölçüde çözülebilirken, ikinci husus problemin zorluğuna ve yapılan gözlemlerden çıkarılan kullanılabilir bilgiyi de içeren çözüm metodolojisine bağlıdır. Uygun dinamik ve gözlem modelleri kullanıldığında Parçacık filtreleme yöntemi ile birçok problem için kabul edilebilir çözümler geliştirilmesi mümkündür.

2.5. Bayesçi Filtreleme Yöntemleri

Bu çalışmada, Bayesçi kestirime dayalı filtreleme yöntemlerinden Kalman, genişletilmiş Kalman, ilerletilmiş Kalman, Gauss toplamı, ızgara-tabanlı, Markov Zinciri Monte Carlo ve parçacık filtreleme incelenmiştir. Bu yöntemlerin, durum geçişi ve gürültü varsayımları, sundukları çözümün optimal ya da yaklaşık olması ve çözüme yaklaşma biçimlerine ilişkin karşılaştırma Tablo 2.1'de, temel kısıt ve avantajları ise Tablo 2.2'de verilmiştir. Bayesçi filtreleme yöntemlerinin özellikleri ve işleyişleri ile ilgili detaylar ise alt başlıklarda ele alınmıştır.

Tablo 2.1. Bayesçi Kestirim Yöntemleri Karşılaştırması

	Durum Geçişi		**Gürültü**		**Çözüm**		**Yaklaşım**	
	Doğrusal	Doğrusal değil	Normal Dağılıma Uyan	Keyfi	Optimal	Yaklaşık	Parametrik	Örneklemeye Dayalı
KF	+		+		+		+	
EKF		+	+			+	+	
GSF		+		+		+	+	
UKF		+		+		+	+	
GBF		+		+		+		+
MCMC		+		+		+		+
PF		+		+		+		+

Tablo 2.2. Bayesçi Kestirim Yöntemlerinin Kısıt ve Avantajları

	Kısıt	**Avantaj**
KF	Proses ve gürültü doğrusal varsayımı	Optimal çözüm
EKF	Yüksek dereceli proseslere duyarsız olması	Doğrusal olmayan sistemler için makul çözüm
GSF	Belirsizlik yayılırken terim ağırlıklarının sabit tutulması	EKF'ye göre daha yaklaşık çözüm
UKF	Sınırlı sayıdaki nokta ile yaklaşım	KF, EKF ve GSF'ye göre daha iyi çözüm
GBF	Sonlu sayıda durum varsayımı	Sonlu durumlu sistemler için makul çözüm
MCMC	Sonsal olasılığa ne kadar yaklaşıldığının bilinememesi	PF'ye alternatif bir yaklaşım
PF	Gözlem modeline bağımlılık	Gerçek dünya problemleri için makul çözüm

2.5.1. Kalman filtresi

Kalman Filtresi, hedef izleme problemini olasılıksal temelde ele alan ilk yöntemlerden birisidir. Bu yöntemde, dinamik sistemin durum geçişleri lineer ve proses ile ölçümlerdeki belirsizlikler de Gauss gürültüsü olarak varsayılarak izleme problemi aşağıdaki durum uzayı modeli ile ifade edilir. (Denklem 2.27)

$$x_k = F_k x_{k-1} + v_{k-1}, \;\; k > 0 \qquad \textbf{(2.27)}$$

Burada, x_k kestirilecek durum vektörü, k zaman adımı ve F_k da lineer bir durum-geçiş matrisidir. v_k ise sistem gürültüsü olarak tanımlanan normal dağılıma uyan bir gürültü

dizisidir. Ölçüm denklemi için de, H_k ölçüm modeli ve w_k ise ölçüm gürültüsü olarak tanımlanan normal dağılıma uyan bir gürültü dizisidir. (Denklem 2.28)

$$z_k = H_k x_k + w_k, \;\; k > 0 \tag{2.28}$$

Q_{k-1} ve R_k sırasıyla v_{k-1} ve w_k gürültülerinin kovaryans matrisleri ve $\mathcal{N}(\mu; P)$ ortalaması μ, kovaryansı P olan normal dağılım olmak üzere; $p(v) \sim \mathcal{N}(0; Q)$ ve $p(w) \sim \mathcal{N}(0; R)$ olarak ifade edilir.

Filtreleme probleminin olasılıksal gösterimi ise aşağıdaki gibi oluşturulur:

$$\begin{aligned} p(x_{k-1}|z_{1:k-1}) &= \mathcal{N}(E[x_{k-1|k-1}]; P_{k-1|k-1}) \\ p(x_k|z_{1:k-1}) &= \mathcal{N}(E[x_{k|k-1}]; P_{k|k-1}) \\ p(x_k|z_{1:k}) &= \mathcal{N}(E[x_{k|k}]; P_{k|k}) \end{aligned} \tag{2.29}$$

Kalman filtresi, Bayes temelli bir kestirim yöntemi olduğu için sonsal olasılığa Kestirim ve Güncelleme olmak üzere iki adımda öz-yinelemeli şekilde yaklaşır. (Denklem 2.30 ve 2.31)

Kestirim Adımı

$$\begin{aligned} x_{k|k-1} &= F_k \times x_{k-1|k-1} \\ P_{k|k-1} &= F_k \times P_{k-1|k-1} \times F_k^T + Q_{k-1} \end{aligned} \tag{2.30}$$

Güncelleme Adımı

$$\begin{aligned} K_k &= P_{k|k-1} \times H_k^T \times (H_k \times P_{k|k-1} \times H_k^T + R_k)^{-1} \\ x_{k|k} &= x_{k|k-1} + K_k \times (z_k - H_k \times \hat{x}_{k|k-1}) \\ P_{k|k} &= P_{k|k-1} - K_k \times H_k \times P_{k|k-1} \end{aligned} \tag{2.31}$$

Kalman filtreleme yönteminde, durum geçişleri lineer, proses ve ölçüm gürültüleri de Normal dağılıma uyan gürültüler olarak varsayılır. Bu varsayımlarla Kalman filtresi,

Bayes çıkarımına dayalı optimal bir çözüm sunar. Birçok gerçek dünya problemi ve özellikle insan hareketi izleme problemi ise bu varsayımlarla modellenemez. Çünkü, bu problemlerde, durum geçişleri çoğunlukla lineer değildir, gürültü dizileri ise Normal dağılıma uymaz. Bu nedenle, bu problemlerin olasılık tabanlı çözümleri için Kalman filtresi yerine daha gelişmiş yöntemlere ihtiyaç duyulur.

2.5.2. Genişletilmiş Kalman filtresi

Kalman filtresi ile lineer olmayan durum geçişlerine sahip olan dinamik sistemler için uygun çözüm bulunamaması, genişletilmiş Kalman filtresinin ortaya çıkmasını sağlamıştır. Genişletilmiş Kalman filtresi, dinamik sistemin durum geçişlerinin lineer olmaması durumunda kullanılır. Bununla birlikte, proses ve ölçüm gürültüleri Kalman filtresinde olduğu gibi Normal dağılıma uyan gürültüler olarak kabul edilir.

Genişletilmiş Kalman filtresinin sonsal olasılığa yaklaşım adımları aşağıda Denklem (2.32) ve (2.33) ile ifade edildiği gibidir.

Kestirim Adımı

$$\begin{aligned} x_{k|k-1} &= f_k \times x_{k-1|k-1} \\ P_{k|k-1} &= \hat{F}_k \times P_{k-1|k-1} \times \hat{F}_k^T + Q_{k-1} \end{aligned} \tag{2.32}$$

Güncelleme Adımı

$$\begin{aligned} K_k &= P_{k|k-1} \times \hat{H}_k^T \times (\hat{H}_k \times P_{k|k-1} \times \hat{H}_k^T + R_k)^{-1} \\ x_{k|k} &= x_{k|k-1} + K_k \times (z_k - h_k \times \hat{x}_{k|k-1}) \\ P_{k|k} &= P_{k|k-1} - K_k \times \hat{H}_k \times P_{k|k-1} \end{aligned} \tag{2.33}$$

Burada f_k ve h_k sırasıyla lineer olmayan süreç ve ölçüm fonksiyonları, $\hat{F}_k$ ve $\hat{H}_k$ ise bunların Taylor serisine göre açılarak lineerleştirilmiş halleridir.

$$\hat{F}_k = \left.\frac{\partial\, f_k(x)}{\partial x}\right|_{x=x_{k|k-1}} \quad \text{ve} \quad \hat{H}_k = \left.\frac{\partial\, h_k(x)}{\partial x}\right|_{x=x_{k|k-1}} \tag{2.34}$$

Proses ve ölçüm gürültüleri ise Kalman filtreleme yöntemindeki gibidir: Q_{k-1} ve R_k sırasıyla v_{k-1} ve w_k gürültülerinin kovaryans matrisleri. Genişletilmiş Kalman filtreleme yönteminde, lineer olmayan durum geçiş ve ölçüm fonksiyonlarını lineerleştirilerek problem modeli basitleştirilir. Problemin çözümü ise yine standart Kalman filtreleme yöntemine göre yapılır. Genişletilmiş Kalman filtesi bu yaklaşımla, çok sayıda gerçek dünya problemi için çözüm sunabilmektedir. Filtrelemeye ihtiyaç duyulan birçok alanda, bu yöntem günümüzde de hala en yaygın olarak kullanılan yöntem durumundadır. Ancak, dinamik modelde yapılan sadeleştirmeler, hassas durum geçişlerine sahip problemler için uygun çözümler bulmayı engellemektedir. Aynı zamanda normal dağılıma uyan gürültü varsayımı da birçok problem için geçerli değildir.

2.5.3. İlerletilmiş Kalman filtresi

İlerletilmiş Kalman filtreleme yöntemi, yapısında Kalman filtreleme ve belirsizlik yayma dönüşümü kullanıldığı için bu adı almıştır. Belirsizlik yayma dönüşümü, lineer olmayan bir dönüşüme zorlanan rassal bir değişkenin ortalama ve varyans gibi istatistiki özelliklerini hesaplamakta kullanılan bir yöntemdir. Bu yöntemde, ortalama etrafında deterministik olarak seçilen sınırlı sayıda nokta, öz-yinelemeli kestirim ve güncelleme aşamalarında ilerki zaman adımlarına yayılarak gerçek ortalama ve varyansa yaklaşılmaya çalışılır. Bu yöntemin genişletilmiş Kalman filtreleme yöntemine göre iki avantajı mevcuttur. Bunların ilki, sonsal olasılığa genişletilmiş Kalman filtresine göre daha iyi yaklaşması, ikincisi ise genişletilmiş Kalman filtreleme yönteminde hesaplanması gereken kısmi türevlerin bu yöntemde hesaplanmasına gerek olmamasıdır. Bununla birlikte, ilerletilmiş Kalman filtreleme yönteminin her adımında Cholesky ayrıştırması işlemi gereklidir ve dolayısıyla bu yöntemin toplamdaki hesap yükü genişletilmiş Kalman filtesine göre daha fazladır. Ayrıca, dönüşüm varsayımlarından ve normal dağılıma uyan gürültü varsayımından dolayı bazı problemler için uygun çözüm sunmaz.

2.5.4. Gauss toplamı filtresi

Gauss toplamı filtreleme yöntemindeki temel düşünce, sonsal olasılığa, ortalama ve varyans özellikleri ile kolay bir şekilde gösterilebilen normal dağılımların toplamı ile yaklaşmaktır. "Normal dağılımların istatistiki özelliklerini tespit edip, bunların toplamı olarak sonsal olasılığın yaklaşık değerini bulmak, herhangi keyfi bir olasılık yoğunluk fonksiyonunun istatistiki özelliklerini bulmaya göre daha kolay bir şekilde yapılabilir" düşüncesinden hareketle çözüm arayan Gauss toplamı filtresi, yapısında genişletilmiş Kalman filtresini de içerir.

Gauss toplamı filtreleme yönteminde, her zaman adımda toplamdaki terim sayısı üstel olarak artar, bu terimlerin sayısının azaltılması için ağırlıkları düşük olanlar toplamdan çıkarılır ya da birbirine yaklaşık değerleri olan terimler birleştirilir. Gauss toplamı filtresi, güncelleme adımında genişletilmiş Kalman filtresi kullandığı için, bu yöntemin dezavantajlarını da yapısında barındırır. Bununla birlikte, çoklu terim kullanımıyla lineerleştirme düzeyi oldukça aşağıya çekilebilir. Böylece genişletilmiş Kalman filtresine göre sonsal olasılık yoğunluğunu daha iyi ifade edebilir.

2.5.5. Izgara tabanlı yöntemler

Izgara tabanlı yöntemlerde (Grid Based Methods), dinamik sistemin durumlarının sonlu ve sınırlı sayıda olduğu varsayılır. Izgara tabanlı yöntemler sınırlı sayıda durum olduğunda etkin çözüm sunabilirler ancak gerçek dünya problemleri ve özellikle insan hareketi izleme probleminde durum uzayı sınırlı değildir. Bu nedenle, insan hareketi izlemede ızgara tabanlı yöntemlerin kullanımı uygun değildir.

2.5.6. Monte Carlo ve Markov zinciri Monte Carlo

Monte Carlo simülasyonu, analitik olarak hesaplanamayan integrallerin değerine yaklaşık çözüm üretmeyi amaçlayan yöntemlerden oluşur. $f(x)$, ve g(x) x'in herhangi birer fonksiyonu ve $p(x)$ de x'in olasılık yoğunluk fonksiyonu olmak üzere Monte Carlo yöntemleri denklem (2.35) ile gösterilebilen integrallerin değerini bulmaya çalışır.

$$I = \int f(x)dx = \int g(x)p(x)dx \tag{2.35}$$

Monte Carlo yöntemleriyle integral değerine yaklaşılırken, ilgili olasılık yoğunluk fonksiyonundan bağımsız rassal örnekler alınabileceği varsayılır. Güçlü sayılar kanuna (Law of Large Numbers) göre, bilinen bir olasılık dağlımına sahip rassal değişkenler dizisinin ortalaması, denklem (2.36)'da ifade edildiği üzere, olasılık dağlımından alınan örnek sayısı sonsuza giderken, beklenen değerine yaklaşır.

$$\frac{1}{N}\sum_{i=1}^{N} f(x_i) \longrightarrow E[f(x)] = \int_a^b f(x)p(x)dx \tag{2.36}$$

$p(x_{0:k}|z_{1:k})$ x'in sonsal olasılık yoğunluk fonksiyonu ve $(.)$ $x_{0:k}$'nın bir fonksiyonu olmak üzere olmak üzere, $x_{0:k}$'nın beklenen değeri denklem (2.37)'de verildiği gibi olacaktır.

$$E_{p(x_{0:k}|z_{1:k})}[g(x_{0:k})] = \int g(x_{0:k}) \times p(x_{0:k}|z_{1:k}) \tag{2.37}$$

Monte Carlo yöntemleriyle, bu beklenen değere denklem (2.38) ile yaklaşılır:

$$E_{p(x_{0:k}|z_{1:k})}[g(x_{0:k})] = \frac{1}{N}\sum_{i=1}^{N} g\left(x_{0:k}^{(i)}\right) \tag{2.38}$$

Markov Zinciri Monte Carlo yöntemleri, durağan dağılımı hedef dağılıma eşdeğer olan bir hedef dağılımından doğrudan örneklemeye dayanır. MCMC tabanlı yöntemlerde, hedef dağılımdan örneklemeyi sağlayacak yapı Markov Zinciri ile oluşturulurken, dağılımın integral değerine yaklaşmak için Monte Carlo integrasyonu kullanılır. Metropolis-Hastings ve Gibbs yöntemleri en yaygın olarak kullanılan MCMC yöntemleri arasındadır.

BÖLÜM 3. PARÇACIK FİLTRESİ İLE PROBLEM FORMULASYONU

3.1. Parçacık Filtresi

Parçacık filtresi literatürde sıralı Monte Carlo yöntemi olarak da bilinen, dinamik sistemlerin durumlarını Bayes kestirimi temelinde analiz etmeyi sağlayan ve sonsal olasılık yoğunluk fonksiyonunun değerine yaklaşmak önem örnekleme tekniğini öz-yinelemeli şekilde uygulayan yöntemler için kullanılan genel bir isimdir. Parçacık filtesi ile modellenen dinamik bir sistemde, her bir parçacık, dinamik sistemin durumlarının tamamını içeren ve bir çözüm adayı olan rassal değişkenler dizisi olarak yapılandırılır. İncelenecek sisteme ya da çözülecek probleme bağlı olarak belirli sayıda parçacık oluşturulur. Bu parçacıkların her birisi bir ağırlık değerine sahiptir. Bu ağırlık, ilgili parçacığın, hedefteki sonsal olasılık yoğunluk fonksiyonunu ne ölçüde temsil ettiği ile orantılıdır. Başlangıç aşamasında tüm parçacıkların eşit bir ağırlığı olabilir ya da dinamik sisteme ilişkin herhangi bir ölçüm yapılmadan önce sistem hakkında sahip olunan bilgi kullanılarak farklı ağılıklara da sahip olabilir.

Parçacık filtresi işleyişinde, öz-yinelemeli Bayes kestiriminin tahmin ve güncelleme aşamaları her bir zaman adımında birbirini takip edecek şekilde uygulanır. Her yeni zaman adımında, parçacıkların ağırlıkları yeniden hesaplanır ve bu parçacıklar içerisinden belli yöntemlerle problemin çözümü olarak bir kestirim elde edilir. Parçacıklar ağırlıklandırılırken, her zaman adımında sisteme ilişkin olarak yapılan ölçümlerden olabilirlik değeri çıkarılır. Ölçüm ya da gözlem modeli parçacık filtresi ile problem çözümünde çok önemlidir, çünkü ağırlıklar çözümü şekillendirir ve bu ağırlıkların yanlış ya da eksik hesaplanması problemin çözümünden uzaklaşılmasına yol açabilir.

Parçacık filtresi teorik olarak, çoklu hipotezi destekleyebilir. Bunun anlamı, birden fazla modu olan bir sonsal olasılık yoğunluk fonksiyonuna Parçacık filtresi kullanılarak yaklaşılabileceğidir. Ayrıca, Parçacık filtresi dinamik sistemin işleyişi ile ilgili kısıtlayıcı varsayımlarda bulunmaz. Sistemin durum geçişleri lineer olmayabilir, proses ve ölçüm gürültüleri de keyfi bir olasılık dağılımına sahip olabilir. Bu bağlamda, insan hareketi izleme problemi de dahil olmak üzere birçok gerçek-dünya probleminin kabul edilebilir çözümü Parçacık filtresinin doğru kullanımıyla sağlanabilir.

3.2. Parçacık Filtresi ile Problem Formülasyonu

Ayrık zamanlı dinamik bir sistemin mevcut zaman adımındaki durumu hakkında bilgi veren, sonsal olasılık yoğunluk fonksiyonu $p(x_{0:k}|z_{1:k})$, denklem (2.18) ile elde edilmişti. Çok güçlü varsayımların kullanıldığı ve çok özel şartların var olduğu durumlar dışında, bu birleşik olasılık dağılımının değerini analitik olarak hesaplamak mümkün değildir. Gerçek-dünya problemlerinin neredeyse tamamı için varsayım ve kısıtlamaların asgari seviyede olması beklenen bir durum olduğundan, çözüm için yaygın olarak tercih edilen yaklaşım; EKF, UKF, GSF, GBF, MCMC ya da bu çalışmanın temel odak noktası olan parçacık filtreleme gibi optimal olmayan Bayesçi kestirim yöntemlerinden birisini kullanarak $p(x_{0:k}|z_{1:k})$ sonsal olasılığının değerine yaklaşık bir değer elde etmeye çalışmaktır.

Parçacık filtresi, diğer Bayesçi kestirim yöntemlerinde olduğu gibi problemin çözümüne her yeni zaman adımında peşpeşe uygulanan tahmin ve güncelleme işlemleri sonucunda ve bir önceki zaman adımındaki bilgiyi kullanmak suretiyle yaklaşır. Bu süreç, Bölüm 2.4.1 ‘de anlatılan Öz-yinelemeli Bayesçi kestirim temelinde işler. Öz-yinelemeli Bayesçi kestirim aslında, incelenenen ayrık zamanlı dinamik sistemin, herhangi bir zaman adımındaki durumununa ilişkin bilginin geçmiş tüm zaman adımları ve ölçümler kullanılarak belli varsayımlar ışığında nasıl elde edildiğini açıklar. Olasılık teorisine ait Bayes ve zincir kuralları ve özellikle durağan geçiş olasılıklı Markov süreci temelinde yapılandırılan böyle bir dinamik sistem parçacık filtresinin işletilebilmesi için gerekli ön-koşuldur. Bu şekildeki bir yapılandırma ile sistemin durumlarına ilişkin bilgi kaybı olması kaçınılmazdır. Ancak,

parçacık filtresi zaten, analitik olarak çözülemeyen problemler için olasılıksal örneklemeye dayanan makul ölçüdeki yaklaşık bir çözümü elde etmeyi sağlayan bir yöntem olarak öne çıkmaktır.

Parçacık filtresi ile problem formulasyonu için Monte Carlo integrasyon yöntemlerinden birisi olan önem örneklemenin uygulanması gereklidir. Monte Carlo, integrasyonu, analitik olarak hesalanması mümkün olmayan integallerin değerini yaklaşık olarak hesaplamaya yarayan yöntemler topluluğuna verilen genel bir isimdir. Kabul-Red (Accept-Reject), Ters Kümülatif Yoğunluk Fonksiyonu Dönüşümü (Inverse CDF Transform), Kontrol Değişkeni (Control Variable) ve Önem Örnekleme (Importance Sampling) yöntemleri Monte Carlo integrasyon yöntemleri arasında yer alır. Bununla birlikte, önem örnekleme parçacık filtresinin temelinde yer alır ve bu çalışmada detaylı olarak ele alınmıştır. Parçacık filtrelemeyi anlayabilmek için önem örnekleme, öncelikle kavranması gereken bir yöntemdir. Önem örnekleme yöntemi hedef integralin değerine denklem (3.1) 'de verildiği gibi yaklaşır.

$$I = \int f(x) \times p(x)dx = \int f(x) \times \frac{p(x)}{q(x)} \times q(x)dx \qquad \textbf{(3.1)}$$

Burada, $p(x)$ örneklenmesi mümkün olmayan bir olasılık yoğunluk fonksiyonu, $q(x)$ önem fonksiyonu olarak adlandırılan ve $p(x) > 0 \Longrightarrow q(x) > 0, \forall x \in \mathbb{R}^n$ şartını sağlayan örneklenmesi kolay bir olasılık yoğunluk fonksiyondur. Önem örneklemesi yöntemindeki temel fikir, Monte Carlo integrasyon yapısına örneklenmesi kolay olan bir olasılık yoğunluk fonksiyonu ekleyerek, örneklenmesi mümkün olmayan hedef olasılık yoğunluk fonksiyonunun değerini elde etmektir. Bu değerin elde edilmesi için gerekli olan bilgi ise, hedef olasılık yoğunluk fonksiyonu ile önem fonksiyonu arasındaki orantıdır. Bu orantı, önem ağırlığı olarak adlandırılır ve değeri parçacık filtreleme formulasyonunda yapılan bazı kabuller ile elde edilir.

Önem örnekleme yöntemi ile denklem (3.1) 'de verilen integralin değerine aşağıdaki şekilde yaklaşılır. (Denklem 3.2)

$$I \cong \frac{1}{N}\sum_{i=1}^{N} f(x_i) \times w(x_i)\,, \qquad w(x_i) = \frac{p(x_i)}{q(x_i)} \tag{3.2}$$

Parçacık filtrelemede, hedef olasılık yoğunluk fonksiyonuna denklem (3.3) 'deki gibi yaklaşılır.

$$p(x_{0:k}|z_{1:k}) \cong \sum_{i=1}^{N} \widetilde{w}_k^{(i)} \times \delta\left(x_{0:k} - x_{0:k}^{(i)}\right), \quad \widetilde{w}_k^{(i)} = \frac{w_k^{(i)}}{\sum_{j=1}^{N} w_k^{(j)}} \tag{3.3}$$

Burada, $\delta(.)$ Dirac-Delta fonksiyonu, $x_{0:k}^{(i)}$, her bir zaman adımındaki $i.$ parçacığın değeri, N parçacık sayısı ve $\widetilde{w}_k^{(i)}$ da k zaman adımındaki $i.$ parçacığın normalleştirilmiş ağırlığıdır.

Optimal öz-yinelemeli Bayes filtresi önceki bölümde denklem (2.26) ile verilmişti. Parçacık filtresi temelli problem formulasyonu süreci, bu denklem ile $\{\, p(x_{0:k}|z_{1:k}) \propto p(z_k|x_k) \times p(x_k|x_{k-1}) \times p(x_{0:k-1}|z_{1:k-1}) \,\}$ ile başlatılır.

İlk aşamada, bir önem fonksiyonu belirlemek gerekir. Bu fonksiyon denklem (3.4) 'deki gibi kabul edildiği takdirde, bir miktar bilgi kaybıyla normalleştirilmiş önem ağırlıkları denklem (3.5) ile hesaplanabilir.

$$q(x_{0:k}|z_{1:k}) \triangleq q(x_{0:k}|x_{0:k-1}, z_{1:k-1}) \times q(x_{0:k-1}|z_{1:k-1}) \tag{3.4}$$

$$\widetilde{w}_k^{(i)} = \frac{p(x_{0:k}|z_{1:k})}{q(x_{0:k}|z_{1:k})} = \frac{p(z_k|x_k) \times p(x_k|x_{k-1}) \times p(x_{0:k-1}|z_{1:k-1})}{q(x_{0:k}|x_{0:k-1}, z_{1:k-1}) \times q(x_{0:k-1}|z_{1:k-1})} \tag{3.5}$$

Burada, denklem (3.5)'in denklemin ardışık zaman adımlarına ait önem ağırlıklarını içerdiği ve $\frac{p(x_{0:k-1}|z_{1:k-1})}{q(x_{0:k-1}|z_{1:k-1})}$ ifadesinin bir önceki zaman adımına ait sonsal olasılık yoğunluk fonksiyonu olduğu açıkça görülebilir. Böylece, öz-yinelemeli bir yapı oluşturulmuş olur. (Denklem 3.6)

$$\widetilde{w}_k^{(i)} \propto \widetilde{w}_{k-1}^{(i)} \times \frac{p(z_k|x_k) \times p(x_k|x_{k-1})}{q(x_{0:k}|x_{0:k-1}, z_{1:k-1})} \tag{3.6}$$

Parçacık filtreleme literatürde çeşitli yöntemlere dayalı olarak gerçekleştirilmiştir. Bu yöntemlerin ayırıcılığını belirleyen faktör kullandıkları varsayımlardır. Bunlar arasındaki en basit fakat özellikle gerçek zaman kısıtı olan uygulamalar için makul derecede etkin olan yöntemlerden birisi Bootstrap parçacık filtresi ya da Condensation algoritmasıdır [31]. Bu yöntemde, denklem (3.6)'daki $q(x_{0:k}|x_{0:k-1}, z_{1:k-1})$ ifadesi ile ilgili olarak ikinci bir kabul yapılır. Bu ifade, hedefteki olasılık yoğunluk fonksiyonun ardışık durumları arasındaki geçiş olasılığı olarak seçilir. (Denklem 3.7)

$$q(x_{0:k}|x_{0:k-1}, z_{1:k-1}) \triangleq p(x_k|x_{k-1}) \tag{3.7}$$

Böylece, öz-yinelemeli Bayesçi kestirim tabanlı parçacık filtresi ile problem formulasyonu aşağıdaki gibi elde edilmiş olur. (Denklem 3.8)

$$\tilde{w}_k^{(i)} \propto \tilde{w}_{k-1}^{(i)} \times p(z_k|x_k) \tag{3.8}$$

Denklem (3.8) incelendiğinde, her bir adımdaki önem ağırlıklarının bir önceki zaman adımındaki ağırlıklara ve $p(z_k|x_k)$ olarak ifade edilen mevcut zaman adımındaki ölçüm olabilirliğine bağlı olduğu açıkça görülebilir. Olabilirlik her bir adımda eldeki sensörler yoluyla elde edilen ölçümlerden çıkarılan bilginin ilgili zaman adımındaki sistem dinamikleri ile ne kadar uyumlu olduğunu gösteren bir oransal değerdir. Bu oransal değer arttıkça incelenen problem için kabul edilebilir çözüme o ölçüde yaklaşılır.

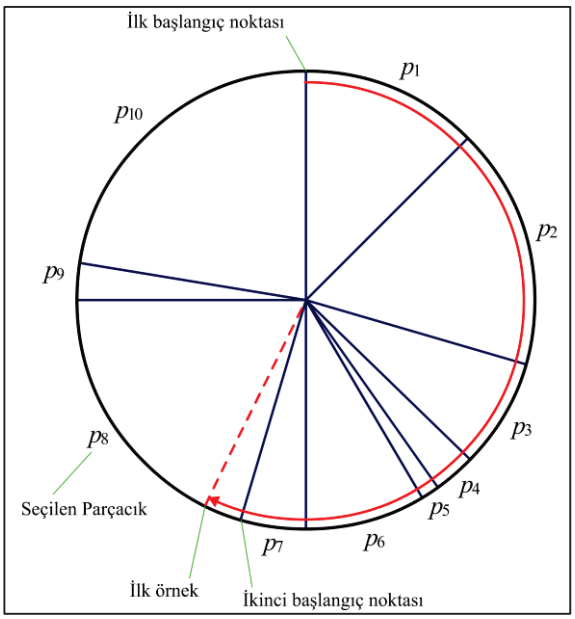
3.3. Yeniden Örnekleme Teknikleri

Parçacık filtrelemenin opsiyonel bir adımı olan yeniden örnekleme, ağırlık dejenerasyonuna karşı, ardışık zaman adımları boyunca ağırlığı büyük olan parçacıkların sayısının artırılıp düşük olanların sayısının azaltılması ilkesine dayanır. Yeniden örnekleme adımından sonra, yüksek ağırlıklı parçacıkların toplam parçacıklar içindeki oranı artarken, bir sonraki zaman adımında kullanılacak olan parçacıkların ağırlıkları eşitlenir. Böylece her yeni zaman adımında daha iyi bir çözüm elde edilir. Parçacık filtreleme literatüründe yeniden örnekleme için en yaygın olarak kullanılan; çok değerli, artık terimli, tabakalı ve sistematik yeniden örnekleme olmak üzere 4

temel teknik bulunur. Bu tekniklerin karşılaştırmalı başarım performansları ile ilgili detaylı bilgi [55] ve [56]'da mevcuttur.

3.3.1. Çok değerli yeniden örnekleme

Çok değerli yeniden örnekleme yönteminde, mevcut parçacık kümesindeki tüm parçacıklar ağırlıklarına oranla çark üzerinde açısal bir bölge işgal edecek şekilde yerleştirilir. Tüm ağırlıkların toplamı 1 olacağından 0.5 ağırlığına sahip bir parçacık çark üzerinde 180°'lik bir açıya ship olacaktır. Yerleştirme işleminden sonra, (0,1) aralığında N adet bağımsız düzgün dağılmış rassal örnek üretilir. İlk parçacığın bölge başlangıcından başlamak üzere, her bir rassal örnek parçacıkların kümülatif ağırlığıyla karşılaştırılır. Rassal örneğin işgal ettiği açısal bölgenin karşılık geldiği parçacık yeni parçacık kümesi için seçilir. Her yeni döngü, bir önceki adımda seçilen parçacığın açısal bölgesi bağlangıç kabul edilerek işletilir. Bu işlem N kez (parçacık sayısı kadar) tekrar edilerek yeni parçacık kümesi oluşturulur. Çok değerli yeniden örnekleme yönteminin 10 örnek için grafiksel bir gösterimi Şekil 3.1'de verilmiştir.



Şekil 3.1. Çok Değerli Yeniden Örnekleme

Bu basit algoritmadan da anlaşılacağı gibi, yüksek ağırlığa sahip parçacıklar düşük ağırlığa sahip olanlara göre daha yüksek seçilme oranına sahip olacaktır. Yeni parçacık

kümesinin oluşturulma süreci tamamlandığında, her bir parçacığın ağırlığı $1/N$ değerine eşitlenir. Böylece, tüm parçacık kümesi içinde yüksek ağırlığa sahip parçacıklar daha fazla yer alırken, her yeni zaman adımında daha iyi sonuç alma ihtimali yükselir. Burada ayrıca, düşük ağırlığa sahip parçacıkların da çözüm olma ihtimali küçük de olsa bulunduğundan bu eşit ağırlık atama işlemiyle bu parçacıkların tamamıyla ortadan kalkmasının önüne geçilmiş olur.

3.3.2. Artık terimli yeniden örnekleme

Artık terimli yeniden örnekleme yöntemi, çok değerli yeniden örnekleme yöntemiyle benzerlikler gösterir. Bu yöntemde, parçacık ağırlıkları oransal olarak sıralandıkdan sonra, her parçacığa oransal ağırlığının tam kısmı kadar seçilme şansı verilir. Parçacık ağırlıklarının tam olmayan kısımlarının örneklemesi ise çok değerli yeniden örnekleme yöntemine uygun olarak yapılır.

3.3.3. Tabakalı yeniden örnekleme

Tabakalı yeniden örnekleme yönteminde, $(0,1]$ aralığı N eşit boyutlu aralığa bölünür. Her bir aralık için bir düzgün dağılama uygun rassal bir sayı üretilir. İlk yapılacak seçim yine tamamen rassal olarak belirlendikten sonra, her bir aralık için üretilen rassal sayıya karşılık gelen parçacık yeni parçacık kümesi için seçilir.

3.3.4. Sistematik yeniden örnekleme

Sistematik yeniden örnekleme yöntemi, aslında tabakalı yeniden örnekleme yöntemiyle oldukça benzerlik gösterir. Bu yöntemde de $(0,1]$ aralığı N eşit boyutlu aralığa bölünür ancak her bir aralık için ayrı bir rassal sayı üretilmesi yerine tüm aralıklar için tek bir tane rassal sayı üretilir. Üretilen rassal sayıya karşılık gelen her parçacık yeni parçacık kümesine dahil edilir.

3.4. Parçacık Filtresi ile Örnek Problem Çözümü

3.4.1. Problemin yapılandırılması

Bu bölümde, parçacık filtreleme kapsamı içinde örnek bir problemin formulasyonu çözümüyle birlikte verilecektir. Burada, birbirinden farklı yarıçaplara sahip ve tekdüze bir arkaplan üzerinde belli bir kurala göre hareket etmeyen (düzensiz hareket eden) üç topun pozisyonunun izlenmesi hedeflenmiştir. Problem bundan sonra M-BT olarak adlandırılacaktır. Böyle bir problem seçmekteki temel amaç, parçacık filtreleme algoritmasının işleyişinin anlaşılmasını kolaylaştırmaktır. M-BT probleminin çözümü parçacık filtresi SIR algoritmasının gerçeklenmesiyle sağlanmıştır.

Parçacık filtrelemede, öncelikle istenen sistem parametrelerini içeren problem-yönelimli olarak oluşturulmuş parçacık yapısının tanımlanması gereklidir. M-BT problemi için, her bir çözüm adayını temsil eden $M \times N \times O$ boyutlarına sahip bir matris oluşturulmuştur. Böyle bir parçacık gösteriminde, M durum vektörünün boyutunu, N parçacık sayısını ve O da izlenecek nesnelerin sayısını göstermektedir.

M-BT problemi için Denklem (3.9) ile verilen durum vektöründe p_x, p_y herhangi bir topun yatay ve dikey yöndeki konumlarını, ve v_x, v_y de bu yönlerdeki hızlarını temsil etmektedir.

$$X = [p_x \quad p_y \quad v_x \quad v_y]^T \tag{3.9}$$

Parçacık filtrelemede, hedef probleme uyan dinamik ve gözlem modelleri oluşturmak gereklidir. Parçacıkların zaman içindeki gelişimi dinamik modelle açıklanır. Diğer taraftan gözlem modeli de, eldeki sensörler kullanılmak suretiyle toplanan verilerden çıkarılan referans bilgiyi modellemek için kullanılır. Birçok gerecek-dünya probleminde olduğu gibi, M-BT probleminde de süreç ve ölçümlerdeki belirsizliklerden dolayı her iki modele gürültü eklenmiştir. Lineer olmayan hedef izleme problemleri için sabit hız, sabit ivme, Brownian ya da çok daha gelişmiş çeşitli hareket modelleri uygulanabilir [54]. M-BT problemi için seçilen ve denklem (3.10)

ile verilen dinamik modelde Gauss gürültülü sabit hız hareket modeli analaşılabilir olması amacıyla kullanılmıştır.

$$\begin{bmatrix} p_x^{(k)} \\ p_y^{(k)} \\ v_x^{(k)} \\ v_y^{(k)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} p_x^{(k-1)} \\ p_y^{(k-1)} \\ v_x^{(k-1)} \\ v_y^{(k-1)} \end{bmatrix} + \mathcal{N}(0; \sigma_{x,y}) \quad \textbf{(3.10)}$$

Burada, k zaman adımı, $\mathcal{N}(0; \sigma_{x,y})$ sıfır ortalama değerli ve yatay ve dikey yönde sırasıyla σ_x ve σ_y varyanslara sahip normal dağılıma uyan proses gürültüsüdür. M-BT probleminde, her bir topun her bir adımdaki düzlemsel konumu bu dinamik model kullanılarak elde edilir. Örneğin, k zaman adımında bir topun dikey konumu o topun $k-1$ (bir önceki) zaman dımındaki dikey konumuna proses gürültüsü eklenmek suretiyle hesaplanır. Parçacıkların ilk konumları, ilk zaman adımından önce belirlenir ve her bir zaman adımında herhangi bir ölçüm yapılmadan, bu dinamik hareket modeli kullanılarak konumları tahmin edilir.

M-BT probleminde, her bir top için tanımlanmış parçacık kümesinin içindeki tüm parçacıklar için arka plan sahnesinin merkez koordinatları farklı gürültü varyansları kullanılarak belirlenmiştir.

M-BT örneğinde, üç topun serbest hareketini içeren bir video gözlem kaynağı olarak kullanılmıştır. Videonun her bir karesi bir zaman adımı olarak kabul edilmiştir. Her bir kare için topların düzlemsel konumları temel görüntü işleme teknikleriyle belirlenmiştir. Bununla birlikte, sensör bilgisinin belirsizlik içerdiği varsayımına bağlı olarak tespit edilen konumlara normal dağılıma uyan ölçüm gürültüsü eklenmiş ve gözlem modeli denklem (3.11)'de görüldüğü gibi oluşturulmuştur.

$$\begin{bmatrix} p_x^{(k)} \\ p_y^{(k)} \end{bmatrix} = \begin{bmatrix} c_x^{(k)} \\ c_y^{(k)} \end{bmatrix} + \mathcal{N}(0; \rho) \quad \textbf{(3.11)}$$

Burada, $c_x^{(k)}$ ve $c_y^{(k)}$, k zaman adımında tek bir topun yatay ve dikeydeki konumunu göstermektedir. Bu konum bilgileri Matlab IPT'a ait basit eşikleme (*thresholding*) ve

RegionProps:Centroid komutları kullanılarak elde edilmiştir. Topların birbirinden ayrılması işlemi, her bir top için *RegionProps:Area* komutuyla elde edilen alan değerlerinin sıralanması ve eşlenmesiyle sağlanmıştır. $\mathcal{N}(0;\rho)$ burada sıfır ortalama değerli ve ρ varyanslı Gauss ölçüm gürültüsüdür. M-BT örneğinde, topların her bir zaman adımındaki düzlemsel konumları bu gözlem modeli kullanılarak elde edilmiştir.

Parçacık filtrelemede, gözlem olabilirliğini hesaplamak için bir benzerlik ölçüsü ya da performans metriği bulmak gereklidir. Video tabanlı hedef izlemede, izlenen hedef yörüngesi ile referans veri yörüngesinin karşılaştırılması bir benzerlik ölçüsü olarak kabul edilmiştir [53]. M-BT örneğinde de, her bir zaman adımında gözlem modeli ile elde edilen top konumları ile dinamik model yoluyla üretilen parçacıkların konumları arasındaki öklid uzaklığı benzerlik ölçüsü olarak alınmıştır. Bu benzerlik ölçüsü denklem (3.12) ile verilmiştir.

$$L^{(k)} = \sqrt{\left(p_{x_i}^{(k)} - c_x^{(k)}\right)^2 + \left(p_{y_i}^{(k)} - c_y^{(k)}\right)^2}, \;\; i = 1..N \qquad \textbf{(3.12)}$$

Burada, $L^{(k)}$, k zaman adımındaki uzaklık matrisi, $p_{x_i}^{(k)}$ ve $p_{y_i}^{(k)}$ de $i.$ parçacığın yatay ve dikey konumları, N ise parçacık sayısıdır. $L^{(k)}$ değerinin hesaplanması, tüm toplar için tekrar edilen bir işlemdir, bununla birlikte notasyon kolaylığı için burada nesne indeksi düşürülmüştür. Hesaplanan uzaklıklar uğraşılan probleme bağlı olarak birbirinden oldukça büyük farklılıklar gösterebilir. Bu nedenle, hesaplanan ağırlıklar bir normalizasyondan geçirilir.

M-BT örneğinde, gözlem olabilirlik değerleri ya da başka bir deyişle parçacık ağırlıkları, top-parçacık uzaklığı ile parçacık ağırlığı ters orantılı olacak şekilde uzaklık matrisinden elde edilmiştir. Olabilirlik değerleri, daha yakın parçacıklar için daha büyük, daha uzak parçacıklar için daha düşük olacak şekilde denklem (3.13) ile hesaplanmıştır.

$$Lh^{(k)} = e^{\frac{-\tilde{L}_i^{(k)}}{2\rho}}, \;\; i = 1..N \qquad \textbf{(3.13)}$$

Burada, $Lh^{(k)}, k$ zaman adımındaki parçacık ağırlıkları matrisi, $\tilde{L}_i^{(k)}$, i. parçacığın normalleştirilmiş uzaklık değeri, ρ de ölçüm gürültüsü varyansıdır. M-BT örneğinde, ölçüm gürültüsünün sonuca olan etkisini göstermek amacıyla her bir top için farklı ρ değerleri seçilmiştir.

Parçacık filtrelemede, parçacık ağırlıkları hesaplandıktan sonra, sonsal olasılık yoğunluk fonksiyonunun değeri, bu ağırlıkların bir önceki zaman adımındaki ağırlıklara olan oranına bağlı olarak elde edilir. M-BT örneği için öngörülen çözüm, her bir zaman adımında, her bir top için düzlemsel konumların kestirimidir. Konum bilgileri taşıyan parçacıkların ortalaması, ağırlıklı ortalaması ya da en yüksek ağırlığa sahip parçacık çözüm olarak kabul edilebilir. M-BT örneğinde bunlardan ilki seçilmiştir.

Parçacık filtreleme algoritmasının aşamaları önceden tanımlı bir süre sonuna kadar ya da belli bir eşik değerine ulaşılıncaya kadar tekrar edilir. M-BT örneğinde izleme 120 zaman adımı için yapılmış ve bu şarta bağlı olarak sonuçlar alınmıştır.

Parçacık filtreleme algoritmasının, son adımı yeniden örnekleme aşaması olarak adlandırılır. Bölüm 3.3'te işaret edildiği gibi, yeniden örnekleme parçacıkların zamana dejenerasyona uğrayarak etkisiz hale gelmesini önlemek amacıyla uygulanır. Bu aşamada, etkin parçacık sayısı N_{eff}'in dejenerasyon katsayısı N_{th}'den daha küçük olması durumunun gerçekleşmesine bağlı olarak, daha yüksek ağırlıklı parçacıklar daha fazla sayıda örneklenmek suretiyle önemsiz parçacıklar devre dışı bırakılır. $N_{eff} < N_{th}$ kriteri sağlandığında, bazı teknikler kullanılarak mevcut parçacık kümesinden, sonraki adımda eşit ağırlığa sahip olacak şekilde yeni bir parçacık kümesi ağırlıkları oranınca yeniden örneklenir. M-BT örneği için belirlenmiş olan ve sırasıyla (3.14) ve (3.15) denklemleriyle verilen N_{eff} ve N_{th} değerleri parçacık filtresi literatüründe yaygın olarak kullanılan değerlerdir.

$$N_{eff}^{(k)} = \frac{1}{\sum_{i=1}^{N} (Lh_i^{(k)})^2}, \quad i = 1..N \tag{3.14}$$

$$N_{th} = {}^{2}/_{3}\, N \tag{3.15}$$

Burada, N parçacık sayısı, N_{th} dejenerasyon katsayısı, $N_{eff}^{(k)}$, k zaman adımındaki etkin parçacık sayısı ve $Lh_i^{(k)}$ da $i.$ parçacığın ağırlığıdır.

M-BT örneğindeki Parçacık filtresi gerçeklemesinde, konunun anlaşılmasını kolaylaştırmak amacıyla, diğerleri arasındaki en basit yeniden örnekleme yöntemi olan çok değerli örnekleme yöntemi kullanılmıştır. Parçacık filtresinin bir döngüsü yeniden örnekleme adımıyla tamamlanmış olur. Algoritma yeni adımda aynı işlemler tekrar edilerek işletilmeye devam edilir.

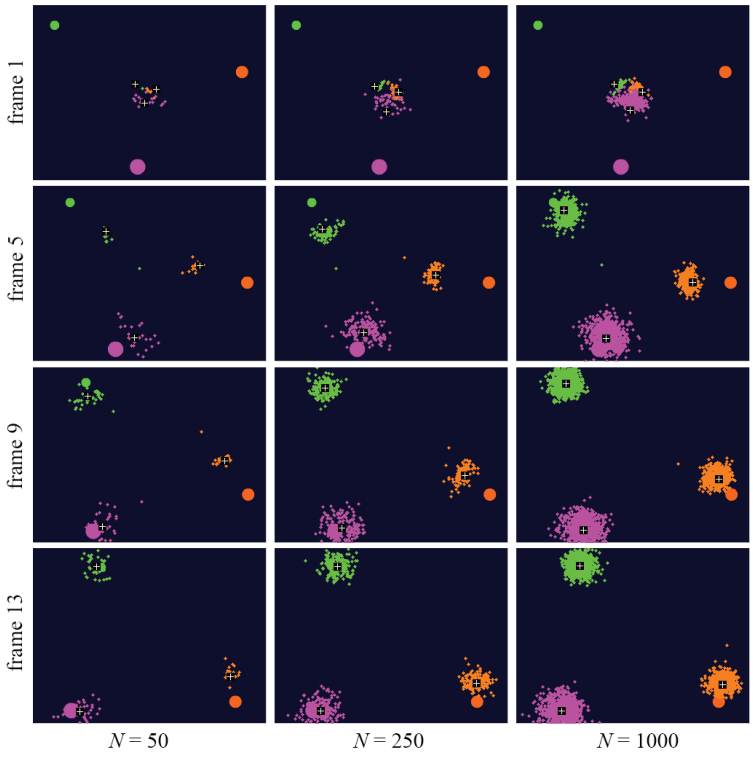
3.4.2. Sonuçlar ve değerlendirme

M-BT probleminin çözümü Parçacık filtresi SIR algoritması kullanılarak serbest harekete eden topların bulunduğu 120 karelik 320×240 piksel çözünürlüğe sahip videoda gerçeklenmiştir. Topların düzlemsel konumları, her sıralı çiftin sırasıyla yeşil, turuncu ve eflatun renkli topların yatay ve dikey konumlarına karşılık geldiği $[30 \quad 50 \quad 144 \quad 222 \quad 287 \quad 92]^T$ vektörü ile başlatılmıştır.

Her biri farklı bir topa ait olan üç parçacık kümesi için parçacıkların ilk düzlemsel konumları, tüm çerçevelerin orta noktası olan $[160 \quad 120]^T$ konumu etrafında rasgele üretilmiştir. Her bir top için yatay ve dikey yönlerdeki gürültü varyansları farklı değerler olarak seçilmiştir. Böyle bir seçim, farklı parametrelere bağlı olan filtreleme performanslarının gözlenmesini sağlamak için tercih edilmiştir. Her bir top için $[\sigma_{p_x} \quad \sigma_{p_y} \quad \sigma_{v_x} \quad \sigma_{v_y}]^T$ şeklinde bir vektör tasarlanarak yeşil, turuncu ve eflatun toplar için $[80 \quad 60 \quad 40 \quad 30]^T$, $[40 \quad 30 \quad 20 \quad 15]^T$, $[160 \quad 120 \quad 80 \quad 60]^T$ değerleri kullanılmıştır.

M-BT örneğinde, ölçüm gürültü varyansaları da her bir top için farklı olacak şekilde seçilmiştir. Ölçüm gürültüsü vektörü toplar için, $[\rho_Y \quad \rho_T \quad \rho_E]^T = [200 \quad 300 \quad 400]^T$ olacak şekilde oluşturulmuştur. M-BT örneğinin çözüm gerçeklemesi sürecinde, ilk düzlemsel konumlar, proses ve gözlem gürültüleri belirlenerek değişen parçacık sayılı bir dizi denemeler yapılmıştır. Buunla birlikte, parçacık sayısı değişiminin sonuca etkisini ayırt ettiği düşünüldüğünden, bu çalışmada

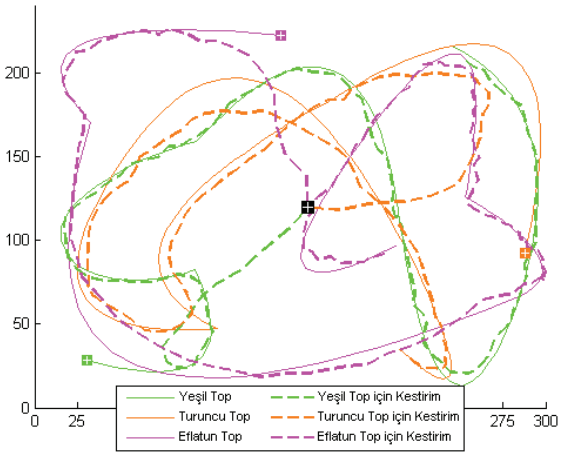
yalnızca, parçacık sayısının 50, 250 ve 1000 olduğu denemelerin grafiksel sonuçları verilmiştir. $N = \{50,250,1000\}$ için, parçacık hareketlerinin 1, 5, 9 ve 13 nolu karelerdeki görüntüleri Şekil 3.2'de verilmiştir.



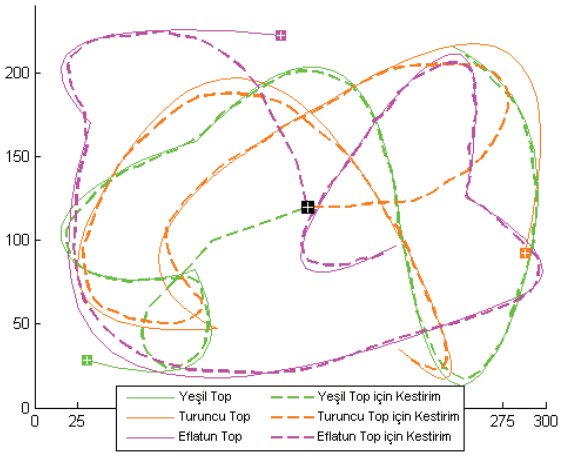
Şekil 3.2. Çoklu Top İzleme Simülasyon Sonuçları

Kareler ilerledikçe tüm parçacıkların toplara daha çok yaklaştığı Şekil 3.2'den açıkça görülmektedir. Bununla birlikte, farklı proses ve ölçüm gürültüsü seçimlerinden dolayı, her bir parçacık kümesi farklı hızlarla yaklaşmaktadır. Tüm filtreleme problemleri için başlangıç değerleri ve gürültü parametreleri ayırt edici performans ölçüleri olarak düşünülür. Bu nedenle, çözümü elde edilmek istenen probleme bağlı olarak bu değer ve parametrelerin kesinlikle özenle seçilmesi gereklidir.

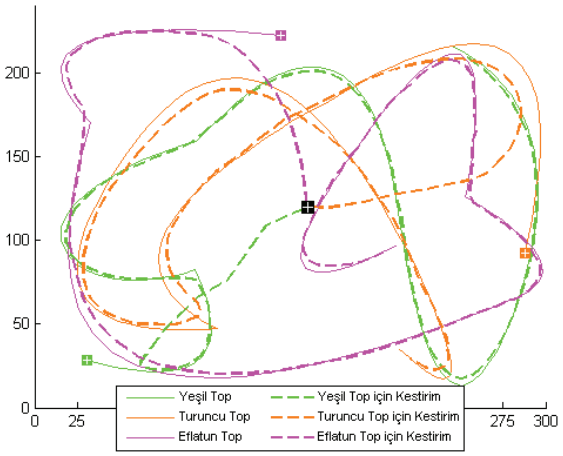
$N = \{50,250,1000\}$ parçacık sayıları için, topların yörüngeleri ve ilgili parçacık filtresi kestirimlerinin yörüngeleri sırasıyla Şekil 3.3, 3.4 ve 3.5 de verilmiştir. Bu şekiller birlikte incelendiğinde, parçacık sayısı 50'den 1000'e doğru artarken, parçacık filtresi kestiriminin gerçek top konumları için giderek daha doğru sonuçlar ürettiği ve daha düzgün yörünge çizdiği görülecektir. Diğer taraftan, gerçek konumlara olan yaklaşım oranları arasındaki fark da ortadadır.



Şekil 3.3. N=50 Parçacık için PF Kestirim Yörüngesi

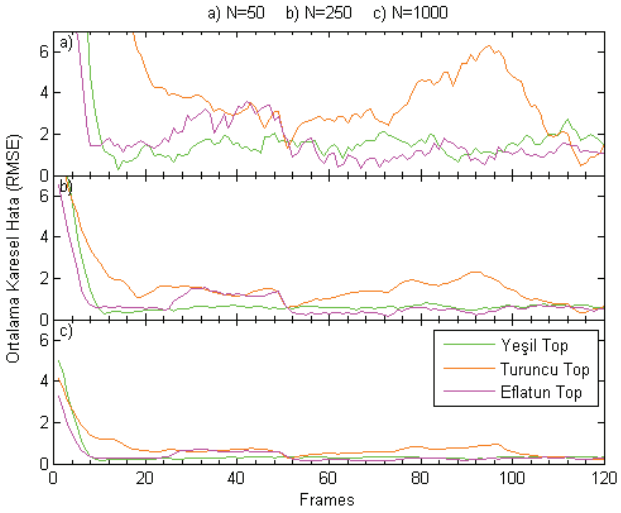


Şekil 3.4. N=250 Parçacık için PF Kestirim Yörüngesi



Şekil 3.5. N=1000 Parçacık için PF Kestirim Yörüngesi

$N = \{50,250,1000\}$ parçacık sayıları için, 120 karedeki ortalama karesel hata değerleri Şekil 3.6'da verilmiştir. Şekil 3.6 incelendiğinde 50 parçacığın iyi bir çözüm üretmediği anlaşılmaktadır. Parçacık sayısı 250'ye çıktığında ortalama karesel hata değerlerinin makul seviyelere düştüğü ancak kestirimin kararlı sonuçlar oluşturmadığı söylenebilir. 1000 parçacık için ise kestirim performansının oldukça iyi aynı zamanda kararlı olduğu gözlenebilir. Çoklu top izleme problemi, gözlem modeline ihtiyaç duyulmayan örnek bir problem olduğu için parçacık sayısı artışının getireceği hesaplama yükü de ihmal edilebilir seviyededir. Bundan dolayı, bu problemin etkin çözümü için 1000 civarında parçacık kullanılabilir.



Şekil 3.6. PF Kestirimleri için Ortalama Karesel Hata Değerleri

Parçacık filtresi kestirimin 120 karedeki ortalama karesel hata değerleri daha detaylı bir şekli ortalamalarla birlikte Tablo 3.1'de verilmişir. Tablo 3.1'deki sonuçlar incelendiğinde, parçacık sayısı arttıkça simülasyon süresinin de arttığı bununla birlikte ortalama karesel hatanın azaldığı görülecektir. *H* denemesi M-BT problemi için kabul edilebilir bir çözüm olarak düşünülebilir. Bu denemede, tüm ortalama karesel hata değerleri 1'in altında iken simülasyon süresi de makul bir değerdedir. Tablo 3.1'deki sonuçlarda ayrıca, *A* denemesi hariç, diğer tüm denemelerde eflatun renkli topa ait ortalama karesel hatanın en düşük olduğu görülmektedir. Sonuç olarak, çoklu top izleme probleminin parçacık filtresi ile gerçeklemesinde, eflatun top için seçilen başlangıç konum değerleri ile gürültü parametrelerin, en uygun olan değerler olduğunu söylemek mümkündür.

Tablo 3.1. Çeşitli Denemeler için Simülasyon Sonuçları

Durum	**Parçacık Sayısı**	**Ortalama Karesel Hata**			**Süre (sn)**
		Yeşil	Turuncu	Eflatun	
A	5	23.21	26.59	23.81	2.767
B	10	10.43	14.71	9.19	2.820
C	25	3.6	6.43	3.13	2.842
D	50	2.31	4.61	1.94	2.900
E	100	1.46	2.87	1.3	3.002
F	250	0.89	1.66	0.81	3.332
G	500	0.61	1.16	0.58	3.811
H	750	0.51	0.86	0.45	4.433
I	1000	0.43	0.75	0.39	4.835
J	5000	0.19	0.32	0.17	13.496
K	*10000*	*0.13*	*0.23*	*0.12*	*24.523*

BÖLÜM 4. VERİ BİRLEŞTİRMEYE DAYALI PARÇACIK FİLTRELEME İLE GERÇEK ZAMANLI NESNE TAKİBİ

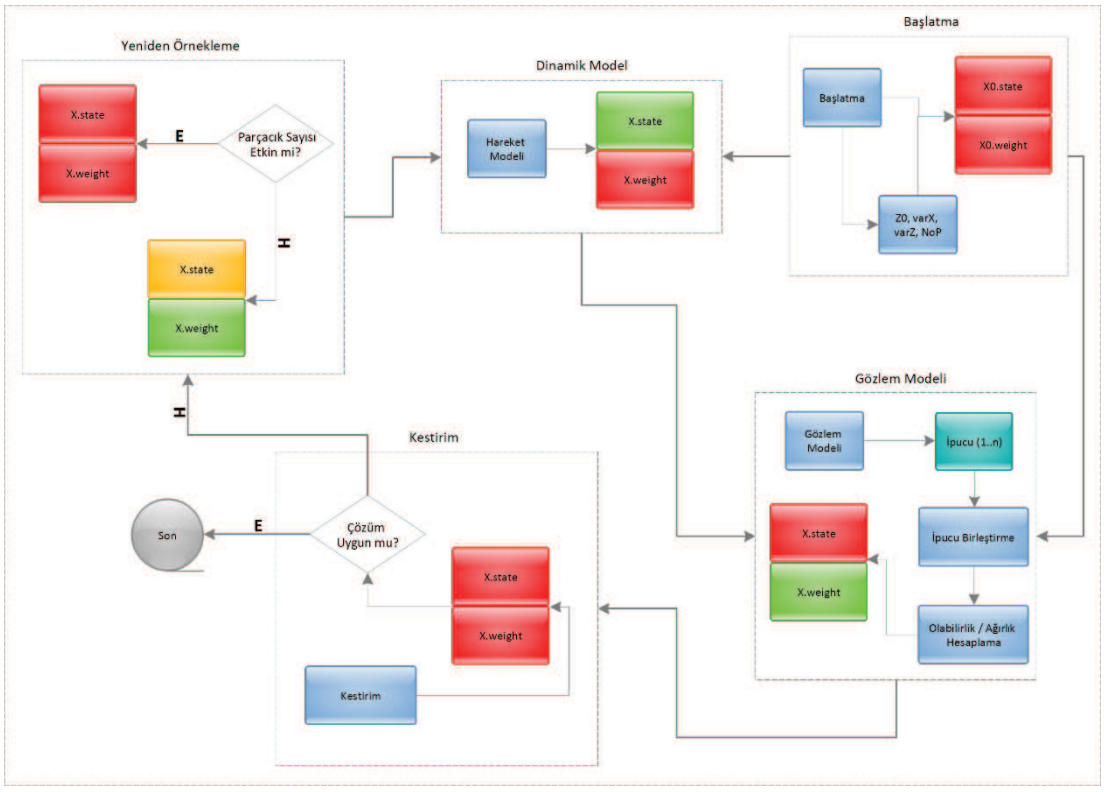
4.1. Problemin Tanımı ve Veri Akışı

Bu bölümde, bu tez çalışmasının temel konusu olan Parçacık filtreleme, farklı sensörlerden gelen verilerden elde edilen bilgilerin birleştirilmesi fikrine (SF-PF) dayalı olarak ele alınmış ve sahnede hareket eden deneğin belirli bir ekleminin (sağ el) gerçek zamanlı olarak takip edilmesi probleminin çözümü sağlanmıştır.

Ele alınan spesifik problem ise Microsoft firmasının bir ürünü olan Kinect cihazının gerçek zamanlı olarak yaptığı hareket izlemenin SF-PF kullanımıyla iyileştirilmesidir. Bu amaçla, öncelikli olarak Kinect cihazının iç-tanımlı hareket izleme algoritmaları ile eklem noktalarını bulma süreci uygulanmıştır. Kinect cihazının, çakışma durumlarında deneğe ait eklem noktalarını büyük hata oranlarıyla bulduğu tespitinden yola çıkılarak, bunun Parçacık filtreleme teknikleriyle iyileştirilebileceği düşünülmüştür. Genel olarak filtrelemenin uygulandığı durumlarda, özel olarak da Parçacık filtrelemede, sensörden gelen verinin gürültülü olduğu varsayılır. Aksi takdirde, yani sensörlerin istenen hassasiyette veri ürettiği durumlarda zaten filtrelemeye ihtiyaç kalmaz. Burada ele alınan problemde de, Kinect cihazı bir sensör olarak düşünülmüş ve ürettiği verilerin gürültülü olduğu kabul edilmiştir.

SF-PF yönteminde birden fazla sensör ya da veri kaynağı olabilmekte ve bunların ürettiği verilerle daha iyi bir sonuç alınacağı öngörülmektedir. Parçacık filtrelemenin önceki bölümlerde anlatılan işleyişinden hareketle, uygulamada bir dinamik model ve gözlem modeli oluşturularak işletilir. Probleme uygun bir parçacık yapısı oluşturularak bir çözüm adayı olarak yapılandırılır. Bu problemde çözüm, takip edilen deneğin sağ el eklem noktasına ait $2B$ eklem koordinatlarının kabul edilebilir bir hata

toleransı ile gerçek zamanlı olarak tespit edilmesidir. Bu bağlamda, her biri bir çözüm adayı olan parçacıklar yatay ve dikey pozisyon bilgisini ve bir ağırlık alanını içerecek şekilde yapılandırılmıştır. Dinamik modelde parçacıklar, izlenen deneğin hareketine uygun olacak şekilde, her bir zaman adımda hareket ettirilir ve (önceki zaman adımında edinilen) eldeki bilgiye göre her birinin ağırlığı hesaplanır. Gözlem modelinde ise sensörlerden gelen veriler kullanılarak parçacıkların ağırlıkları güncellenir. Parçacık filtrelemenin son ve opsiyonel adımı olan yeniden örnekleme adımında ise ağırlık dejenerasyonuna karşı ağırlığı büyük olan parçacıkların sayısı artırılıp düşük olanların sayısı azaltılarak ağırlıklar tekrar eşitlenir. Böylece her yeni zaman adımında daha iyi bir çözüm elde edilir. Probleme ait veri akışı aşağıdaki şekilde verilmiştir:



Şekil 4.1. SF-PF Algoritması

Şekil 4.1'de görüldüğü gibi problem çözümü beş ana blokta tanımlı işlemlerle sağlanmaktadır. İlk adım başlatma adımıdır. Bu adımda, parçacık sayısı ve yapısı, çözüme ilişkin bilgiye dayalı olmayan ilk inanç ile dinamik model ve sensörlere ilişkin

gürültü değerleri gibi parametreler tanımlanır. Bu adımın sonunda, toplam ağırlık 1 olacak şekilde, her bir parçacığın ağırlığı eşit olarak belirlenir.

İkinci adım olan dinamik model adımında, parçacıklar, tanımlı olan hareket modeline göre çözüm uzayında hareket ettirilir. Burada, sabit hızlı hareket modeli kullanılmıştır. Bu adımın sonunda, parçacıkların durumunu ifade eden konum bilgileri değişirken ağırlıklarında henüz bir değişiklik yapılmaz.

Üçüncü adım olan gözlem modelinde ise, sensörlerden elde edilen verilerden çıkartılan bilgiler kullanılarak parçacıkların yeni ağırlıkları hesaplanır. Burada, üç adet sensör olduğu kabul edilmiştir. Bunlardan ilki Kinect cihazının kendi iç-tanımlı algoritmalarıyla ürettiği koordinatlar, ikinci ve üçüncüsü ise Kinect cihazından alınan bu koordinatların etrafındaki bir bölgenin Renk ve Derinlik imgelerinden elde edilen koordinatlardır. Bu üç ayrı koordinat bilgisi birleştirilerek kestirim adımına iletilir. Kestirim adımında, bunların içindekilerden en iyisi belirlenir. Kestirim için ortalama, en büyük ağırlığa sahip parçacık ya da sağlam ortalama seçenekleri kullanılabilir. Bu kısımda anlatılan gerçeklemede kestirim olarak, sağlam ortalama (en yüksek ağırlığa sahip %10 parçacığın ortalaması) kullanılmıştır. Kestirim adımının sonunda, seçilen parçacık, hem ağırlık ve hem de durum olarak o zaman adımına ait olan tüm parçacıklardan farklı değere sahip olabilir. Kestirim adımında elde edilen değerlerin çözüm için yeterli olup olmadığı sorgulanır, yeterli görüldüğü takdirde süreç sonlandırılır. Yeterli görülmez ise yeniden örnekleme adımı işletilir. Yeniden örnekleme adımı, zaman içinde parçacıklarda oluşan ağırlık dejenerasyonuna karşı uygulanan bir adımdır. Bu adımda, öncelikle etkin parçacık sayısının var olup olmadığı denetlenir. Etkin parçacık sayısı genellikle toplam parçacık sayısının 2/3'ü olarak kullanılır. Bu sayıda etkin parçacık var ise parçacık filtreleme tekrar dinamik modelden başlayarak işletilir. Etkin parçacık sayısı mevcut değilse bu durumda, çeşitli yeniden örnekleme yöntemlerinden birisi kullanılarak, parçacıklar yeniden örneklenir. Bu işlem sonucunda, toplam parçacık sayısı sabit kalmak üzere, ağırlığı yüksek olan parçacıkların sayısı ağırlığı oranında artılırken, düşük olanların ise sayısı da yine ağırlığı oranında azaltılır. Böylece çözüme olan yaklaşıklık artılırır. Son olarak ise, yeni oluşan parçacık kümesindeki tüm parçacıkların ağırlığı eşitlenir. Böylece

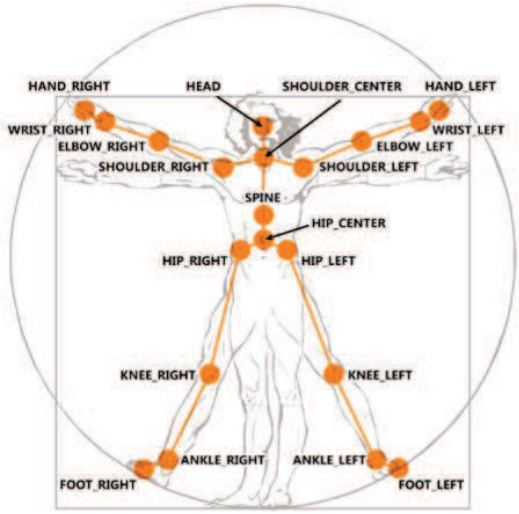
algoritma yeniden başlamış gibi işletilmeye hazır hale gelir. Bununla birlikte, önceki adımda edinilen bilgiler korunduğu için performans daha iyi hale getirilir.

4.2. Ortam, Kurulum ve Başlangıç Koşulları

Bu kısımda ele alınan problemin hangi ortamda çözüldüğü, kurulum altyapısı olarak gereksinimleri ve başlangıç varsayımlarının neler olduğu ele alınmaktadır. Öncelikle, uygulama kapalı bir ortamda ve Kinect cihazı bağlı bir bilgisayarla gerçekleştirilmiştir.

Şekil 4.2. Kinect Cihazı

Kinect, Microsoft'un geliştirdiği ve birisi kızılötesi (IR) birisi de RGB olamak üzeri iki kameradan oluşan bir cihazdır. Kinect cihazı, yazılım geliştirme araçları desteğiyle (belirli filtreler aracılığıyla) insan modelini oluşturup izleyebilmektedir. Bunun için RGB ve kızılötesi, kameralarından alınan görüntüler birlikte kullanılmaktadır. Şekil 4.2'de Kinect cihazının bir resmi, Şekil 4.3'de ise cihazın desteklediği insan modeli verilmiştir [57].



Şekil 4.3. Kinect İnsan Modeli

Şekil 4.4'de Kinect cihazından elde edilen bir çalışma ortamı resmi gösterilmektedir.

Şekil 4.4. Çalışma Ortamı

Resimde görüldüğü üzere, deneğin eli üzerinde bir işaretçi bulunmaktadır. Problem için hedeflenen, manuel olarak işaretlenen bu koordinatın işletilen algoritma ile otomatik olarak bulunmasıdır.

Bu problemin çözümünde, tüm gerçek dünya problemlerinde olduğu gibi ortam ve görünümle ilgili bir dizi başlangıç varsayımı kullanılmıştır. Bunlar aşağıda sıralanmıştır:

- Uygulama kapalı bir ortamda yapılmıştır.
- Ortamın ışık değerleri sabitlenmiştir.
- Kinect cihazı konumu sabittir (hareketsizdir).
- Sahnede yalnızca bir adet denek bulunmaktadır.
- Denek kolları kapatan sıkı bir kıyafet giymektedir.
- Denek çok hızlı olmayan bir şekilde hareket etmektedir.
- Denek hareketi Kinect cihazının görüş alanı içindeki bir bölgede olmaktadır.

4.3. Parçacık Filtresi SIR Algoritması

Tablo 4.1'de parçacık filtresi SIR algoritmasının sözde kodu verilmiştir. Şekil 4.5'de ise bu algoritmanın veri birleştirmeye dayalı olarak yapılmış, tek zaman adımındaki

uygulaması görülmektedir. Bu uygulama istenen sonuç elde edilinceye kadar tüm zaman adımlarında tekrar edilir.

Tablo 4.1. Parçacık Filtresi (SIR) Algoritması

Adım 1: Başlatma

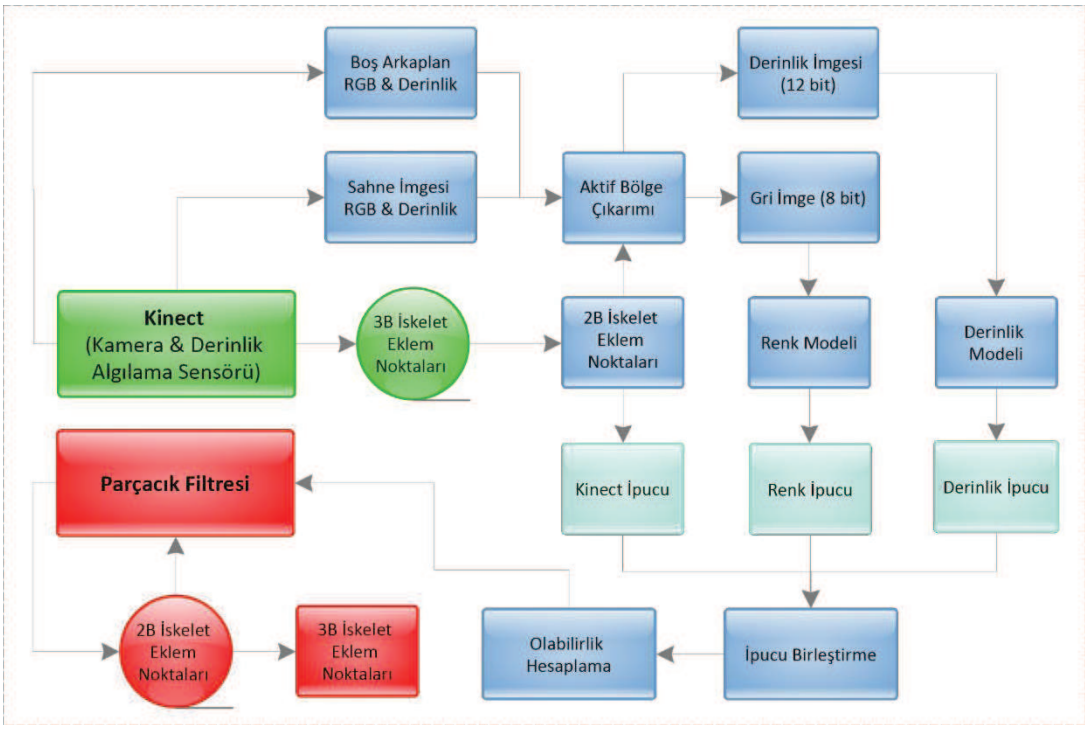
- *Gürültü parametrelerini ve başlangıç değerlerini belirle.*
- *İlk parçacık kümesini oluştur ve ilk ağırlıkları ata.*

Adım 2: Önem Örnekleme

- *Zaman adımını artır.*
- *Dinamik hareket modeline göre parçacıkların yeni konumlarını hesapla.*
- *Ölçümleri al ve kullanılabilir bilgileri çıkar.*
- *Ölçüm olabilirliğini(parçacık ağırlıklarını) hesapla.*
- *PF kestirimini elde et ve çözümün uygunluğunu denetle.*
  - *Durum 1: Çözüm uygun. **Bitir.***
  - *Durum 2: Çözüm uygun değil. Etkin parçacık sayısını denetle.*
    - *Durum 1: Etkin parçacık sayısı yeterli. **2. Adıma** git.*
    - *Durum 2: Yeniden örneklemeye ihtiyaç var. **3. Adıma** git.*

Adım 3: Yeniden Örnekleme

- *Belirlenen tekniğe göre parçacıkları yeniden örnekle.*
- *Parçacık ağırlıklarını eşitle.*
- ***2.Adıma** git.*



Şekil 4.5. SF-PF Algoritmasının İşleyişi

Öncelikle Kinect ilk veri kaynağı olarak belirlenir ve cihazın iç-tanımlı algoritmalarıyla üretilen $3B$ eklem noktalarının yatay ve dikey koordinatları alınır. Yine Kinect cihazının ürettiği RGB (8 bit 3 katman) ve derinlik (12 bit) resimleri alınarak ikinci ve üçüncü sensörler için veri kaynağı olarak tanımlanır. Kinect cihazından alınan RGB ve derinlik imgeleri, hem sahne boş iken hem de denek mevcut iken alınır. Burada, önceki bölümde de bahsedildiği gibi Kinect cihazının bulduğu sonuçların çakışma durumlarında iyi olmadığı ve bunların iyileştirilebileceği fikrinden hareket edilmiştir. Bu bağlamda, Kinect cihazının ürettiği koordinatlardan (sağ) el avuç içi eklem noktası için olan değerler, ikinci (RGB) ve üçüncü (Derinlik) tabanlı olarak elde edilecek koordinatlar için merkez olarak kabul edilmiştir. Bu kabul çerçevesinde, 141×141 boyutunda aktif pencere boyutu tanımlanarak ikinci ve üçüncü sensörlerin bu pencerede işlem yapması sağlanır. Böylece, problemin çözüm uzayı toplam pencere büyüklüğü olan 640×480 boyutundan daha küçük bir boyuta taşınarak buradan hız ve performansta etkinlik kazanılmıştır.

RGB olarak alınan imge, üç katmanlı olduğundan bunun işlem yükü daha fazla olabilmektedir. Bunun için, bu imge 8 bit'lik (256 değer) gri indeksli imgeye dönüştürülmüştür. Derinlik imgesi kaynağından 12 bit'lik olarak gelmektedir. Bunun üzerinde bir kuantalama yapılarak daha kolay işlem yapılabilecek ve işlem yükünü azaltılacak şekilde 8 bit gri imgeye dönüştürülmüştür.

Burada elde edilen imgeler sırasıyla Renk ve Derinlik modelinde kullanılarak bu imgelerden hedeflenen el koordinatları çıkartılmaktadır.

Bu adım sonrasında elde Kinect, Renk ve Derinlik olmak üzere üç veri kaynağından elde edilen $2B$ koordinat değerleri bulunmaktadır. Bu koordinat değerleri ile parçacıklar arasındaki uzaklık hesaplanarak (Denklem 3.12) her bir parçacık için üç adet uzaklık değeri bulunur. Denklem (3.13) ile verilen eşitlik kullanılarak bu uzaklıklardan olabilirlik değeri elde edilir. Olabilirlik değerleri $(0,1)$ aralığında normalize edilerek ağırlıklara dönüştürülür.

Elde edilen ağırlıklardan ortak bir ağırlık çıkarımı yapılır. Bu çıkarım her üç ağırlığın (karşılıklı elemanların) birbiriyle çarpılmasıyla yapılır. Elde edilen yeni ağırlıklar

tekrar $(0,1)$ aralığında normalize edilir. Böylece, tüm parçacıkların çözümü hangi oranda sağladıklarına ilişkin bir değer elde edilmiş olur. Parçacıklardan bir kestirim yapılarak önce $2B$ sonrada $3B$ nokta koordinatları elde edilir. Örneğin 0.05 ağırlığındaki bir parçacık çözüm için $\%5$ oranında bir doğruluk ifade edebilir. Buradan yola çıkarak, en yüksek ağırlığa sahip parçacığın probleme en iyi çözüm getirdiği düşünülebilir. Kestirim yapılırken uygulanan yöntemlerden birisi bu olsa da, bu kesin bir bilgi değildir, dolayısıyla kestirim yapılırken ortalama ya da sağlam ortalama almak bazen daha iyi sonuçlar verebilmektedir.

4.4. Dinamik Model

Parçacık filtrelemede başarımı etkileyen faktörlerden birisi de dinamik modelin uygun olarak belirlenmesidir. Burada, dinamik model olarak denklem (3.10) ile verilen sabit hızlı hareket modeli kullanılmıştır. Gerçek-zamanlı insan hareketi izleme probleminde, deneğin el avuç içi eklem noktasının her bir adımdaki düzlemsel konumu, bu dinamik model kullanılarak elde edilir. Örneğin, k zaman adımında deneğin el avuç içi eklem noktasının dikey konumu eklem noktasının $k-1$ (bir önceki) zaman adımındaki dikey konumuna sabit dikey hız değeri ve süreç gürültüsü eklenmek suretiyle hesaplanır. Parçacıkların ilk konumları, ilk zaman adımından önce belirlenir ve her bir zaman adımında herhangi bir ölçüm yapılmadan bu dinamik hareket modeli kullanılarak konumları tahmin edilir.

4.5. Gözlem Modeli

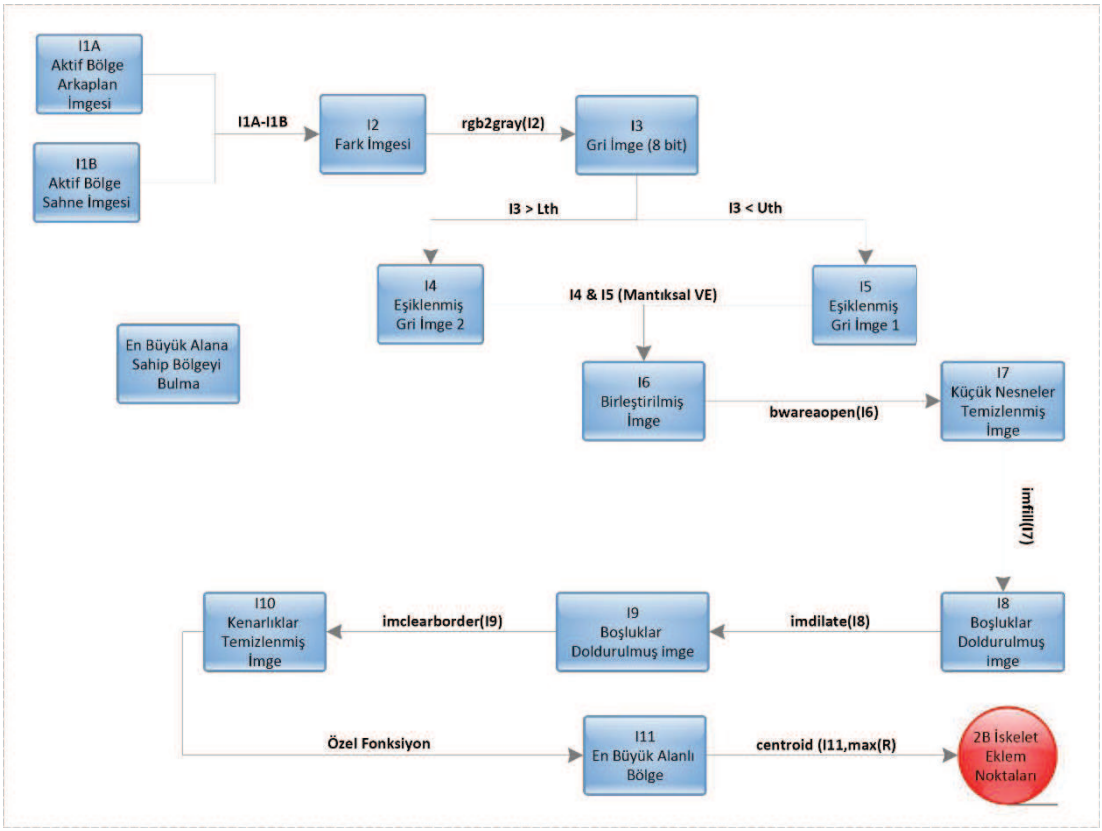
Gözlem modeli sensörlerden gelen verilerden anlamlı ve kullanışlı bilginin çıkartılmasına dayanır. Gözlem modeli parçacık filtrelemenin başarısını etkileyen en önemli faktördür. Gözlem modeli iyi belirlenmediğinde doğru sonuçlar alınması mümkün değildir.

Gerçek zamanlı denek hareketi izleme probleminde kullanılan Gözlem modelinde yukarda belirtildiği üzere üç sensör, veri kaynağı olarak kullanılmıştır. Aslında bu üç sensör, fiziki olarak yalnızca tek bir Kinect cihazından oluşmaktadır. Ancak, Kinect cihazı içinde RGB ve kızılötesi olmak üzere iki adet sensör mevcuttur. Burada, Kinect

cihazının iç-tanımlı algoritmalarla ürettiği bilgi ilk veri kaynağı olarak kullanılmış, RGB ve kızılötesi kameralardan elde edilen imgeler ise sırasıyla ikinci ve üçüncü veri kaynakları olarak kullanılmıştır. Kinect cihazının ürettiği koordinat değerleri, herhangi bir işleme tabi tutulmadan ilk ipucu değerleri olarak alınmıştır. Bu kısımda, Gözlem modelinde RGB ve kızılötesi kameralardan alınan imgeler kullanılarak bilginin nasıl çıkartıldığı ve istenen eklem noktası koordinatlarının nasıl elde edildiği açıklanmıştır.

4.5.1. Renk tabanlı gözlem modeli

Renk tabanlı gözlem modelinde, Kinect RGB kamerasından her bir zaman adımında elde edilen 640×480 boyutundaki 8 bit ve 3 katmanlı imgelerden bilginin çıkarımı anlatılmaktadır. Şekil 4.6'da bu sürecin nasıl işlediği görülmektedir.

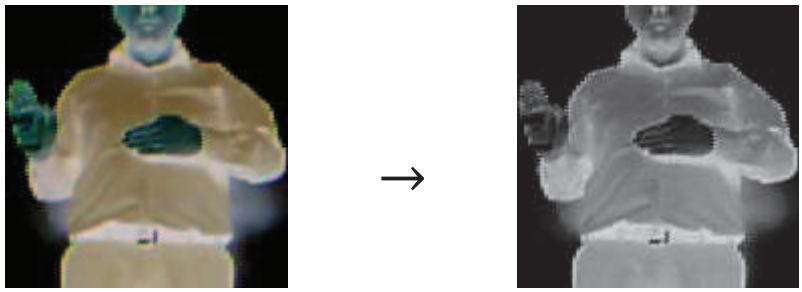


Şekil 4.6. Renk Tabanlı Gözlem Modeli

Renk modelinde, aktif bölgeye ait $I1A$-arka plan (denek olmadan) ve $I1B$-sahne (denek varken) imgeleri elde mevcuttur. Öncelikli olarak arka plandan sahne imgesi çıkartılarak sadece deneğe ait görüntü elde edilir. $\{ I2 = I1A - I1B \}$

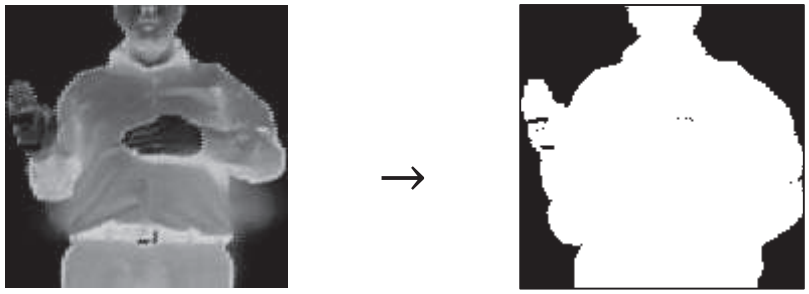
Şekil 4.8. (a) I1A Şekil 4.8. (b) I1B Şekil 4.8. (c) I2

$I2$ fark imgesi bu noktada hala 3 katmanlıdır. Bu imge 8 bit'lik gri imgeye dönüştürülür. { $I3 = rgb2gray(I2)$ }



Şekil 4.9. (a) I2 Şekil 4.9. (b) I3

$I3$ gri imgesi bu aşamada, arka arkaya iki eşikleme işlemine tabi tutulur. Küçük eşik (L_{th}) değeri ile arka plana ait artık pikseller temizlenirken, büyük eşik değeri (U_{th}) ile denek etrafındaki artık pikseller temizlenir. { $I4 = I3 < L_{th}$; $I5 = I3 > U_{th}$ }



Şekil 4.10. (a) I3 Şekil 4.10. (b) I4

Şekil 4.11. (a) I3

Şekil 4.11. (b) I5

Burada, $I4$ ve $I5$ imgeleri artık 1 bitlik (siyah-beyaz / mantıksal) imgeler haline dönüşür. Bu aşamada, $I4$ ve $I5$ imgeleri $mantıksal\ VE$ işlemine tabi tutularak, izlenmesi istenen el avuç içi eklem noktasının konumuna ilişkin bir ilk bilgi elde edilir. $\{\ I6\ =\ I4\ \&\ I5\ \}$

Şekil 4.12. (a) I4

Şekil 4.12. (b) I5

Şekil 4.12. (c) I6

Bu bilgi, deneğin üzerindeki kıyafet nedeniyle, el avuç içi eklem noktasının etrafındaki piksellerin farklı yoğunluklara sahip olması dolayısıyla elde edilir. Farklı ışık koşullarında buradan elde edilen bilgi farklılıklar gösterebilir ve tamamen yanlış olabilir. Ancak bu çalışmada ışık koşulları sabitlendiğinden üretilen değerler birbirine çok yakındır.

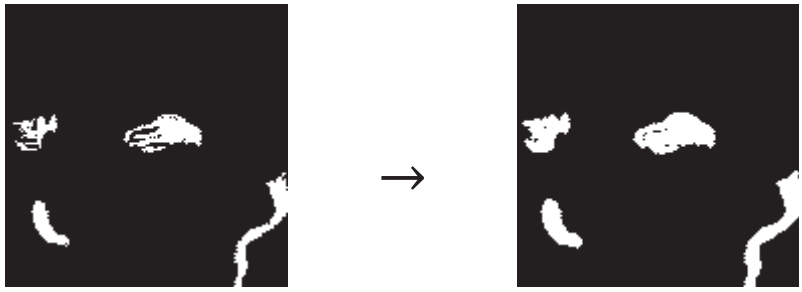
Bu aşamadan sonra, artık morfolojik işlemler arka arkaya uygulanarak izlenmesi istenen el avuç içi eklem noktasını içeren en uygun imge elde edilmeye çalışılır. İlk olarak, birleştirilmiş imgedeki küçük imgelerin temizlenmesi sağlanır. Böylece, sadece belli büyüklükteki, bir başka ifadeyle el olma ihtimali daha yüksek olan bölgelere sahip bir imge elde edilmiş olur. $\{\ I7\ =\ bwareaopen(I6,50)\ \}$

Şekil 4.13. (a) I6 Şekil 4.13. (b) I7

Sonraki adımda, el olması ihtimali yüksek olan bölgelerde mevcut olan boşluklar doldurularak bu bölgelerin daha belirgin hale gelmesi sağlanır. $\{ I8 = imfill(I7, 'holes') \}$

Şekil 4.14. (a) I7 Şekil 4.14. (b) I8

Boşluk doldurma işlemi, sadece etrafı bölge ile çevrili olan piksellere uygulanan bir işlemdir. Bu işlem bağlantısı kopmuş olan bölgeleri birbirine bağlama işlemi için uygun değildir. Bunun için daha uygun olan eklemleme (dilation) işlemi kulanılarak bağlantısı kopmuş olan bölgeler birleştirilir. $\{ I9 = imdilate(I8, 'diamond', 1) \}$



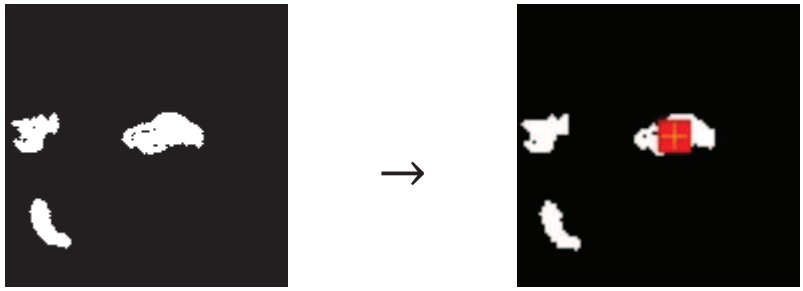
Şekil 4.15. (a) I8 Şekil 4.15. (b) I9

Son olarak ise, yapılan morfolojik işlemlerin etkisiyle imgede oluşan kenar piksellerinin temizlenmesi ya da en aza indirilmesi için kenar temizleme işlemi uygulanır. { $I10 = imclearborder(I9,8)$ }

Şekil 4.16. (a) I9 Şekil 4.16. (b) I10

Bu aşamadan sonra, içinde, izlenmesi istenen elin belirgin şekilde bulunma ihtimali yüksek olan bir imge elde edilmiş olur. Burada, mevcut imgedeki en büyük alana sahip bölgenin el olduğu kabulüyle, imge üzerindeki en büyük alanlı bölge, özel bir fonksiyonla tespit edilir. { $[R, ind] = TTGetLAC(I10)$ }

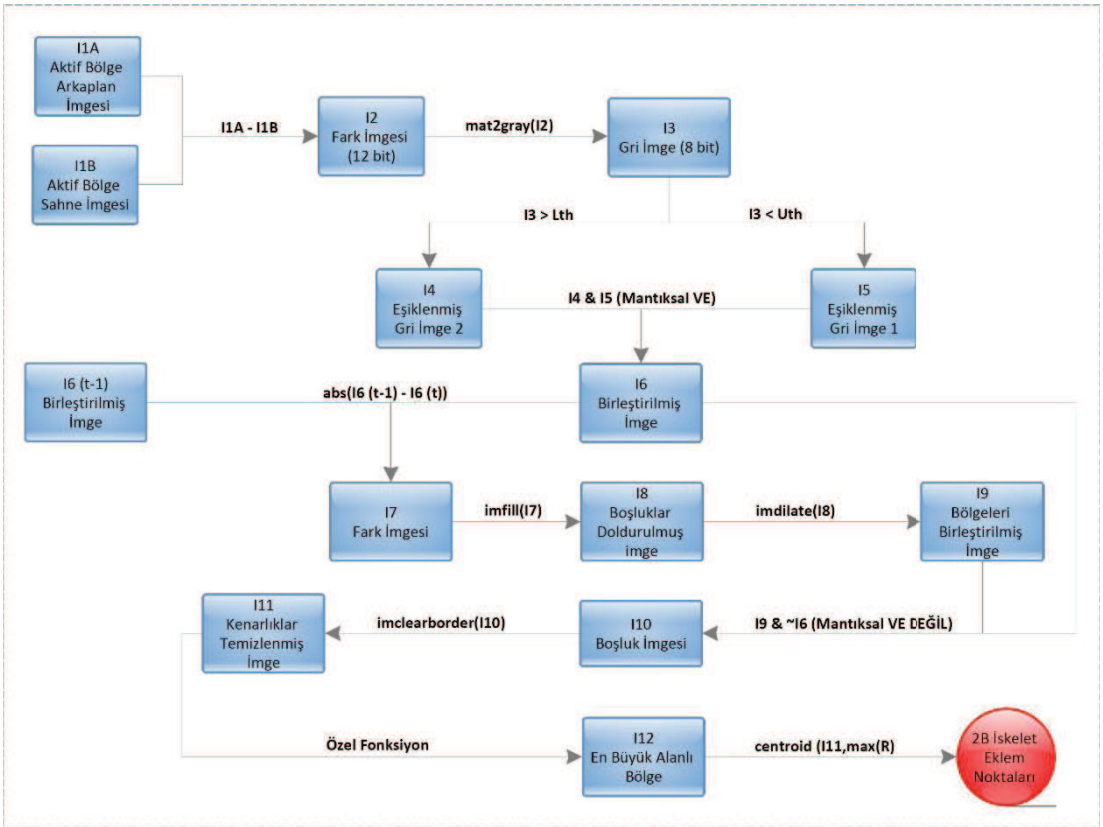
Özel fonksiyonla bulunan bölgenin merkez koordinatları da izlenmesi istenen el avuç içi eklem noktasının koordinatları olarak alınır. { $[elx, ely] = centroid(R)$ }



Şekil 4.17. (a) I10 Şekil 4.17. (b) I11

4.5.2. Derinlik tabanlı gözlem modeli

Derinlik Gözlem modelinde Kinect kızılötesi kamerasından her bir zaman adımında elde edilen 640×480 boyutundaki 12 bit'lik imgelerden bilginin çıkarımı anlatılmaktadır. Şekil 4.7'de bu sürecin nasıl işlediği görülmektedir.



Şekil 4.7. Derinlik Tabanlı Gözlem Modeli

Derinlik modelinde, aktif bölgeye ait $I1A$-arka plan (denek olmadan) ve $I1B$-sahne (denek varken) imgeleri elde mevcuttur. Öncelikli olarak arka plandan sahne imgesi çıkartılarak sadece deneğe ait görüntü elde edilir: $\{\ I2\ =\ I1A - I1B\ \}$

Şekil 4.18. (a) I1A

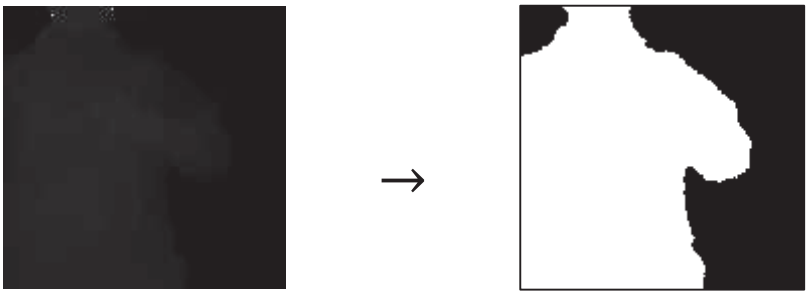
Şekil 4.18. (b) I1B

Şekil 4.18. (c) I2

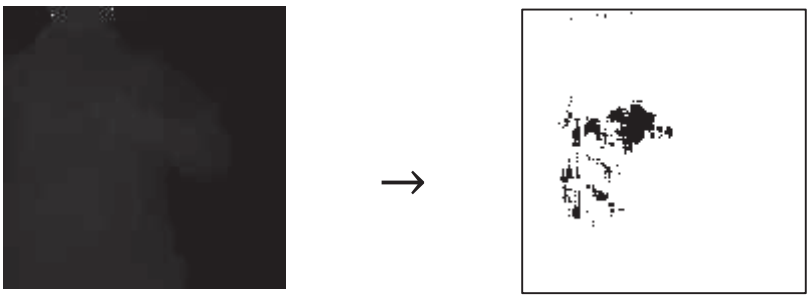
$I2$ fark imgesi, bu noktada hala 12 bit'liktir. Bu imge 8 bit'lik gri imgeye dönüştürülür. Şekil 4.19 (c)'de $I3$ imgesi histogram eşitleme işlemi uygulandıktan sonra daha net görünmektedir. Burada, Şekil 18 (a), (b) ve (c) de imgelerin karanlık görünmelerinin sebebi piksel değerlerinin 0'a çok yakın olmasıdır. $\{\ I3\ =\ mat2gray(255 * I2)\ \}$

Şekil 4.19. (a) I2 Şekil 4.19. (b) I3 Şekil 4.19. (c) I3*

$I3$ gri imgesi bu aşamada, arka arkaya iki eşikleme işlemine tabi tutulur. Küçük eşik değeri ile arka plana ait artık pikseller temizlenirken, büyük eşik değeri ile denek etrafındaki fazlalık pikseller temizlenir. $\{ I4 = I3 < L_{th} ; I5 = I3 > U_{th} \}$

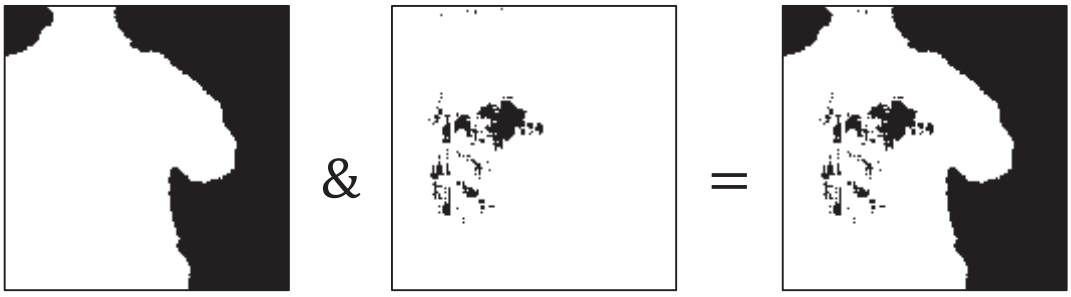


Şekil 4.20. (a) I3 Şekil 4.20. (b) I4



Şekil 4.21. (a) I3 Şekil 4.21. (b) I5

Burada, $I4$ ve $I5$ imgeleri artık 1 bit'lik imgeler haline dönüşür. Bu aşamada, $I4$ ve $I5$ imgeleri $mantıksal\ VE$ işlemine tabi tutularak, izlenmesi istenen el avuç içi eklem noktasının konumuna ilişkin bir ilk bilgi elde edilir. $\{ I6 = I4 \,\&\, I5 \}$



Şekil 4.22. (a) I4 Şekil 4.22. (b) I5 Şekil 4.22. (c) I6

Derinlik gözlem modelinde bu aşamada, önceki zaman adımının aynı aşamasında elde edilmiş olan birleştirilmiş imge kullanılır. Önceki zaman adımına ait imge ile mevcut zaman adımında elde edilmiş olan imge arasındaki mutlak fark alınarak hareketin olduğu pikseller tespit edilir. Bu işlem dolayısıyla, deneğin el haricindeki hareket eden kısımlarına ilişkin pikseller de seçilebilir. Ancak en büyük hareketi el bölgesinde gerçekleştiği kabul edildiğinden, elin büyük bir bölge olarak içinde olduğu bir imge elde edilmesi kuvvetle muhtemeldir. $\{ I7 = abs(I6(t-1) - I6(t)) \}$

Şekil 4.23. (a) I6 (t-1) Şekil 4.23. (b) I6 (t) Şekil 4.23. (c) I7

Bu aşamadan sonra, artık morfolojik işlemler arka arkaya uygulanarak izlenmesi istenen el avuç içi eklem noktasını içeren en uygun imge elde edilmeye çalışılır. İlk olarak, birleştirilmiş imgedeki el olması ihtimali yüksek olan bölgelerde mevcut olan boşluklar doldurularak bu bölgelerin daha belirgin hale gelmesi sağlanır.

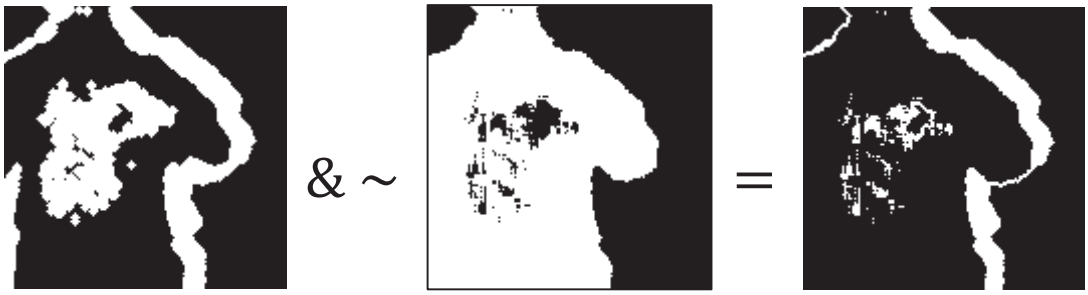
$\{ I8 = imfill(I7, 'holes') \}$

Şekil 4.24. (a) I7 Şekil 4.24. (b) I8

Bir sonraki adımda, bağlantısı kopmuş olan bölgeleri birbirine bağlamak için eklemleme (dilation) işlemi kullanılarak, el olması ihtimali yüksek olan bölgelerin daha belirgin hale gelmesi sağlanır. { $I9 = imdilate(I8,'diamond',1)$ }

Şekil 4.25. (a) I8 Şekil 4.25. (b) I9

Derinlik tabanlı gözlem modelinde bu aşamada, $I9$ imgesindeki boşluk piksellerini elde etmek için daha önce elde edilmiş $I6$ imgesi ile $mantıksal\ VE\ DE$ĞİL işlemi uygulanır. { $I10 = I9\ \&\ {\sim}I6$ }



Şekil 4.26. (a) I9 Şekil 4.26. (b) I6 Şekil 4.26. (c) I10

Son olarak ise, yapılan morfolojik işlemlerin etkisiyle imgede oluşan kenar piksellerinin temizlenmesi ya da en aza indirilmesi için kenar temizleme işlemi uygulanır. { $I11 = imclearborder(I10,4)$ }

Şekil 4.27. (a) I10

Şekil 4.27. (b) I11

Bu aşamadan sonra, içinde izlenmesi istenen elin belirgin şekilde bulunma ihtimali yüksek olan bir imge elde edilmiş olur. Burada, mevcut imgedeki en büyük alana sahip bölgenin el olduğu kabulüyle imge üzerindeki en büyük alanlı bölge özel bir fonksiyonla tespit edilir. { $[R, ind] = TTGetLAC(I11)$ }

Özel fonksiyonla bulunan bölgenin merkez koordinatları da izlenmesi istenen el avuç içi eklem noktasının koordinatları olarak bulunur. { $[elx, ely] = centroid(R)$ }

Şekil 4.28. (a) I11

Şekil 4.28. (b) I12

4.6. Veri Birleştirme

Bu çalışmada gerçekleştirilen veri birleştirmeye dayalı parçacık filtreleme ile hareket izleme sürecinin iyileştirilmesi uygulamasının, klasik parçacık filtreleme uygulamalarından temel farkı veri birleştirme adımıdır. Veri birleştirme, aslında birçok gerçek-dünya probleminin çözümünde uygulanması muhtemel olan bir işlemdir. Farklı sensörlerden elde edilen verilerden problemin yapısına uygun olarak çıkarılan anlamlı bilgilerin Bayesçi olasılık temelinde birleştirilerek daha iyi sonuçlara ulaşılabildiği teorik olarak kanıtlamış ve literatürdeki ve güncel hayattaki birçok uygulamaya yansıtılmıştır. Bu bağlamda, bu çalışmada da sahnede hareket eden deneğin belli bir eklem noktasının (el) hareketi, Kinect cihazının kendi iç tanımlı

algoritmalarla gerçek-zamanlı olarak izlendiğinde çakışma durumlarında oluşan hataların azaltılması amacıyla yine aynı cihazdan alınan RGB ve derinlik imgelerinden çeşitli imge işleme teknikleri kullanılarak çıkarılan bilgi, Kinect cihazından gelen bilgiyle birleştirilmiş ve daha iyi sonuçlar ortaya çıktığı gösterilmiştir. Çalışmada yapılan uygulamada, aşağıdaki adımlarla veri birleştirme sağlanmıştır:

Renk (R), Derinlik (D) ve Kinect (K) olarak belirlenmiş veri kaynaklarının her biri kullanılarak hesaplanan koordinatlar ile insan operatörlerin belirlediği noktalar arasındaki öklid uzaklığı hesaplanmıştır. (Denklem 4.1)

$$L_{st}^{(k)} = \sqrt{\left(p_{x_i}{}^{(k)} - st_x{}^{(k)}\right)^2 + \left(p_{y_i}{}^{(k)} - st_y{}^{(k)}\right)^2}, st = \{K, R, D\}, i = 1..N \quad \textbf{(4.1)}$$

Burada, $L_{st}^{(k)}$ ilgili sensör için k zaman adımında hesaplanan uzaklık matrisi, $p_{x_i}{}^{(k)}$ ve $p_{y_i}{}^{(k)}$ de sırasıyla $i.$ parçacığın, yatay ve dikey konumlar, N ise parçacık sayısıdır.

Her bir veri kaynağı için hesaplanan uzaklıklar kullanılarak olabilirlik (likelihood) değerleri hesaplanır. (Denklem 4.2)

$$Lh_{st}^{(k)} = e^{\frac{-L_{st}^{(k)}}{2\rho_{st}}}, st = \{K, R, D\}, k = 1..T \quad \textbf{(4.2)}$$

Burada, T simülasyon zamanı, $Lh_{st}^{(k)}$ ilgili sensör için k zaman adımındaki olabilirlik değerleri, ρ_{st}, ise gürültü varyanslarıdır. Olabilirlik değeri hesaplanırken, uzaklık değeri ile olabilirlik arasında ters orantılı bir ilişki kurulmuştur. Böylece, insan operatörlerin belirlediği noktalara yakın olan parçacıkların olabilirliği yüksek, uzak olanların ise düşük olarak elde edilir.

Olabilirlik değerleri hesaplandıktan sonra, her bir veri kaynağı için bu değerler (0,1) aralığına normalize edilir. Daha sonra bu olabilirlik değerleri denklem (4.3) kullanılarak birleştirilir.

$$Lh^{(k)} = \widetilde{Lh}_K^{(k)} \otimes \widetilde{Lh}_R^{(k)} \otimes \widetilde{Lh}_D^{(k)}, k = 1..T \quad \textbf{(4.3)}$$

Burada, $Lh^{(k)}$ k zaman adımındaki birleştirilmiş olabilirlik değeri, $\widetilde{Lh}_K^{(k)}$, $\widetilde{Lh}_R^{(k)}$ ve $\widetilde{Lh}_D^{(k)}$ de sırasıyla Kinect, Renk ve Derinlik kaynaklarından elde edilen olabilirlik değerlerinin normalize edilmiş halidir.

Veri kaynakları için k zaman adımında elde edilen birleştirilmiş olabilirlik değeri $Lh^{(k)}$, parçacıkların önceki zaman adımındaki ağırlıklarıyla çarpılarak yeni parçacık ağırlıkları elde edilir. Elde edilen yeni parçacık ağırlıkları ile kestirim yapılır. Yapılan kestirimin uygunluğuna göre parçacıklar yeniden örnekleme adımına taşınır.

BÖLÜM 5. BULGULAR VE DEĞERLENDİRME

Veri birleştirmeye dayalı parçacık filtresi ile bu çalışmada ele alınan problem Microsoft Kinect cihazının gerçek-zamanlı olarak yürüttüğü hareket izleme sürecinin iyileştirilmesidir. Bu kısımda, yapılan çalışma sonucunda elde edilen bulgular paylaşılmış ve bu bulgular ile ilgili değerlendirmeler yapılmıştır.

Çalışmada, dinamik sistemin hareket modeli ya da diğer bir ifadeyle parçacık hareketi için gürültü değerleri deneysel olarak seçilmiştir. k zaman adımını göstermek üzere sırasıyla yatay konum, düşey konum, yatay hız ve düşey hız için gürültü değerleri her bir parçacık için aynı olacak şekilde $varX_k = [0.8;\ 0.6; 0.2; 0.15]$ olarak belirlenmiştir. Hareket denklemi olarak ise Denklem (5.1) uygulanmıştır.

$$X_k = dyneq\ \times X_{k-1} + \sqrt{varX} \otimes randn(size(varX)) \tag{5.1}$$

$$dyneq = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.2}$$

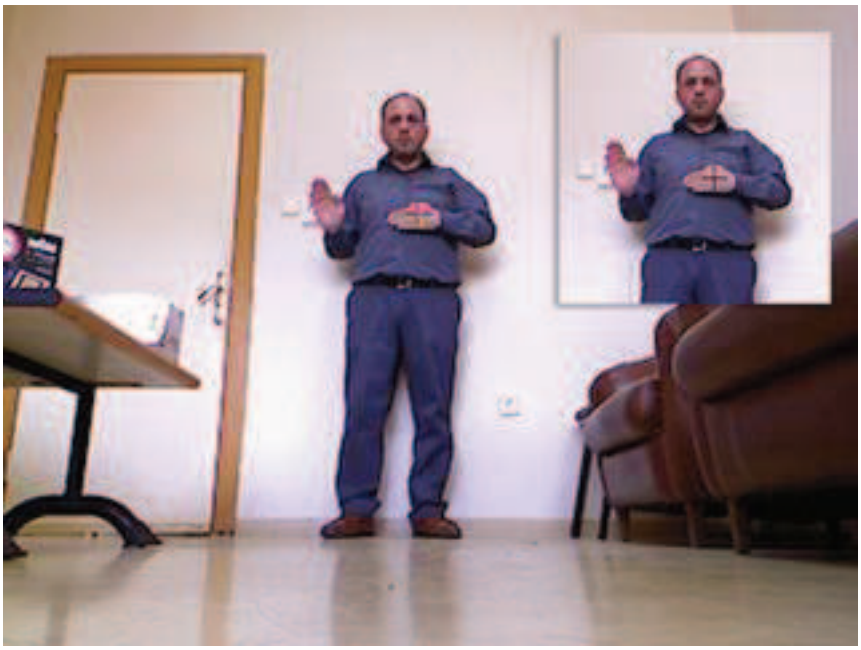
Burada, $dyneq$ sabit hızlı hareket modeli katsayı matrisini, $\otimes$ elemanların karşılıklı çarpımını ve $randn$ de standart normal dağılımla oluşturulan rassal bir değeri ifade etmektedir. Uygulamada, Kinect (K), Renk (R) ve Derinlik (D) sensörleri için de gürültü değeri $varK = varR = varD = 12$ olarak belirlenmiştir.

Bölüm 4.6'da veri birleştirme adımında açıklanan, uzaklık (Denklem 4.1) ve olabilirlik (Denklem 4.2) hesaplama eşitlikleri, veri birleştirmeye dayalı parçacık filtresi ile gerçek-zamanlı hareket izleme iyileştirme probleminin çözümünde kullanılmıştır. Burada, ipuçlarının elde edilmesinde, bulunan el koordinat merkezine sırasıyla Renk ve Derinlik için standart sapması 15 ve 7 olan Gauss gürültüsü eklenmiştir. Kinect

cihazının ürettiği değerler ise zaten gürültülü varsayıldığından bu değerler olduğu gibi kullnılmış, herhangi bir gürültü eklenmemiştir. Renk ve Derinlik ipuçları için gürültü eklenmesinin nedeni, bu çalışmada uygulanan varsayımlar dolayısıyla bulunan koordinat değerlerinin filtrelemeye ihtiyaç bırakmamasıdır. Bununla birlikte, gerçek dünya problemlerinde kaynaklardan gelen veriler hemen her durumda gürültülü olduğu için filtreleme yoluyla bu kaynaklardan elde edilen bilgilerle daha iyi çözümler elde edilmesi çoğu zaman mümkün olur.

5.1. Yöntem Geçerlilik Testi Sonuçları

Veri birleştirmeye dayalı parçacık filtreleme ile hareket izleme sürecinin iyileştirilmesi probleminin çözümünde kullanılan yöntemin performansı insan operatörlerin bulduğu sonuçlar ile karşılaştırılarak test edilmiştir. Bu test sürecinde, üzerinde işlem yapılan 111 kareli bir video çekimi, bir web arayüzü üzerinden insan operatörlerin kullanımına açılmıştır. Çalışmada 7 insan operatör 111 kare için seçim yapmıştır. Hareket eden insan deneğin elinin merkez koordinatları, yine insan operatörlerin her karede yaptıkları seçimlerle kayıt altına alınmış ve bunların ortalaması ilgili kare için referans kabul edilmiştir.



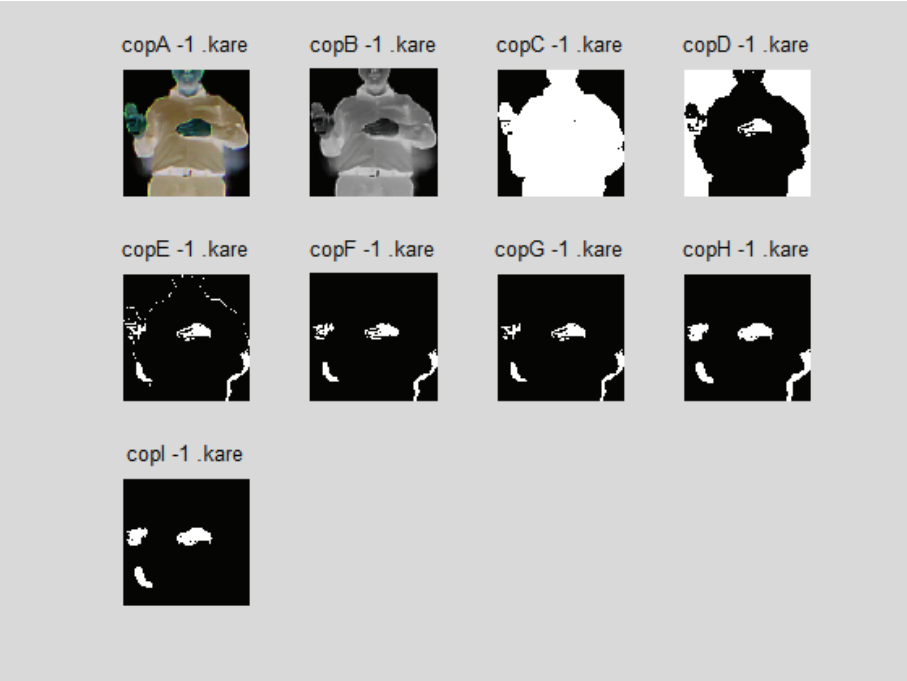
Şekil 5.1. Referans Değerler (1.Kare)

Şekil 5.1 ve 5.2'de sırasıyla 1. ve 81. karelerde operatörlerin yaptıkları seçimler (büyük resimler) ile bunların ortalamaları (küçük resimler) görünmektedir. Her karede (zaman adımında) renk tabanlı, derinlik tabanlı ve Kinect cihazından elde edilen koordinatlar ile veri birleştirmeye dayalı parçacık filtreleme ile kestirimi yapılan koordinatlar operatörlerin yaptığı seçimlerin ortalamalarıyla karşılaştırılmış ve performans değerleri elde edilmiştir.

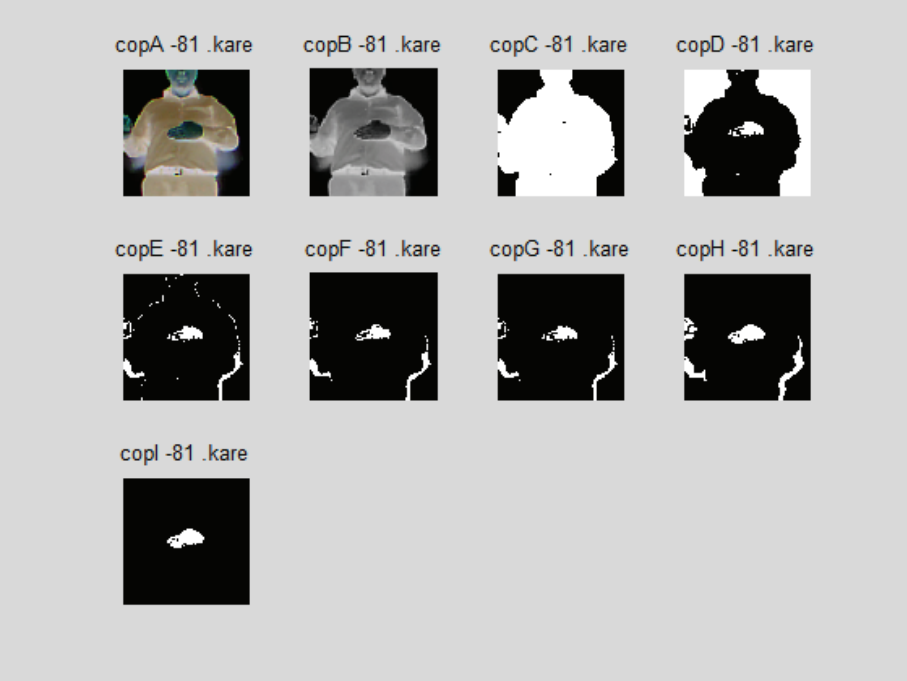
Şekil 5.2. Referans Değerler (81.Kare)

5.1.1. Renk ipucu sonuçları

Veri birleştirmeye dayalı parçacık filtreleme ile hareket izleme sürecinin iyileştirilmesi probleminin çözümünde, renk ipucunun elde edilmesinde kullanılan yöntemler Bölüm 4.5.1'de anlatılmıştı. Her bir zaman adımında uygulanan işlem ile elde edilen sonuç imgeleri Şekil 5.3 (1. kare) ve 5.4 (81. kare) 'de verilmiştir.



Şekil 5.3. Renk İpucu İşlemleri (1.Kare)



Şekil 5.4. Renk İpucu İşlemleri (81.Kare)

Renk ipucunun elde edilmesi sürecinde, kaynak olarak Kinect cihazının ürettiği imgelerden elde edilen (kırpılan) 8 bit'lik 141×141 boyutundaki RGB imgeler (boş ve dolu sahne imgeleri) kullanılmıştır. Gerçek zamanlı bir hareket izleme süreci hedeflendiğinden, burada ağırlıklı olarak 1 bit'lik imgeler üzerinde mantıksal

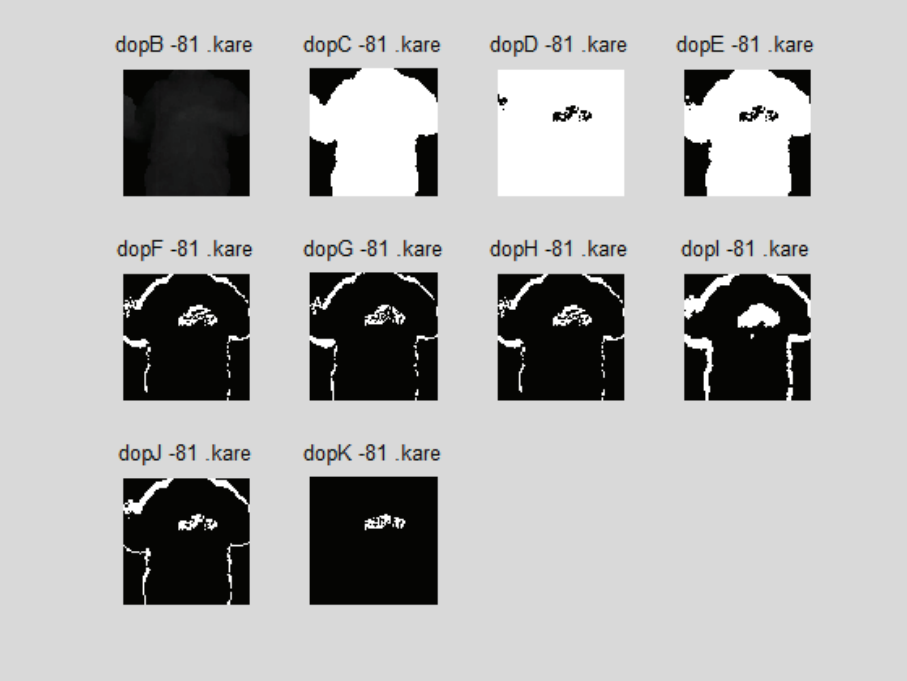
operatörler ile morfolojik işlemler yapılmıştır. Kabul edilen varsayımlar dolayısıyla, hedeflenen koordinatlar büyük bir doğrulukla elde edilmiştir. Ancak, gerçek dünya problemlerinde bu varsayımlar geçerli olmadığından, filtrelemenin etkinliğinin gösterilmesi açısından bulunan koordinat değerlerine Gauss gürültüsü eklenmiştir.

5.1.2. Derinlik ipucu sonuçları

Veri birleştirmeye dayalı parçacık filtreleme ile hareket izleme sürecinin iyileştirilmesi probleminin çözümünde, derinlik ipucunun elde edilmesinde kullanılan yöntemler Bölüm 4.5.2'de anlatılmıştı. Her bir zaman adımında uygulanan işlem ile elde edilen sonuç imgeleri Şekil 5.5 (1. kare) ve 5.6 (81. kare) 'de verilmiştir.

Derinlik ipucunun elde edilmesi sürecinde, kaynak olarak Kinect cihazının ürettiği derinlik imgelerinden elde edilen 12 bit'lik 141×141 boyutundaki imgeler (boş ve dolu sahne imgeleri) kullanılmıştır. Renk ipucunun elde edilmesinde olduğu gibi, derinlik ipucunun elde edilmesinde de gerçek-zaman kısıtlamasından dolayı imgeler üzerinde mantıksal operatörler ile morfolojik işlemler uygulanması tercih edilmiştir. Derinlik ipucunun elde edilmesinde ik bir işlem adımı daha uygulanmıştır. Bu adımda, imgedeki hareketli bölgelerin tespiti için bir önceki zaman adımında elde edilen imge ile mevcut zaman adımındaki imge arasındaki mutlak fark hesaplanmıştır. Derinlik ipucu için uygulanan yöntemlerde, kabul edilen varsayımlar renk ipucu için kabul edilen varsayımlara göre daha az kısıtlayıcı olduğundan hedeflenen koordinatlar renk tabanlı modele göre daha düşük doğruluk düzeyiyle elde edilmiştir. Renk tabanlı modele benzer şekilde, derinlik tabanlı modelde de filtrelemenin etkinliğinin gösterilmesi için bulunan koordinat değerlerine Gauss gürültüsü eklenmiştir.

Şekil 5.5. Derinlik İpucu İşlemleri (1.Kare)



Şekil 5.6. Derinlik İpucu İşlemleri (81.Kare)

5.2. El Bulma İşlemi Sonuçları

Renk ve derinlik ipuçlarının elde edilmesi süreçlerindeki ortak son işlem adımı, var olan son imge üzerindeki en büyük alana sahip bölgenin merkez koordinatlarının basit

bir algoritmayla bulunması işlemidir. Şekil 5.7 ve 5.8'de 61, 71, 81 ve 91. karelerde renk tabanlı (kırmızı) ve derinlik tabanlı (yeşil) olarak hesaplanan koordinatların bulunduğu konum işaretlenmiş olarak görülmektedir. Burada, özellikle derinlik tabanlı modelde zaman zaman yanlış konumlar hesaplanmıştır. Bununla, birlikte uygulanan veri birleştirmeye dayalı filtreleme yöntemi dolayısıyla, bu tür yanlış sonuçlar da tolera edilebilmiştir.

Şekil 5.7. Renk & Derinlik Tabanlı El Bulma 61 ve 71. Kareler

Şekil 5.8. Renk & Derinlik Tabanlı El Bulma 81 ve 91. Kareler

5.3. Filtreleme Sonuçları Karşılaştırması

Gerçek-zamanlı hareket izleme iyileştirme sürecinde, yapılan uygulamaya ilişkin öncelikli olarak, parçacık sayısının artışına bağlı olarak ortaya çıkan göresel sonuçlar ile bu sonuçlarla bağlantılı ortalama karesel hata değerleri karşılaştırmalı olarak verilmiştir. Çalışmada $N = \{10,25,50,100,250,500,1000,2500,5000\}$ parçacık sayıları için sonuçlar elde edilmiştir.

Şekil 5.9. Parçacık Sayısı: 10 (Kare 1 & 81)

Şekil 5.10. Parçacık Sayısı: 25 (Kare 1 & 81)

Şekil 5.11. Parçacık Sayısı: 50 (Kare 1 & 81)

Şekil 5.12. Parçacık Sayısı: 100 (Kare 1 & 81)

Şekil 5.13. Parçacık Sayısı: 250 (Kare 1 & 81)

Şekil 5.14. Parçacık Sayısı: 500 (Kare 1 & 81)

Şekil 5.15. Parçacık Sayısı: 1000 (Kare 1 & 81)

Şekil 5.16. Parçacık Sayısı: 2500 (Kare 1 & 81)

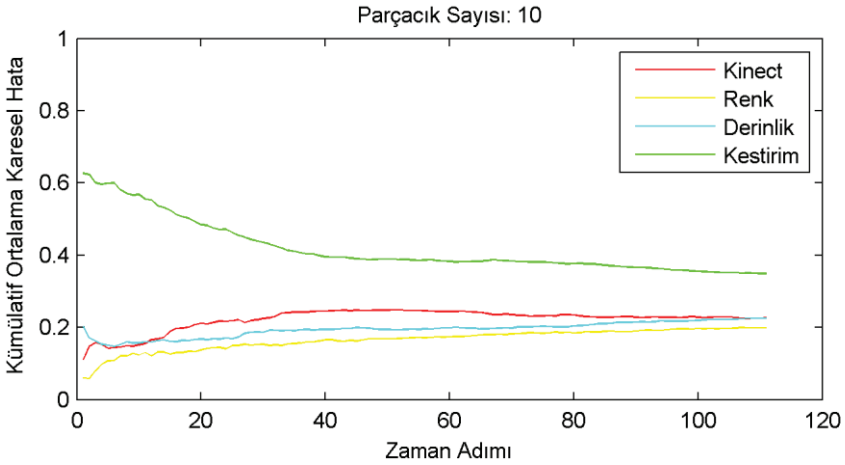
Şekil 5.17. Parçacık Sayısı: 5000 (Kare 1 & 81)

5.4. Ortalama Karesel Hata Değerleri

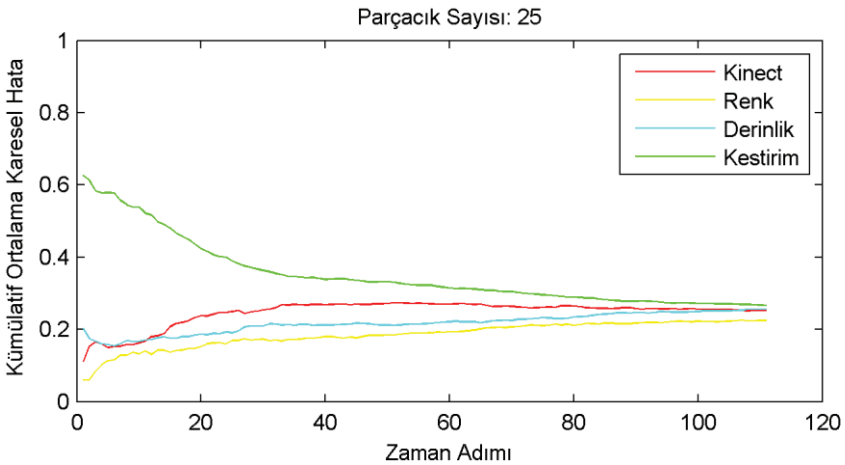
Bu kısımda, $N = \{10,25,50,100,250,500,1000,2500,5000\}$ parçacık sayıları için hesaplanan normalize edilmiş kümülatif ortalama karesel hata değerlerini içeren grafikler verilmiştir. Bu değerlerin hesaplanması için öncelikle, veri birleştirmeye dayalı parçacık filtreleme uygulaması 20 kez arka arkaya çalıştırılmıştır. Bu çalıştırmaların sonucunda, parçacık sayısı denemelerinin her birisi için satırları üzerinde çalışılan 111 zaman adımı, sütunları sırasıyla Kinect, Renk, Derinlik ve Kestirim tabanlı olarak elde edilen değerler olmak üzere $111 \times 4 \times 20$ boyutunda ortalama karesel hata değerleri matrisi oluşturulmuştur. Daha sonra, bu matristeki değerlerin zaman adımı bazındaki ortalamaları alınmıştır. Örneğin, her bir çalıştırmanın 1. zaman adımında parçacık filtresi kestirimi ile elde edilen ortalama karesel hataların ortalaması alınarak bir zaman adımı için bir hata değeri elde edilmiştir. Sonraki adımda, her bir parçacık sayısı denemesi için elde edilmiş olan ortalama karesel hata değerleri, sensör bazında normalize edilmiştir. Örnek olarak, 45. zaman adımında Kinect, Renk, Derinlik ve Kestirim tabanlı olarak elde edilen değerler $(0,1)$ aralığında normalize edilmiştir. Son adımda ise, normalize edilmiş ortalama

karesel hata değerlerinin kümülatif ortalamaları hesaplanmıştır. Böylece, Kinect, Renk, Derinlik ve Kestirim tabanlı olarak hesaplanan hata değerlerinin zaman adımları boyunca seyri elde edilmiştir.

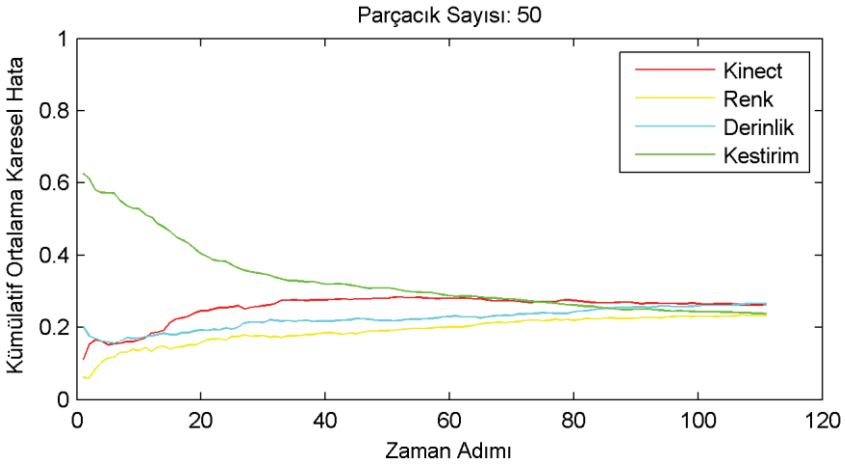
Şekil 5.18, 5.19 ve 5.20 incelendiğinde, parçacık sayısı 10, 25 ve 50 olduğunda filtreleme sonuçlarının uygun olmadığı anlaşılmaktadır. Bunun temel nedeni ise parçacık sayısının yetersiz olmasıdır.



Şekil 5.18. Kümülatif Ortalama Karesel Hata Değerleri (N=10)

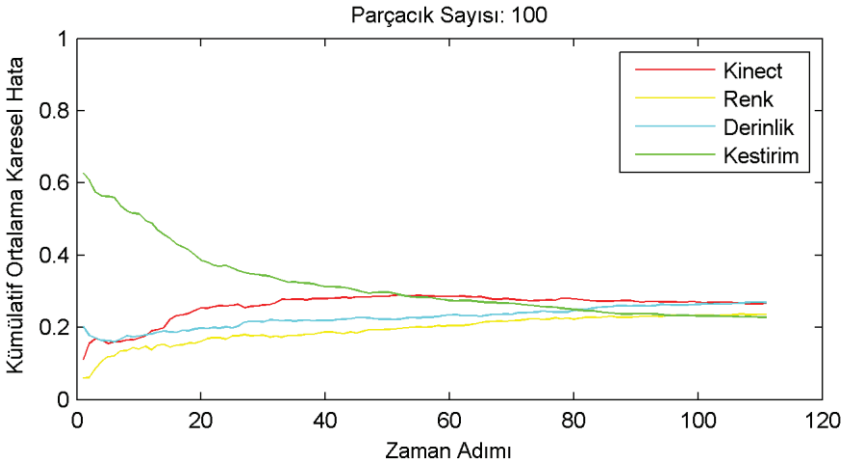


Şekil 5.19. Kümülatif Ortalama Karesel Hata Değerleri (N=25)

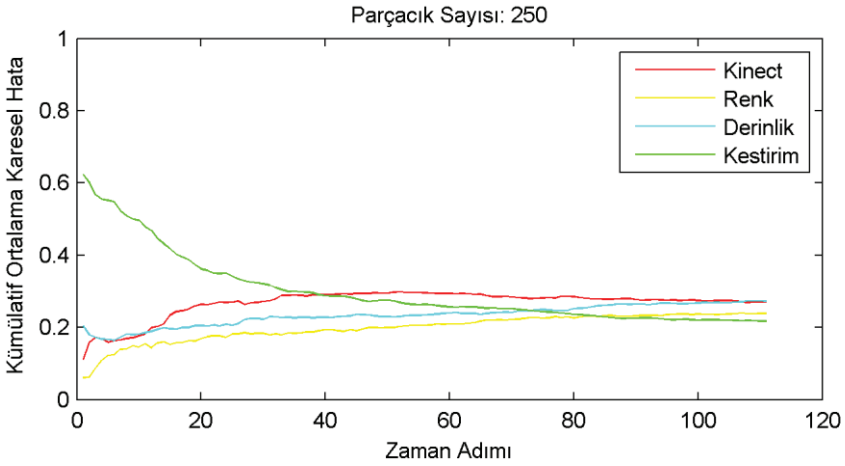


Şekil 5.20. Kümülatif Ortalama Karesel Hata Değerleri (N=50)

Parçacık sayısı 100 ve 250 seviyelerine çıktığında, sonuçların daha makul olduğu Şekil 5.21 ve 5.22'den anlaşılmaktadır. Burada, parçacık sayısı artışıyla doğru orantılı olarak parçacık kestirimi kestiriminin ortalama karesel hata değerlerinin kümülatif ortalamasının zaman adımları boyunca azaldığı ve Kinect cihazının sonuçlarına göre daha iyi olmaya başladığı görülmektedir.

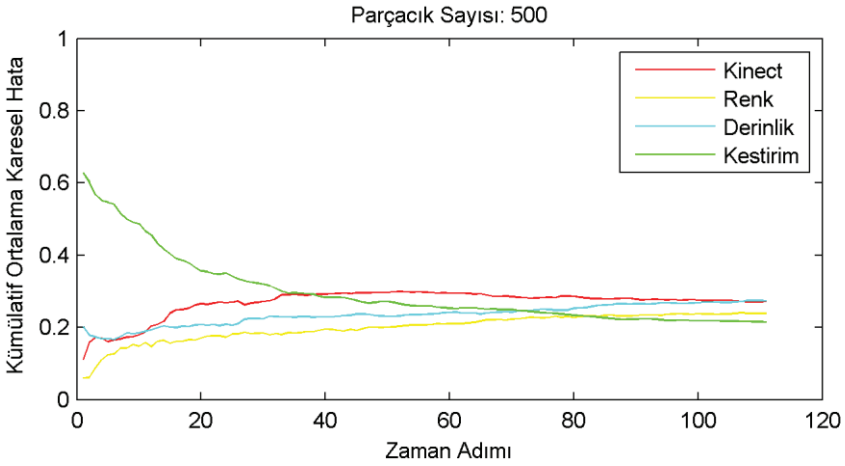


Şekil 5.21. Kümülatif Ortalama Karesel Hata Değerleri (N=100)

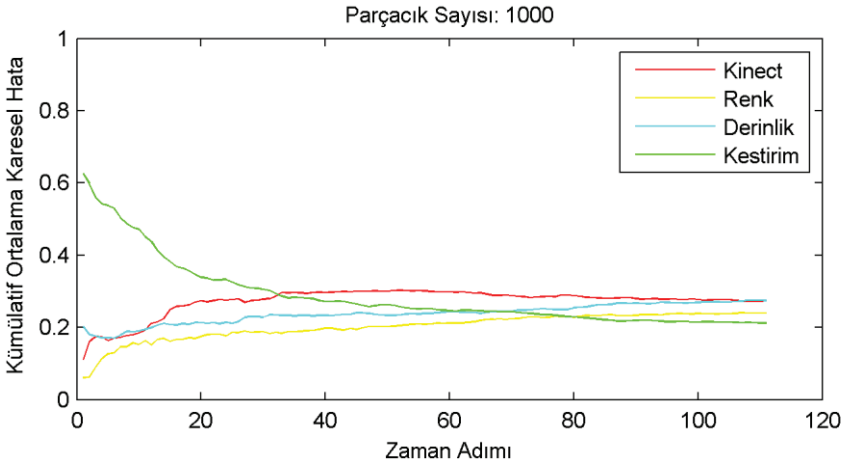


Şekil 5.22. Kümülatif Ortalama Karesel Hata Değerleri (N=250)

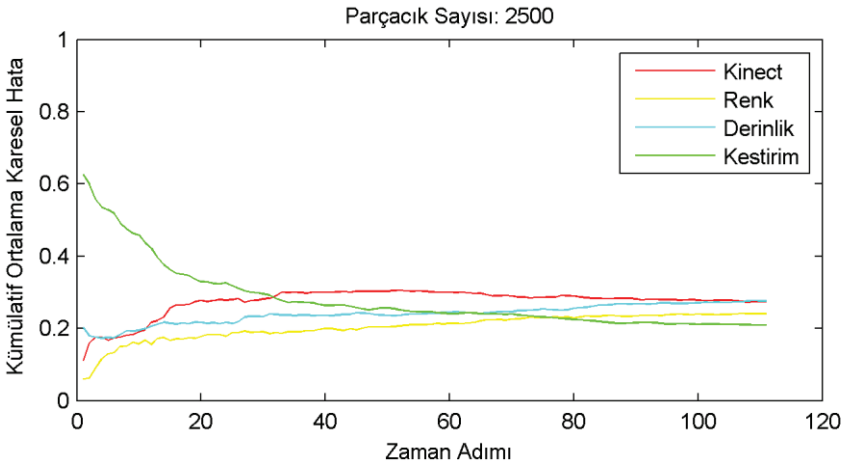
Şekil 5.23 – 5.26 arasında parçacık sayısının 500, 1000, 2500 ve 5000 olduğu durumlar görünmektedir. Burada, parçacık sayısı önceki denemelere göre oldukça yüksek oranlara çıktığı halde, kümülatif ortalama karesel hatanın ortalamasının büyük bir azalma göstermediği anlaşılmaktadır. Sonuçlar karşılıklı olarak incelendiğinde, parçacık sayısının 100 ile 250 arasında olduğu durumun uygun bir çözüm olduğu düşünülebilir.



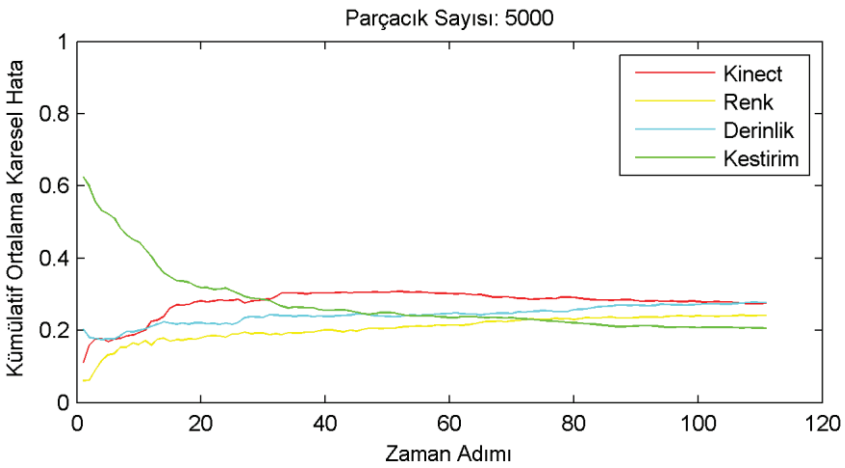
Şekil 5.23. Kümülatif Ortalama Karesel Hata Değerleri (N=500)



Şekil 5.24. Kümülatif Ortalama Karesel Hata Değerleri (N=1000)



Şekil 5.25. Kümülatif Ortalama Karesel Hata Değerleri (N=2500)



Şekil 5.26. Kümülatif Ortalama Karesel Hata Değerleri (N=5000)

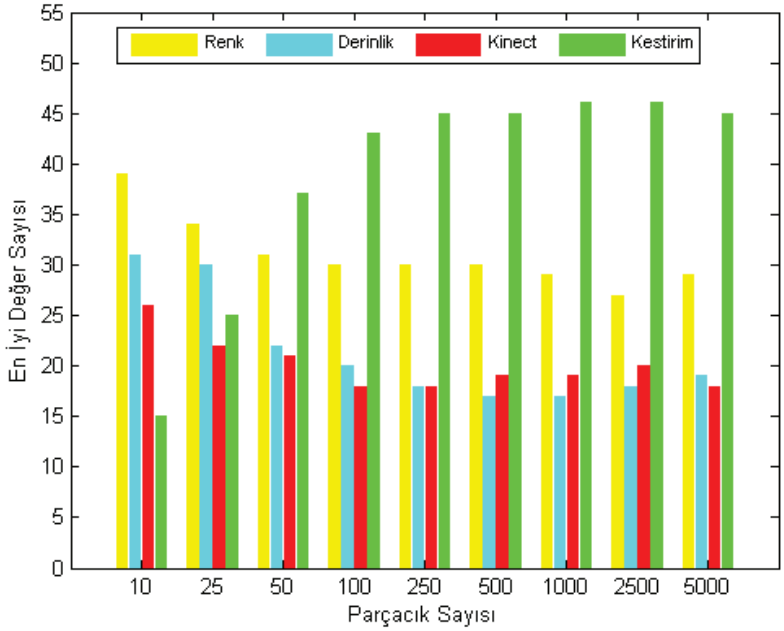
Şekil 5.25 ve 5.26 incelendiğinde parçacık sayısının 2500 ve 5000 olduğu durumlarda kümülatif ortalama karesel hata ortalamalarının Kinect cihazına göre önceki denemeler kıyasla daha iyi olduğu gözlenmektedir. Bununla birlikte, her zaman adımında her bir parçacık ile renk ve derinlik gözlem modelleri ile Kinect cihazından elde edilen koordinatlar arasındaki uzaklık hesaplamasının getirdiği hesaplama yükü ve sebep olduğu gecikme dolayısıyla parçacık sayısının 2500 ve 5000 olduğu durumların uygun çözüm sunmadığı kabul edilebilir.

5.5. Parçacık Sayısı ve Performans İlişkisi

Parçacık filtreleme ile problem çözümlerinde çözümün uygunluğunu belirleyen belli başlı etkenler bulunur. Parçacık sayısı, ilk bilgi (initial belief), dinamik hareket modeli, gözlem modeli ve bu modellere ait gürültü parametreleri bu etkenler arasında yer alır. Bu kısımda parçacık sayısın peformansa etkisi incelenmiştir.

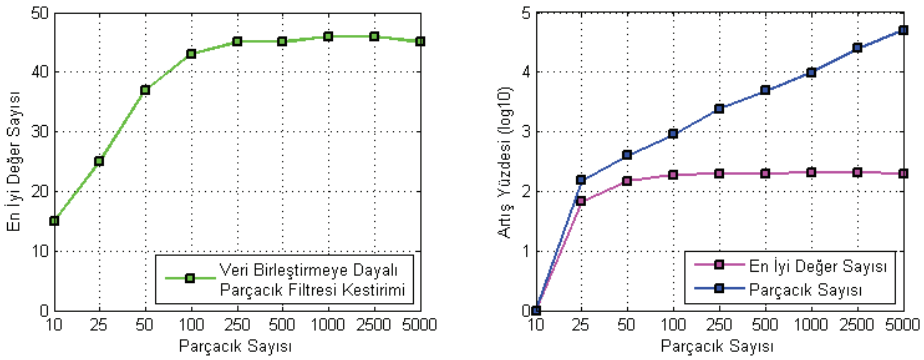
Gerçek-zamanlı hareket izleme sürecinin iyileştirilmesi probleminin çözümü için, bu çalışmada uygulanan veri birleştirmeye dayalı parçacık filtreleme uygulamasında, Bölüm 5.4'de ifade edildiği üzere 100 ile 250 arasındaki parçacık sayısının makul çözüm sunduğu kabul edilmiştir. Bu kısımda, parçacık sayısı 150 olarak belirlenmiş ve her zaman adımında renk ve derinlik gözlem modellerinden elde edilen koordinatlar ile veri birleştirmeye dayalı filtreleme ile elde edilen koordinatların, insan operatörlerin yaptığı seçimlerin ortalaması olarak kabul edilen referans koordinatların arasındaki uzaklıklar karşılaştırılmıştır. Burada, her zaman adımında bulunan en küçük uzaklık en iyi değer olarak alınmıştır. Şekil 5.27'de $N = \{10,25,50,100,250,500,1000,2500,5000\}$ parçacık sayıları için bulunan en iyi değer sayılarının grafiği verilmiştir. Bu grafik incelendiğinde, parçacık sayısı arttıkça filtreleme ile yapılan kestirimin renk, derinlik ve Kinect cihazı sonuçlarına göre daha yüksek olduğu anlaşılmaktadır. Özellikle, parçacık sayısı 50'yi aştıktan sonra en iyi sonucu sürekli olarak filtreleme ile yapılan kestirimin verdiği görülmektedir. Sonuçlara detaylı bakıldığında, ikinci en iyi değerlerin renk modeli ile elde edilen sonuçlar olduğu görülmektedir. Derinlik modeli ile elde edilen sonuçlar ve Kinect cihazından alınan sonuçların ise birbirlerine yakın seyrettiği, bazı durumlarda derinlik

modeli ile elde edilen sonuçların, bazı durumlarda ise Kinect cihazından alınan sonuçların daha iyi olduğu tespit edilmektedir.



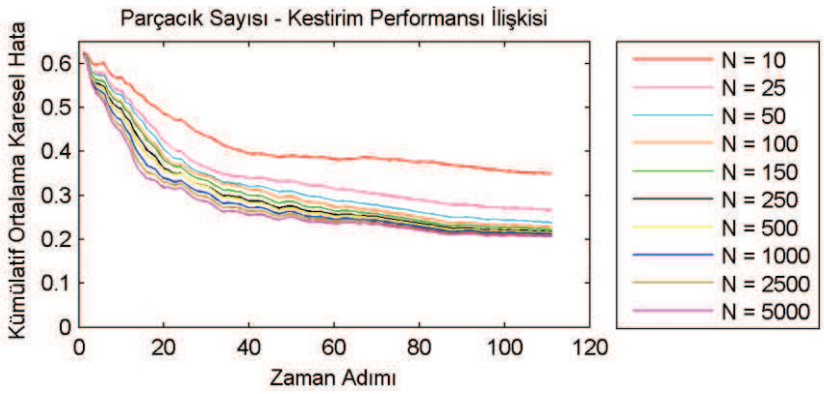
Şekil 5.27. Parçacık Sayısı Artışına Bağlı En İyi Değer Sayısı

Şekil 5.28'de parçacık sayısı artışına bağlı olarak veri birleştirmeye dayalı parçacık filtreleme kestirimi ile elde edilen en iyi değer sayısının durumu verilmiştir. Şekil 5.28'de parçacık sayısının artışına bağlı en iyi değer sayısı ile parçacık sayısı ve bulunan en iyi değer sayısındaki artış yüzdeleri logaritma ölçeğinde grafiğe dökülmüştür. Şekil 5.28'den parçacık sayısının artışının, bulunan en iyi değer sayısına aynı oranda yansımadığı ve özellikle parçacık sayısı 250'yi aştıktan sonra, bulunan en iyi değer sayısındaki artışın, ihmal edilebilir düzeyde olduğu görülmektedir.



Şekil 5.28. En İyi Değer ve Parçacık Sayısı Artış İlişkisi

Şekil 5.29'da ise, kümülatif ortalama karesel hata ortalamalarının, parçacık sayısı artışına bağlı olarak zaman adımları boyunca seyri verilmiştir. Şekil 5.29 incelendiğinde de, parçacık sayısının 100 olduğu durumdan itibaren, kümülatif ortalama karesel hata ortalamalarının büyük oranda değişmediği ve değerlerin birbirine çok yakın olduğu anlaşılmaktadır. Parçacık sayısı artışının getirdiği hesap yükü ve gecikme nedeniyle, 500 ve üzeri parçacık sayısında elde edilen çözümlerin gerçek-zaman kısıtlamasını ihlal etmesi söz konusudur.

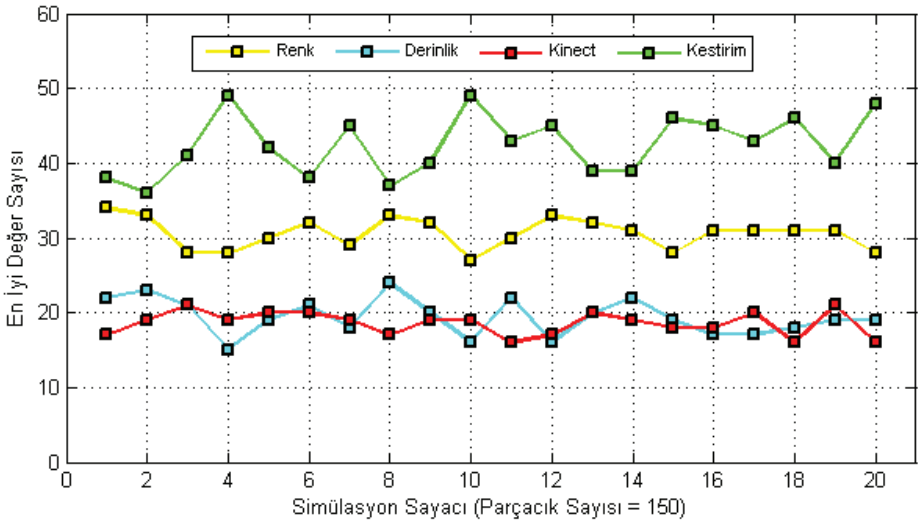


Şekil 5.29. Parçacık Sayısı ve Kestirim Performansı İlişkisi

Buradan hareketle, bu tez çalışmasında çözümü aranan problemde, uygun parçacık sayısının $100 - 250$ arasında olduğu sonucuna varılabilir.

5.6. Çoklu Çalıştırma Sonuçları

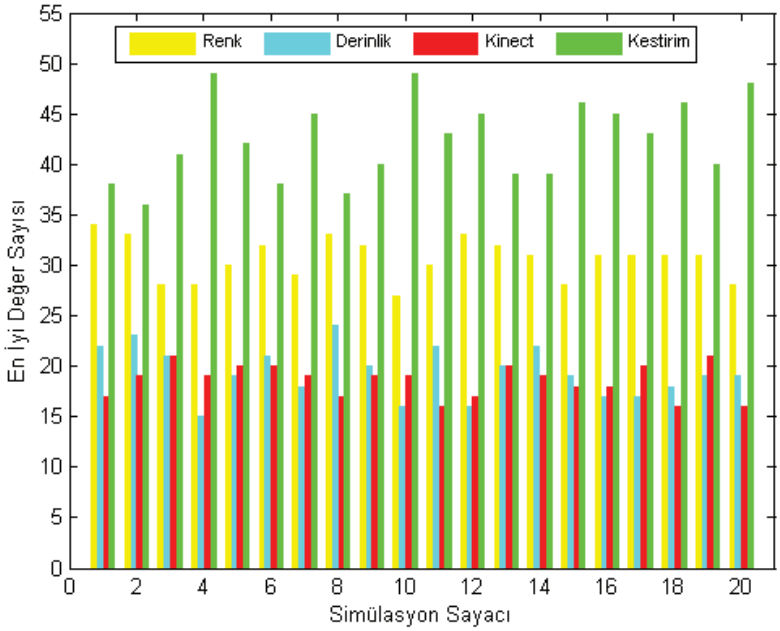
Parçacık filtreleme ile problem çözümlerinde, çözümün uygunluğunu belirleyen etkenlerden önemli bir tanesi de gürültü parametreleridir. Gürültü parametrelerinin tespit edilmesi çoğu durumuda deneysel olarak gerçekleştirilir. Bu kısımda, bölüm başında belirtilen gürültü parametrelerinin parçacık sayısı 150 olarak alındığında çözüme olan etkisi ele alınmıştır. Bu etki, yapılan uygulamanın çoklu çalıştırma sonucunda üretilen değerler üzerinden incelenmiştir. Çalışmada, yapılan uygulama 20 kez arka arkaya çalıştırılmış ve Şekil 5.30'da verilen grafik elde edilmiştir.



Şekil 5.30. En İyi Değer Sayısı Karşılaştırması (N=150)

Şekil 5.30 incelendiğinde, çoklu çalıştırma sonucunun da bir kez çalıştırmada alınan sonuçlarla uyumlu olduğu görülmektedir. Bulunan en iyi değer sayılarında, veri birleştirmeye dayalı parçacık filtreleme kestirimi ile elde edilen sonuçların en iyi olduğu, renk tabanlı model ile elde edilen sonuçların ikinci en iyi olduğu görülmektedir. Derinlik modeli ile elde edilen sonuçların bazı durumlarda Kinect cihazından alınan sonuçlardan daha iyi bazı durumlarda da kötü olduğu anlaşılmaktadır.

Şekil 5.31'de 20 çalıştırmadaki tüm en iyi değer sayılarının grafiği verilmiştir. Grafik detaylı olarak incelendiğinde, filtreleme kestirimi ile elde edilen sonuçların tüm simülasyonlarda 111 zaman adımının en az 35'inde en iyi değerleri ürettiği görülmektedir. Renk modeli sonuçları, 25 ile 30 arasında en iyi değerler üretirken, derinlik modeli ve Kinect cihazı ortalama $18-20$ aralığında en iyi değerlere sahiptir. Şekil 5.31'de ayrıca, farklı çalıştırmalarda renk ve derinlik modelleri ile filtreleme kestirimi sonuçlarının farklı değerler ürettikleri görülmektedir. Parçacık filtreleme yapısı içerisindeki dinamik hareket ve gözlem modelleri için kullanılan rassal gürültü parametreleri dolayısıyla keyfi gürültü değerleri oluştuğundan bu sonucun ortaya çıkması doğaldır. Burada, Kinect cihazından alınan sonuçların da renk ve derinlik modelleri ile filtreleme kestirimi ile elde edilen sonuçlardaki değişimin etkisiyle farklı çalıştırmalarda farklı olduğu, ancak diğerlerine göre birbirine daha yakın değerler ürettiği görülmektedir.

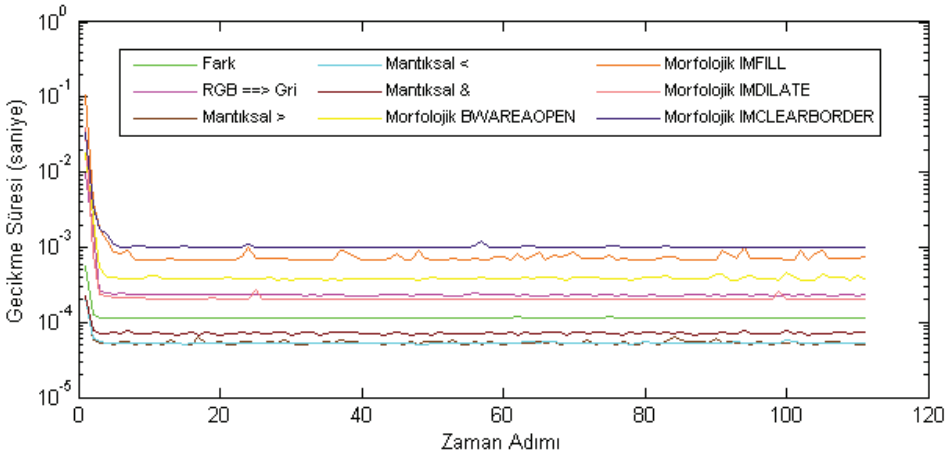


Şekil 5.31. 20 Çalıştırmadaki En İyi Değer Sonuçları (N=150)

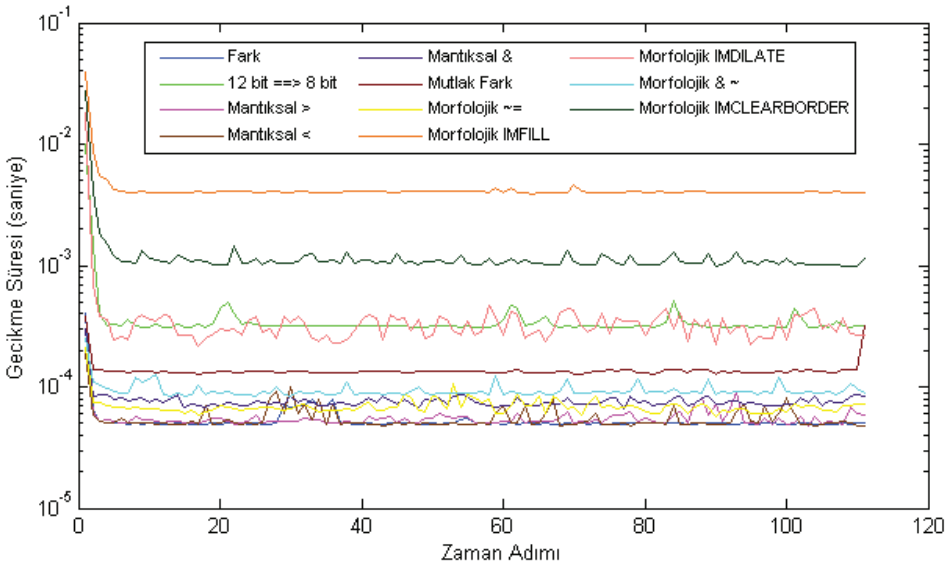
5.7. Gecikme Zamanları

Bu çalışmada, hareket izleme sürecinin gerçek-zamanlı olarak iyileştirilmesi hedeflendiği için, veri birleştirmeye dayalı parçacık filtreleme kestiriminin gözlem modelinde uygulanan ara işlemlerin meydana getirdiği gecikmenin de belli bir düzeyde olması gerekmektedir. Bu kısımda, renk ve derinlik gözlem modellerinin önce ayrı ayrı, sonra da birlikte meydana getirdiği gecikme süreleri incelenmiştir.

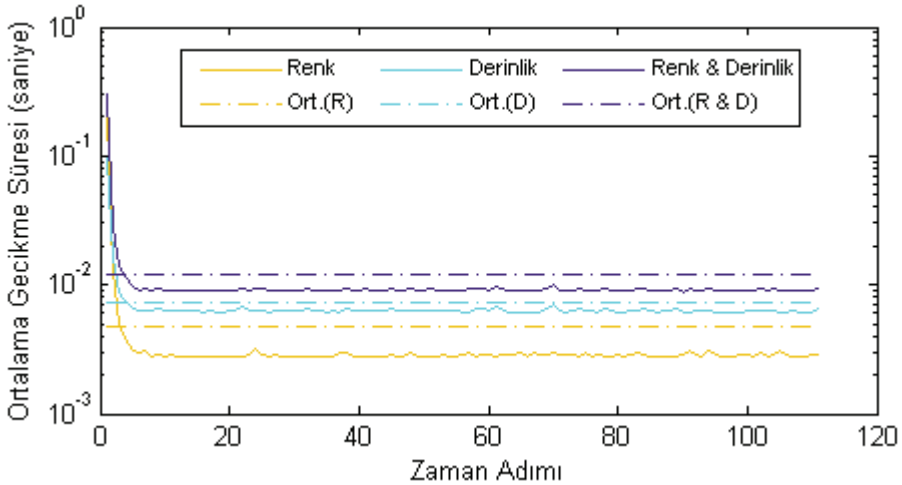
Şekil 5.32 ve 5.33'de sırasıyla renk ve derinlik tabanlı gözlem modellerinde, her bir zaman adımında yapılan işlemlerin yarattığı gecikmeler, saniye cinsinden verilmiştir. Şekil 5.34'de ise her iki gözlem modelinin birlikte oluşturduğu gecikmeler gösterilmiştir. Şekil 5.34 incelendiğinde, ortalama gecikme süresinin 10 milisaniye seviyesinde olduğu anlaşılmaktadır. Bu noktadan hareketle, bu çalışmada uygulanan veri birleştirmeye dayalı parçacık filtreleme kestiriminin, 1 karenin işlenmesi için kullanılabilir sürenin 33 milisaniye olduğu (30 fps için) gerçek-zamanlı hareket izlemeye engel oluşturmadığı sonucu ortaya çıkmaktadır.



Şekil 5.32. Renk İpucu Bulma İşlemlerinin Oluşturduğu Gecikme



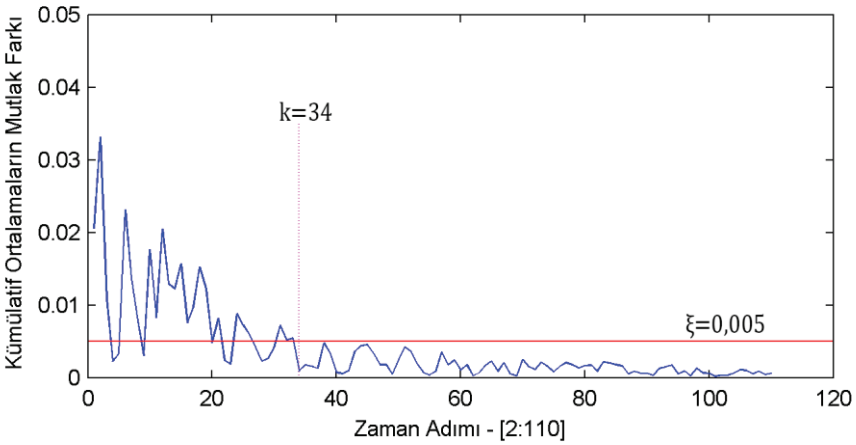
Şekil 5.33. Derinlik İpucu Bulma İşlemlerinin Oluşturduğu Gecikme



Şekil 5.34. İpucu İşlemlerinin Oluşturduğu Ortalama Gecikme

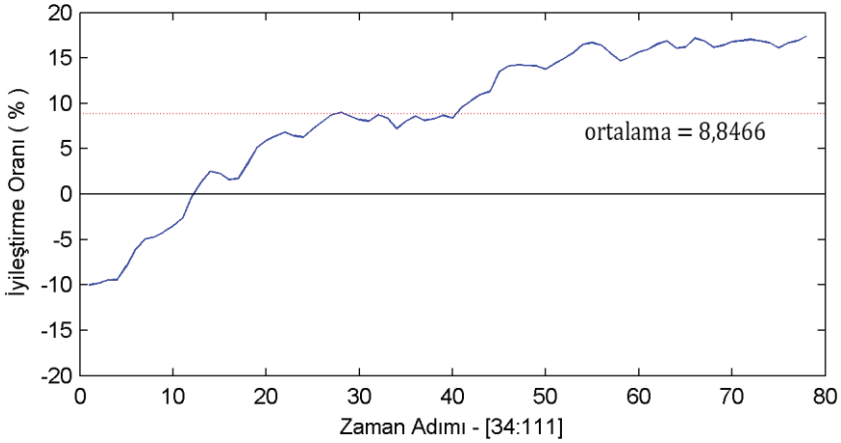
5.8. İyileştirme Oranları

Hareket izleme sürecinin gerçek-zamanlı olarak iyileştirilmesi uygulamasında, veri birleştirmeye dayalı parçacık filtreleme kestiriminin, çakışma durumlarında, Kinect cihazının iç-tanımlı algoritmalarıyla ürettiği sonuçlara göre daha iyi sonuçlar ürettiği ve 150 parçacığın doğruluk seviyesi ve gecikme süresi bakımılarından kabul edilebilir bir iyileştirme için yeterli olduğu bu bölümde verilen karşılaştırma grafikleriyle ortaya konulmuştur. Uygulamada, 150 parçacık için yapılan iyileştirmenin sayısal değerini tespit etmek amacıyla, ayrık zamanlı olay simülasyonu için kullanılan, literatürdeki istatistiksel proses kontrolü yöntemlerinden Kelton ve Law tarafından geliştirilen yöntem benimsenmiştir [58]. Bu yöntemde, çoklu çalıştırma sonucunda elde edilen normalize edilmiş sonuçların kümülatif ortalamaları incelenir. Ardışık zaman adımlarında, kümülatif ortalamadaki mutlak değişim belli bir eşik değerinin altına düştüğünde, kullanılan kestirimcinin o zaman adımından itibaren kararlı olduğu kabul edilir. Kestirimcinin kararlı olduğu zaman adımından önceki zaman adımları ise simülasyon ısınma zamanı olarak tanımlanır. Bu çalışmada da, 150 parçacık için parçacık filtreleme kestirimi 20 kez arka arkaya çalıştırılmış ve her bir deneme için elde edilen kümülatif ortalama karesel hatadaki mutlak değişim (ξ), % 0.5 olarak alınmış ve buna bağlı olarak ilk 33 zaman adımını simülasyon ısınma zamanı olarak belirlenmiştir. (Şekil 5.35)



Şekil 5.35. Simülasyon Isınma Zamanı

Yapılan iyileştirmenin sayısal değerinin tespiti için, 20 çalıştırmadaki ortalama karesel hata değerlerinin zaman adımı bazında ortalaması alınarak 34. zaman adımından itibaren her bir zaman adımı için parçacık filtresi kestiriminin Kinect cihazına göre oranı bulunmuştur. Bu oranların ortalaması ise iyileştirme değeri olarak hesaplanmıştır. Sonuçta, Şekil 5.36'da görüldüğü üzere, 150 parçacık için veri birleştirmeye dayalı parçacık filtresi kestiriminin, Kinect cihazının ürettiği hata değerlerine göre yaklaşık % 8.84 oranında daha iyi olduğu belirlenmiştir.



Şekil 5.36. SF-PF Kestiriminin İyileştirmesi

BÖLÜM 6. SONUÇ VE ÖNERİLER

Lineer olmayan durum kestirimi, mühendislik, doğa bilimleri ve hatta sosyal bilim dallarına ait çok sayıdaki araştırma alanında üzerinde çalışılan bir problemdir. İnsan hareketi izleme problemi de lineer olmayan durum geçişi ve normal dağılıma uymayan süreç ve ölçüm gürültüsü içerebilen bir gerçek-dünya problemi olarak çok sayıda uygulama için çözülmesi gerekli bir problemdir. Uygun dinamik ve gözlem modelleri kullanılması durumunda, kabul edilebilir yaklaşık çözümler getirme potansiyeli olduğundan, yapılan bu çalışmada da kullanılan Bayesçi kestirime dayalı parçacık filtreleme, son yıllarda giderek sık başvurulan bir yöntem haline gelmiştir. Çalışmada, Bayesçi kestirimin teorik temelleri matematiksel çıkarımlarıyla birlikle sunularak, sonraki araştırmacıların kestirim problemlerini modellemeleri için bir temel oluşturulmuştur. Biri örnek bir diğeri de gerçek-dünya problemi olmak üzere iki problem Parçacık Filtreleme yöntemiyle çözülerek yorumlanmış, böylece teorik yaklaşımın uygulamaya nasıl geçirileceği ile ilgili alan araştırmacılarına pratik bir başvuru kaynağı sağlanmıştır.

Microsoft firmasının insan-bilgisayar etkileşimi için tasarladığı Kinect cihazının kullanıldığı bu çalışmada, sabit aydınlatmalı ve karmaşık arka plana sahip kapalı bir ortamda, sahnede hareket eden insan deneğin el avuç içi merkezinin gerçek-zamanlı takibi uygulaması yapılmıştır. Üzerinde RGB ve kızılötesi olmak üzere iki adet kamera bulunduran Kinect cihazı, görüntülediği insan deneğe ait eklem noktalarını 3 boyutlu olarak gerçek-zamanda takip ederek, özellikle harekete duyarlı oyun ve bilgisayar kontrolü uygulamaları için imkân sağlamaktadır. Kinect cihazının özellikle, deneğin vücut kısımlarının birbiriyle çakışması durumlarında, iç-tanımlı algoritmalarıyla hatalı sonuçlar üretmesi ve dolayısıyla hareket izlemeye bağlı uygulamalarda verimsizliğe yol açması, çakışma durumlarında daha başarılı olabilecek bir hareket izleme uygulaması yapma hususunda motivasyon sağlamıştır.

Yapılan uygulamada, sahnedeki insan deneğin hareketi, Gauss gürültüsü içeren sabit hızlı hareket şeklinde modellenerek, izlenen el avuç-içi merkezinin her bir zaman adımında görüntü imgesindeki konumu tahmin edilmiştir. Gözlem modeli olarak, imge işleme yoluyla elde edilen renk ve derinlik tabanlı ipuçları ile Kinect cihazının ürettiği eklem noktası koordinatları birlikte kullanılmıştır. Kinect cihazının ürettiği sonuçlar yaklaşık değer olarak kabul edilmiş, renk ve derinlik ipuçları, bu sonuçlar etrafındaki bir bölgede aranarak çözüm uzayı daraltılmıştır. Ayrıca, izlemede renk, derinlik ve Kinect tabanlı olarak ayrı ayrı elde edilen ipuçları üzerinde veri birleştirme uygulanarak, kestirim için her bir ipucunun müstakil olabilirliğine göre daha fazla bilgi içeren ve dolayısıyla daha doğru izleme sonuçları vermesi beklenen ortak olabilirlik elde edilmiştir. Gözlem modeliyle her bir zaman adımında elde edilen ortak olabilirlik ile o zaman adımına ait parçacık kümesinin ağırlıkları güncellenmiştir. Çalışmada, Parçacık filtreleme SIR algoritması uygulanmış ve artık terimli yeniden örnekleme yöntemiyle, ağırlığı daha yüksek olan parçacıkların toplam parçacıklar içindeki oranı her yeni zaman adımında artırılarak kestirimin en yüksek ağırlıklı parçacıklar arasından yapılması sağlanmıştır.

Uygulamada, 7 farklı insan operatörün 111 zaman adımında insan deneğe ait imge dizisi üzerinde belirlediği el avuç içi merkezi koordinatlarının ortalama değeri referans veri olarak kullanılmıştır. Renk, derinlik ve Kinect tabanlı ipuçları ile veri birleştirmeye dayalı parçacık filtreleme kestirimi ile elde edilen simülasyon sonuçları, referans veriye bağlı ortalama karesel hata değerleriyle birlikte verilmiştir. Ayrıca, başlangıç ve gürültü parametreleri sabit tutularak parçacık sayısı artırılmak suretiyle, parçacık sayısının başarım performansına olan etkisi incelenmiştir. Elde edilen sonuçlarda, parçacık sayısının 150 civarında olmasının bu çalışmada yapılan uygulama için yeterli olduğu, parçacık sayısının daha fazla artırılmasının getireceği işlem yükü dolayısıyla olumsuz etki edeceği tespit edilmiştir. Çalışmada daha sonra, makul sonuçlar veren 150 parçacık sabit tutularak, uygulamanın çoklu çalıştırma sonuçları elde edilmiştir. Uygulama peş peşe 20 kez çalıştırılarak elde edilen sonuçlar grafiklerle sunulmuş ve veri birleştirmeye dayalı parçacık filtreleme kestirimi ile elde edilen sonuçların renk, derinlik ve Kinect tabanlı sonuçlardan daha iyi olduğu gösterilmiştir. Son olarak, çalışmada uygulanan gözlem modeli içindeki işlemlerin meydana getirdiği gecikmeler işlem bazında ve toplam olarak hesaplanarak

görselleştirilmiştir. Bu çalışmada, gözlem modelinin toplamda saniyenin yüzde biri kadar bir gecikmeye sebep olduğu tespit edilmiştir. Saniyede 25 kare ve üstünün gerçek-zamanlı işlemler için sınır kabul edildiği düşünüldüğünde, bu gecikmenin gerçek-zamanlı hareket izleme süreci için olumsuz bir durum oluşturmayacağı söylenebilir.

Bu tez çalışmasında, veri birleştirmeye dayalı parçacık filtreleme konusunda edinilen deneyimler kullanılarak, Kinect tabanlı bir fizik-tedavi uygulaması, Türk işaret dili alfabesinin tanınması ve anlatımının gerçek-zamanlı olarak metne çevrilmesi ve programlanabilir maket hava aracı ile yön tayini ve engel içeren güzergâh üzerinden iniş yapılması gibi çalışmaların yapılması planlanmaktadır. Ayrıca, bu çalışmada parçacık filtreleme ile gerçek-zamanlı hareket izleme probleminin çözümü için oluşturulan teorik temel ve yapılan uygulamaların gözetleme, animasyon ve kontrol, sanal ve artırılmış gerçeklik ile performans analizi gibi alanlarda araştırmacılara benzer uygulamalar yapma imkânı sağlayacağı öngörülmektedir.

KAYNAKLAR

[1] DEVLAEMINCK, R., Human Motion Tracking With Multiple Cameras Using A Probabilistic Framework For Posture Estimation, Y.Lisans Tezi, Purdue University, 2006.

[2] GREWAL, M. S.; ANDREWS, A. P., Kalman filtering: theory and practice using MATLAB, John Wiley & Sons, 2011.

[3] http://sci.esa.int/rosetta, Erişim Tarihi: 18.11.2014.

[4] JAZWINSKI, A. H., Stochastic Processes and Filtering Theory, Courier Dover Publications, 2007.

[5] DEMİRBAŞ, K., Real-time Recursive State Estimation for Nonlinear Discrete Dynamic Systems with Gaussian or non-Gaussian Noise, Discrete Time Systems, Part, 1: 3-18, 2011.

[6] MAYBECK, P. S., Stochastic Models, Estimation, and Control. Vol. 1, Academic Press, 1979.

[7] JUNG, B., SUKHATME, G. S., Real-time Motion Tracking from a Mobile Robot, International Journal of Social Robotics, 2.1: 63-78, 2010.

[8] SIOURIS, G. M., Missile Guidance and Control Systems, Springer, 2004.

[9] WILLIAMS, S. B., et al., Autonomous Underwater Navigation and Control, Robotica, 19.05: 481-496, 2001.

[10] GAGALOWICZ, A., QUAH, C. K., 3D Model-based Marker-less Human Motion Tracking in Cluttered Environment, In Proc. Computer Vision Workshops (ICCV), 12th International Conference on. IEEE, p. 1042-1049, 2009.

[11] POPPE, R., Vision-based Human Motion Analysis: An Overview, Computer Vision and Image Understanding, 108.1: 4-18, 2007.

[12] FRIGOLA, R., RASMUSSEN, C. E., Integrated Pre-processing for Bayesian Nonlinear System Identification with Gaussian Processes, In: Decision and Control (CDC), 52nd Annual Conference on. IEEE, p. 5371-5376, 2013.

[13] MÜNDERMANN, L., et al., Measuring Human Movement for Biomechanical Applications Using Markerless Motion Capture, In: Electronic Imaging 2006. International Society for Optics and Photonics, p. 60560R-60560R-10, 2006.

[14] MOESLUND, T. B., GRANUM, E., A Survey of Computer Vision-based Human Motion Capture, Computer Vision and Image Understanding, 81.3: 231-268, 2001.

[15] SIGAL, L., BLACK, M. J., Humaneva: Synchronized Video and Motion Capture Dataset for Evaluation of Articulated Human Motion, Brown University, 2006.

[16] KAPUR, A., et al. A Framework for Sonification of Vicon Motion Capture Data, In: Conference on Digital Audio Effects, p. 47-52, 2005.

[17] CHEN, S., LI, Y., KWOK, N. M., Active Vision in Robotic Systems: A Survey of Recent Developments, The International Journal of Robotics Research, 30.11: 1343-1377, 2011.

[18] DEUTSCHER, J., BLAKE, A., REID, I., Articulated Body Motion Capture by Annealed Particle Filtering, In: Computer Vision and Pattern Recognition, Proceedings. p. 126-133, 2000.

[19] SEPEHRI, A., Visual Tracking of Human Hand and Head Movements and its Applications, 2007.

[20] HECHT, F., AZAD, P., DILLMANN, R., Markerless Human Motion Tracking with a Flexible Model and Appearance Learning, In: Robotics and Automation, 2009. ICRA'09. IEEE International Conference on. IEEE, p. 3173-3179, 2009.

[21] BROX, T., ROSENHAHN, B., CREMERS, D., Contours, Optic Flow, and Prior Knowledge: Cues for Capturing 3D Human Motion in Videos, In: Human Motion. Springer Netherlands, p. 265-293, 2008.

[22] AZAD, P., et al., Image Based Markerless 3D Human Motion Capture using Multiple Cues, In: International Workshop on Vision Based Human-Robot Interaction, Palermo, Italy. 2006.

[23] MOESLUND, T. B.; HILTON, A. KRÜGER, V., A Survey of Advances in Vision-based Human Motion Capture and Analysis, Computer Vision and Image Understanding, 104.2: 90-126, 2006.

[24] MOESLUND, T. B., GRANUM, E., A Survey of Computer Vision-based Human Motion Capture, Computer Vision and Image Understanding, 81.3: 231-268, 2001.

[25] SIAPAS, A. G., A Global Approach to Parameter Estimation of Chaotic Dynamical Systems, MASSACHUSETTS INST OF TECH CAMBRIDGE ARTIFICIAL INTELLIGENCE LAB, 1992.

[26] SÄRKKÄ, S., Bayesian Filtering and Smoothing, Cambridge University Press, 2013.

[27] SENGIJPTA, S. K., Fundamentals of Statistical Signal Processing: Estimation Theory, Technometrics, 37.4: 465-466, 1995.

[28] SABERI, A., STOORVOGEL, A. A., SANNUTI, P., Filtering Theory: with Applications to Fault Detection, Isolation, and Estimation, Springer, 2007.

[29] ALDRICH, J., et al., RA Fisher and the Making of Maximum Likelihood 1912-1922, Statistical Science, 12.3: 162-176, 1997.

[30] MYUNG, J., Tutorial on Maximum Likelihood Estimation, Journal of Mathematical Psychology, 47.1: 90-100, 2003.

[31] M. S. GREWAL, A. P. ANDREWS, Applications of Kalman Filtering in Aerospace 1960 to the Present, Control Systems, IEEE, 30(3), 69, 2010.

[32] WELLING M., The Kalman Filter, Caltech Class Notes, Tech. Rep., Jan. 2000, http://www.ics.uci.edu/~welling, Erişim Tarihi: 10.09.2014.

[33] G. WELCH, G. BISHOP, An Introduction to the Kalman Filter, UNC-Chapel Hill, Tech. Rep., TR 95-041, Jul. 2006.

[34] RIBEIRO, M. I., Kalman and Extended Kalman Filters: Concept, Derivation and Properties, Institute for Systems and Robotics, 43, 2004.

[35] CHEN, W-s., Bayesian Estimation by Sequential Monte Carlo Sampling for Nonlinear Dynamic Systems, Doktora Tezi, The Ohio State University, 2004.

[36] GORDON, N., RISTIC, B., ARULAMPALAM, S., Beyond the Kalman Filter: Particle Filters for Tracking Applications, Artech House, London, 2004.

[37] ARULAMPALAM, M. S., et al., A Tutorial on Particle Filters for Online Nonlinear/non-Gaussian Bayesian Tracking, Signal Processing, IEEE Transactions on, 50.2: 174-188, 2002.

[38] DOUCET, A., GODSILL, S., ANDRIEU, C., On Sequential Monte Carlo Sampling Methods for Bayesian Filtering, Statistics and Computing, 10.3: 197-208, 2000.

[39] HAUG A. J., A Tutorial on Bayesian Estimation and Tracking Techniques Applicaple to Nonlinear and Non-Gaussian Processes, The MITRE Corporation, USA, Tech. Rep., 05W0000004, Jan. 2005.

[40] DJURIC, P. M., et al., Particle Filtering, Signal Processing Magazine, IEEE, 20.5: 19-38, 2003.

[41] TEREJANU, G., et al., A Novel Gaussian Sum Filter Method for Accurate Solution to the Nonlinear Filtering Problem, In: Information Fusion, 2008 11th International Conference on. IEEE, p. 1-8, 2008.

[42] CAPPÉ, O., GODSILL, S. J., MOULINES, E., An Overview of Existing Methods and Recent Advances in Sequential Monte Carlo, Proceedings of the IEEE, 95.5: 899-924, 2007.

[43] DONG W., et al., Particle Filtering Algorithm Based on Recursive Bayesian Estimation Using Gaussian Sum in WSN, Journal of Information & Computational Science, 9(2), 377-486, 2012.

[44] A. E. WAN, R. Van Der MERWE, The Unscented Kalman Filter for Nonlinear Estimation, Proceedings of Symposium 2000 on Adaptive Systems for Signal Processing, Communication and Control (AS-SPCC), IEEE, Lake Louise, Alberta, Canada, pp. 153–158, 2000.

[45] CHEN, Z., Bayesian Filtering: From Kalman Filters to Particle filters, and Beyond, Statistics, 182.1: 1-69, 2003.

[46] AZIMI-SADJADI, B., KRISHNAPRASAD, P. S., Approximate Nonlinear Filtering and its Application in Navigation, Automatica, 41.6: 945-956, 2005.

[47] RICHEY, M., The Evolution of Markov Chain Monte Carlo Methods, The American Mathematical Monthly, 117.5: 383-413, 2010.

[48] GORDON, N. J., SALMOND, D. J., SMITH, A. FM., Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation. In: IEE Proceedings F (Radar and Signal Processing). IET Digital Library, p. 107-113, 1993.

[49] ISARD M., BLAKE A., Condensation—Conditional Density Propagation for Visual Tracking, Intl. J. of Computer Vision, 29(1), 1998.

[50] VAN DER MERWE, R., et al., The Unscented Particle Filter, In: NIPS, p. 584-590, 2000.

[51] ANDERSON, J. L., ANDERSON, S. L., A Monte Carlo Implementation of the Nonlinear Filtering Problem to Produce Ensemble Assimilations and Forecasts, Monthly Weather Review, 127.12: 2741-2758, 1999.

[52] CASELLA G., ROBERT C. P., Monte Carlo Statistical Methods, Biometrika, 83(1), 81-94, 1996.

[53] SEBASTIAN, P., VOON, Y. V., COMLEY, R., Performance Evaluation Metrics for Video Tracking, IETE Technical Review, 28(6), 2011.

[54] LI, X. R., JILKOV, V. P., Survey of Maneuvering Target Tracking: Dynamic Models, In: AeroSense 2000. International Society for Optics and Photonics, p. 212-235, 2000.

[55] HOL, J. D., SCHON, T. B., GUSTAFSSON, F., On Resampling Algorithms for Particle Filters, In: Nonlinear Statistical Signal Processing Workshop, IEEE, p. 79-82, 2006.

[56] DOUC, R., CAPPÉ, O., Comparison of Resampling Schemes for Particle Filtering, In: Image and Signal Processing and Analysis, 2005. ISPA 2005. Proceedings of the 4th International Symposium on. IEEE, p. 64-69, 2005.

[57] http://msdn.microsoft.com/en-us/magazine/jj159883.aspx, Erişim Tarihi: 20.11.2014.

[58] ROBINSON, S., A Statistical Process Control Approach to Selecting a Warm-up Period for a Discrete-event Simulation, European Journal of Operational Research, 176.1: 332-346, 2007.

[59] ZHAI, Y. Improved Nonlinear Filtering for Target Tracking, Doktora Tezi, Proquest Tez Veritabanı (UMI No. 3256650), 2007.

[60] WANG, J., Reliable and Efficient Tracking of Human Motion Using Particle Filtering, Doktora Tezi, 2008, Proquest Tez Veritabanı (UMI No. 3351568).

[61] VENKATARAMAN, V., Advanced Machine Learning Approaches for Target Detection, Tracking and Recognition, Doktora Tezi, Proquest Tez Veritabanı (UMI No. 3443606), 2010.

[62] VELMURUGAN, R., Implementation Strategies for Particle Filter Based Target Tracking, Doktora Tezi, Proquest Tez Veritabanı (UMI No. 3261730), 2007.

[63] SIDENBLADH, H., Probabilistic Tracking and Reconstruction of 3D Human Motion in Monocular Video Sequences, Doktora Tezi, 2001, http://www.csc.kth.se, Erişim Tarihi: 15.08.2014.

[64] SAHA, S., Topics in Particle Filtering and Smoothing, Doktora Tezi, 2009, http://eprints.eemcs.utwente.nl, Erişim Tarihi: 25.07.2014.

[65] RYU, H., Multiple Object Tracking Using Particle Filters, Doktora Tezi, Proquest Tez Veritabanı (UMI No. 1435942), 2006.

[66] PUNITHAKUMAR, K., Nonlinear Filtering Algorithms for Multitarget Tracking, Doktora Tezi, McMaster University, 2008.

[67] ÖZKAN, E., Particle Methods for Bayesian Multi-Object Tracking and Parameter Estimation, Doktora Tezi, 2009, http://theses.eurasip.org, Erişim Tarihi: 20.06.2014.

[68] NEDDERMEYER, J. C., Sequential Monte Carlo Methods for General State-Space Models, Doktora Tezi, 2006, http://www.rzuser.uni-heidelberg.de, Erişim Tarihi: 08.08.2014.

[69] LMACCORMICK, J., Probabilistic Modelling and Stochastic Algorithms for Visual Localisation and Tracking, Doktora Tezi, 2000, http://users.dickinson.edu/~jmac/publications/thesis.pdf, Erişim Tarihi: 11.03.2014.

[70] LEE, D.-J., Nonlinear Bayesian Filtering With Applications To Estimation And Navigation, Doktora Tezi, 2005, http://repository.tamu.edu, Erişim Tarihi: 10.09.2014.

[71] KAYABOL, K., İmge Kaynaklarının Ayrılmasında Bayesçi Yaklaşımlar, Doktora Tezi, 2008, http://theses.eurasip.org, Erişim Tarihi: 24.04.2014.

[72] KARLSSON, R., Particle Filtering for Positioning and Tracking Applications, Doktora Tezi, 2005, http://www.control.isy.liu.se, Erişim Tarihi: 02.02.2014.

[73] ISARD, M. A., Visual Motion Analysis by Probability Propagation of Conditional Density, Doktora Tezi, 1998, http://www.robots.ox.ac.uk/~misard, Erişim Tarihi: 05.07.2014.

[74] HA, J.-C., Real-Time Visual Tracking Using Image Processing and Filtering Methods, Doktora Tezi, Proquest Tez Veritabanı (UMI No. 3364212), 2008.

[75] GENÇAĞA, O. D., Sequential Bayesian Modeling of Non-Stationary Non-Gaussian Processes, Doktora Tezi, 2007, http://www.ee.boun.edu.tr, Erişim Tarihi: 18.01.2014.

[76] CHENG, H.-Y., Multi-object Tracking via Particle Sampling and Enhanced Probabilistic Data Association for Event Detection in Intelligent Video Systems, Doktora Tezi, University of Washington, 2008.

[77] CHEN, W-s., Bayesian Estimation by Sequential Monte Carlo Sampling for Nonlinear Dynamic Systems, Doktora Tezi, The Ohio State University, 2004.

[78] BOLIĆ, M., Architectures for Efficient Implementation of Particle Filters, Doktora Tezi, 2004, http://www.site.uottawa.ca, Erişim Tarihi: 15.06.2014.

[79] BERGMAN, N., Recursive Bayesian Estimation: Navigation and Tracking Applications, Doktora Tezi, 1999, http://linkinghub.elsevier.com, Erişim Tarihi: 13.04.2014.

EKLER

EK 1. Temel Olasılık Teorisi Kavramları

Değişkenlerinin değerleri bilinen problemlere deterministik yöntemler kullanılarak çözüm bulunabilir, ancak rassal değişkenlere sahip problemlerin çözümü olasılıksal modeller (zamanla değişen istatistiksel parametreleri olan modeller) kurularak çözülebilir. Aslında doğada karşılaşılan ve mühendisliğin çeşitli disiplinlerinin ele aldığı problemlerin birçoğu karmaşık sistemlerdir ve ancak olasılıksal modellerden yararlanılarak çözüme kavuşturulabilir. Bilgisayar Görmesi (Computer Vision) alanındaki İnsan hareketinin yakalanması ve izlenmesi problemi de deterministik yöntemlerle çok özel koşullar haricinde analitik çözümü elde edilemeyen problemler arasındadır. Bu ve benzer problemlerin olasılıksal modelleri oldukça karmaşıktır ve etkin bir model oluşturabilmek için olasılık ve olasılıksal yaklaşımların iyi bir şekilde anlaşılması gerekir. Bu bağlamda, çalışmanın bu kısmında olasılıkla ilgili temel kavramlar ele alınarak, problemlerin olasılık tabanlı yöntemlerle nasıl çözülebileceği anlatılacaktır.

1.1. Örnek Uzay, Rassal Olay, Rassal Değişken

Olasılık Teorisi, belirsizliğin matematiksel modelinin oluşturulması ile ilgilenen bir bilim dalıdır. Bu bağlamda, bir olayın meydana gelme sıklığını ifade eden istatistikler, olaya ilişkin eldeki kesin olmayan veriler ve olaya ait önermelerin doğruluğuna ilişkin inançlar gibi kaynaklardan yararlanılarak çözüme ulaşılmaya çalışılır.

Örnek uzay, belirli bir deneyin muhtemel tüm sonuçlarını içeren kümeye denir. Örneğin, iki madeni paranın atılması deneyi için örnek $S = \{TT, TY, YT, YY\}$ (T: Tura & Y: Yazı) şeklinde ifade edilir.

Rassal değişken (random variable), gelecekteki bir deneyde alacağı değer kesinlikle bilinmeyen bir değişkendir. Bir madeni paranın atılmasıyla, hangi yüzünün üste geleceği bilinemeyeceğinden bu rassal bir değişken olarak düşünülmelidir.

Örnek. 2016 olimpiyatlarına katılan ve en fazla madalya alan 20 ülkenin bulunduğu bir listeden (L) rastgele bir ülke seçilsin:u. Rassal değişken M bu ülkenin aldığı madalya sayısı olsun.

$M \to L: \mathbb{N}, u \mapsto M(u) = u'$nun aldığı madalya sayısı.

$\mathrm{P}(M = \text{"Türkiye"}) = 0.05$ (Türkiye'nin bu ülkelerden birisi olma olasılığı)

Rassal değişkenlerin belirsizliği, bu değişkenlerin alacağı değerlerin önceden tahmin edilemeyen çok sayıda etkene bağlı olmasından kaynaklanır. Bu etkenler, olayın kendi dinamikleri ile ilgili olabileceği gibi olayla ilgili bilgi eksikliği de olabilir. Rassal değişkenlerin deterministik yaklaşımlarla incelenmesi mümkün değildir, ancak olasılıksal yaklaşımlar kullanılarak bu değişkenler incelenebilir.

Bir rassal değişkeninin gelecekteki bir deney sırasında alacağı değer bilinemez, bununla birlikte deneyin sonuçlanmasıyla birlikte değişken bir değer alır ve buna rassal olay denir. Rassal olay aslında, deneye ait muhtemel sonuçları içeren örnek uzayın alt kümelerinden birisidir. İki madeni paranın atılması deneyinde, her iki paranın birbirinden farklı gelmesi olayı $\Theta = \{YT, TY\}$ şeklinde ifade edilir.

Bir rassal olayın olacağı önceden bilinmemekle birlikte, bu olayın görülme olasılığı hesaplanabilir. Örneğin, bir zar atıldığında, zarın üst yüzünde görülen sayının 1 ila 6 arasında bir sayı olacağı bellidir. Ancak zarın üst yüzünde görülen sayının 2 olması rassal bir olaydır.

Olasılık teorisinde, bir rassal olayın meydana gelme şansı 0 ila 1 arasında bir değerle ifade edilir ve buna olasılık (ihtimal) adı verilir. Rassal olayın hiçbir şekilde karşılaşılmayacak bir olay olması o olayın ihtimalini 0 yaparken, kesin olarak

karşılaşılacak bir olay olması da ihtimali 1 yapacaktır. Deney sayısı arttıkça, rassal olayların görülme sıklığı da artacaktır.

Rassal bir değişken X ile, bu değişkenin bir deney sonucunda aldığı değer de x ile ifade edilecek olursa, rassal olayın olasılığı $p(X = x_i) = p(x_i)$ şeklinde olacaktır. Bütün rassal olayların olasılıkları toplamı ise 1'e eşit olacaktır (Bir madeni paranın atılması olayının sonucu % 50 ihtimalle ya yazı ya da tura olacaktır.

1.2. Rassal Değişken Dağılımları

Bir rassal değişken ve bu değişkenin içinde bulunduğu topluluğa ait rasgele olaylar toplu olarak bir dağılım fonksiyonu ile ifade edilebilir. Birçok rassal değişkenin alacağı değerler belirli bir sınır içinde olsa da, sürekli değişmesi dolayısıyla rassal olayların örnek uzayı sonsuz sayıda elemana sahip olacaktır. Örneğin, bir zar atıldığında 1 ila 6 arasında bir sayı gelecektir, ancak zarın her atılışında gelecek olan sayı değişecektir, yani her bir sayının gelme olasılığının hesaplanabilmesi için zar sonsuz kez atılmalıdır. Ancak pratikte, deneyler belirli bir sayıda ölçümler de belirli bir hassasiyetle yapılabileceğinden rassal olayların olasılıkları rassal değişkenlerin ayrık ya da sürekli olması durumuna bağlı olarak bir olasılık fonksiyonuyla gösterilir. İlgilenilen rassal olay, ayrık (discrete) rassal değişkenlerle modelleniyorsa değişken dağılımı, olasılık dağılım fonksiyonuyla, sürekli (continuous) rassal değişkenlerle modelleniyorsa olasılık yoğunluk fonksiyonuyla ile ifade edilir. Aşağıda ayrık ve sürekli rassal değişkenlere ve bu değişkenlerin olasılık dağılımlarına ilişkin örnekler verilmiştir:

- **Ayrık Rassal Değişken** (Olasılık değeri $\sum_X p(X \leq x)$ ile hesaplanır.)
    - Atılan bir zarın 2 gelmesi: $p(X = 2) = 1/6$
    - 10'den küçük sayılar kümesinde seçilen sayının asal olması $(2, 3, 5, 7)$
        - $p(X < 10) = p(X = 2) + p(X = 3) + p(X = 5) + p(X = 7) = 0.4$

- **Sürekli Rassal Değişken** ($f(x)$ X'in Olasılık yoğunluk fonksiyonu, ε belirlenmiş bir eşik değeri olmak üzere, olasılık değeri $\int_a^b f(x)dx$ ile hesaplanır.)

- 1 *KM* koşmak için geçen süre (sürenin değeri bilinemeyeceği için 3 saniye eşik değeri kullanılır. $p(|100 - X| < 3) = \int_{97}^{103} f(x)dx$

1.3. Olasılık Dağılımı Parametreleri

Bir rassal değişkenin herhangi bir deney sonucunda alacağı değer önceden bilinemez ancak, o değişkenin içinde bulunduğu toplulukla ilgili dağılım fonksiyonu bu değişkenin davranışları hakkında bilgi verir. Dağılım fonksiyonunun sağlayacağı bilgilerin tümü her durumda elde edilemeyebilir ve her zaman da gerekli olmayabilir, bu durumda değişkenin en önemli davranışları dağılım parametreleri aracılığıyla ifade edilir. Dağılım parametrelerinin sağladığı bilgiler ise birçok mühendislik problemi için yeterlidir. En çok kullanılan parametreler ortalama değer(mean) ve ortalamadan sapma değerini veren varyanstır(variance).

Ortalama: Bir rassal değişkenin ortalama değeri, deneyler sonucunda değişkenin dağılım merkezini (beklenen değerini) ifade ettiği için en anlamlı parametre olarak kabul edilir.

Varyans: Ortalama değer, rassal değişkenin dağılım merkezini gösterir, ancak dağılım merkezi etrafındaki yayılma hakkında bilgi vermez. Bu bilgi varyans parametresiyle elde edilir.

1.4. Temel Olasılık Kuralları

Rassal olaylara ilişkin ortalama ve varyans gibi özelliklerin tekrarlanma sayısı arttıkça belli değerlere yaklaştığı ve bu değerler etrafında sabitlendiği görülmüştür. Olasılık teorisinin kuralları, matematiksel modeller oluşturularak olaylarla ilgili çıkarım yapılıp, gerçek dünyaya ait problemlerin olasılığa dayalı çözümlerinin elde edilmesini sağlarlar.

Aşağıda, hemen her problem için gereksinim duyulan temel olasılık kuralları ve açıklamaları verilmektedir.

S örnek uzay ve $f(x)$ X'in olasılık yoğunluk fonksiyonu olmak üzere;

- Olasılık yoğunluk fonksiyonunun değeri hiçbir zaman negatif olamaz.

$$f(x) \geq 0 \; \forall x \in S \;)$$

- X'in alabileceği tüm değerlerin olasılıkları toplamı 1'dir.

$$\int_{-\infty}^{+\infty} f(x)dx = 1 \text{ (sürekli)}$$

$$\sum_{i=1}^{k} f(X_i) = 1 \text{ (ayrık)}$$

- X'in alabileceği tüm değerler belirli bir aralıkta ($[a, b]$) ise bu aralıktaki toplam olasılık 1'e eşittir.

$$p(a \leq X \leq b) = \int_a^b f(x)dx = 1$$

Örnek uzay, bir deneyin muhtemel tüm sonuçlarını içeren bir küme olarak tanımlanır. Örnek uzayın alt kümeleri ise olay olarak adlandırılır ve deneyin muhtemel sonuçlarından birisine karşılık gelir. Bir küme fonksiyonu olarak düşünülebilecek olasılık, küme teorisinde geçerli olan bazı kuralların, olasılık teorisinde de geçerli olması sonucunu ortaya çıkarır. ¬ *mantıksal DEĞİL* işlemi olmak üzere, Q ve R olayları ile bunlara ait $p(Q)$ ve $p(R)$ olasılıkları için aşağıdaki eşitlikler geçerlidir.

$$p(Q) = 1 - p(Q\neg)$$

$$p(Q \cap R\neg) = p(Q) - p(Q \cap R)$$

$$p(Q \cup R) = \begin{cases} p(Q) + p(R), eğer\ Q \cap R = \emptyset\ ise \\ p(Q) + p(R) - p(Q \cap R), diğer \end{cases}$$

Koşullu Olasılık (Conditional Probability) – $\boldsymbol{p(Q|R)}$: R olayının gerçekleştiği bilindiğinde, Q olayının olma olasılığı olarak tanımlanır.

Birleşik Olasılık (Joint Probability) – $\boldsymbol{p(Q,R)}$: Q ve R olaylarının birlikte gerçekleşme olasılığı olarak tanımlanır.

Eğer iki olayın gerçekleşme ihtimalleri birbirinden karşılıklı olarak bağımsız ise bu olayların birleşik olasılığı ayrı ayrı olasılıkları çarpımına eşittir. Aksi durumda birleşik olasılık, koşullu olasılık yoluyla hesaplanır.

$$p(Q,R) = \begin{cases} p(Q) \times p(R), eğer\ Q \cap R = \emptyset\ ise \\ p(Q|R) \times p(R) = p(R|Q) \times p(Q), diğer \end{cases}$$

Zincir Kuralı (Chain Rule): Çok sayıda rassal olayın birleşik olasılığı, koşullu olasılıklar cinsinden ifade edilebilir.

$$p(X_n, X_{n-1}, \dots, X_1) = p(X_n|X_{n-1}, \dots, X_1) \times p(X_{n-1}, \dots, X_1)$$

Toplam Olasılık Kuralı (Total Probability Rule): Bir rassal olayın toplam olasılığı, bağımlı olduğu diğer tüm rassal olayların her biriyle birleşik olasılıkları toplamına eşittir.

$$p(X_n) = p(X_n, X_{n-1}) + p(X_n, X_{n-2}) + \cdots . + p(X_n, X_2) + p(X_n, X_1)$$

Marjinal Olasılık (Marginal Probability): Bir rassal olayın toplam olasılığı, bağımlı olduğu diğer tüm rassal olaylara ilişkin koşullu olasılıkları toplamına eşittir.

$$p(X_n) = p(X_n|X_{n-1}) + p(X_n|X_{n-2}) + \cdots + p(X_n|X_2) + p(X_n|X_1)$$

Bayes Teoremi: İki rassal olayın birleşik olasılığından yola çıkarak, olaylardan herhangi birinin diğeri üzerindeki koşullu olasılığını, diğer olayın kendisi üzerindeki koşullu olasılığı cinsinden ifade eden önemli bir teoridir.

$$p(Q,R) = p(Q|R) \times p(R) = p(R|Q) \times p(Q)$$

$$p(Q|R) = \frac{p(R|Q) \times p(R)}{p(Q)} = \frac{p(R|Q) \times p(Q)}{\int p(R|Q) \times p(Q)} \propto \eta \times p(R|Q) \times p(Q)$$

$$Sonsal(Posterior) = \frac{Olabilirlik(Likelihood)\ \times\ İlksel(Prior)}{Kanıt(Evidence)}$$

Bayes Teoremi, eldeki tüm bilgiyi problemin çözümünde kullanmak için bir yol sunmasıyla gerçek dünyaya ait birçok problemin olasılıksal olarak modellenebilmesine imkân tanıdığı için oldukça yaygın bir kullanıma sahiptir.

EK 2. Stokastik Sistemler ve Süreçler

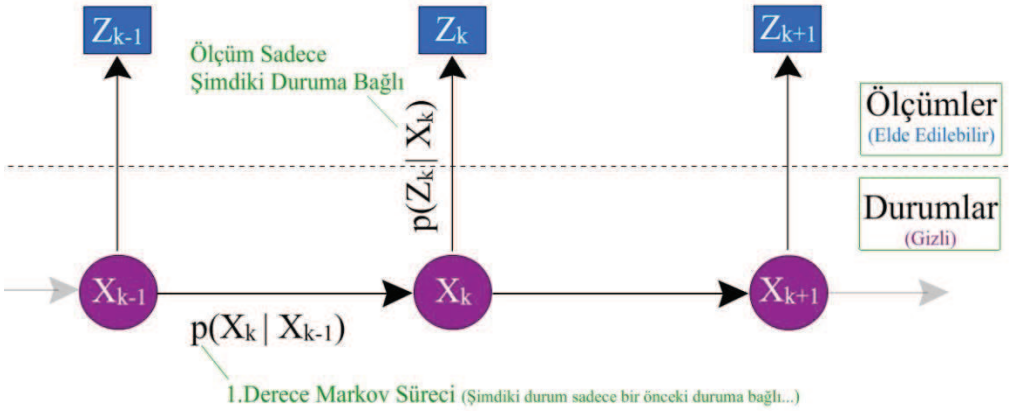
Bileşenleri açıkça gösterilebilen dinamik sistemlerin yanında, "durum uzayı" yaklaşımıyla, çözülmesi mümkün olmayan dinamik sistemler ancak istatistiksel özellikleri incelenerek çözülebilmektedir. Bu tür sistemlere Stokastik Sistemler denilmektedir. Stokastik sistemlere örnek olarak bir reaksiyon odasındaki gaz molekülleri verilebilir. Bu tür sistemler için durum uzayı yaklaşımı pratik değildir. Dolayısıyla bu tür sitemler istatistiksel özellikleri incelenerek ve dinamikleri "rassal süreç" olarak kabul edilerek ele alınır. Stokastik süreçlerle çalışılırken aslında çok sayıda rassal değişkenle çalışılmaktadır. Sürecin parametre uzayına bağlı olarak bu değişkenlerin sayısı sonlu ya da sonsuz olabilir. Bir rassal değişkeni karakterize eden temel nitelik, o değişkenin olasılık dağılımıdır. Bir stokastik sürecin olasılıksal özellikleri, süreci oluşturan rassal değişkenlerin birleşik dağılımdan toplanmıştır. Bu bağlamda stokastik süreçle modellenen sistemlerde rassal değişkenlerin birleşik dağılımı ile ilgilenilir.

2.1. Markov Süreci ve Markov Zinciri

Stokastik süreçler, olasılık dağılımları üzerindeki kısıtlamalara bağlı olarak sınıflandırılırlar. Markov süreci, sonlu durumlu, şimdiki durumu sadece önceki durumuna bağlı olan ve olasılığı zamanda durağan olan bir stokastik süreçtir. Markov süreci sürekli ya da ayrık zamanlı olabilir. Ayrık zamanlı Markov sürecine Markov Zinciri denilir. Markov Zinciri, aslında ayrık zamanlı bir rassal değişkenler dizisidir. Markov Zinciri olarak modellenen ayrık zamanlı dinamik bir sistem için aşağıdaki iki varsayım geçerlidir:

$x_k \in \mathbb{R}^n$ sistemin k zaman adımında sistemin durumunu, $z_k \in \mathbb{R}^m$ ise k zaman adımında sisteme ilişkin yapılan ölçümü göstermek üzere Şekil 2.1.'de Markov zinciri

varsayımları gösterilmiş ve daha sonra da açıklamaları verilmiştir.



Şekil 2.1. Markov Zinciri Varsayımları

- **Varsayım 1:** Sistemin mevcut durumu, geçmişteki tüm zaman adımlarındaki durumları ve bu adımlarda yapılan ölçümlerden bağımsız; sadece bir önceki zaman adımındaki durumuna bağlıdır.

$$p(x_k|x_{0:k-1}, z_{1:k-1}) = p(x_k|x_{k-1})$$

- **Varsayım 2:** Sistemin mevcut durumuna ilişkin ölçüm, sistemin mevcut durumuna kadarki tüm zaman adımlarındaki durumları ve bu adımlarda yapılan ölçümlerden bağımsız; sadece mevcut zaman adımındaki durumuna bağlıdır.

$$p(z_k|z_{1:k-1}, x_{0:k}) = p(z_k|x_k)$$

Bu bağlamda, ~ simgesi bir rassal değişkenler dizisinden aynı dağılıma sahip bağımsız rassal değişkenlerin örneklenmesi (i.i.d – independent and identically distributed) anlamına gelmek üzere, Markov Zinciri olarak modellenen dinamik sistemlerin durum ve ölçüm denklemleri olasılıksal olarak aşağıdaki gibi ifade edilir:

$$x_k \sim p(x_k|x_{k-1})$$
$$z_k \sim p(z_k|x_k)$$

Bu gösterim yoluyla, dinamik sistemlerin durumları arasındaki geçiş ilişkisi ve ölçümlerin mevcut durumla hangi düzeyde örtüştüğü, olasılık dağılımlarıyla ifade edilerek sistem davranışları incelenir ve sistem parametrelenin değerleri olasılıksal olarak kestirilir.

EK 3. İmge İşleme

İmgelerin bilgisayarda işlenebilmesi için öncelikle sayısallaştırılması gereklidir. Günümüzde imgeleri sayısal olarak kaydeden kameralar mevcuttur. Sayısal olarak kaydedilen imgeler, bilgisayara aktarıldıktan sonra uygun bir veri yapısında tutulur. İmgelerle çalışan birçok bilgisayar yazılımı, imgeleri matrislerde tutarak işlemektedir.

3.1. İmge Gösterimi

Matris formundaki bir imge için imgenin yüksekliği matrisin satır sayısı, imgenin genişliği ise matrisin sütun sayısı olarak düşünülebilir. Bu kısımda, imge işleme için MATLAB yazılım ortamı (R2013a) kullanılmıştır. Şekil 3.1.'de bir imgenin gösterimi verilmiştir.

Şekil 3.1. (a) Asıl İmge

$$I = \begin{bmatrix} \vdots & \cdots & \vdots \\ \cdots & \ddots & \cdots \\ \vdots & \cdots & \vdots \end{bmatrix}_{120\times160}$$

Şekil 3.1. (b) İmge Gösterimi

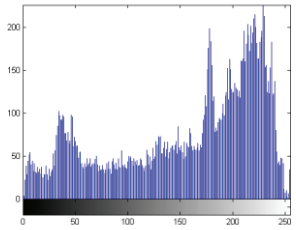
Matrisin her bir elemanı, imgedeki bir piksele karşılık gelmektedir. Örneğin $I(1,1)$ elemanı imgenin sol üst köşesindeki pikseli göstermektedir. Bu elemanın değeri ise, imgenin o pikselin parlaklık değerini ifade etmektedir. Renkli imgelerin gösterim biçimi gri skaladaki (Siyah (0) ve beyaz (255) toplam 256 gri renk değeri) imgeler gibidir ancak RGB formatındaki bir imgede sırasıyla kırmızı (R), yeşil (G) ve mavi (B) renkler için bir katmanda gösterim yapılır.

3.1.1. İmge histogramı

Bir imgenin histogramı imgedeki her bir gri renk değerinin toplam sayısını gösteren bir grafiktir. Sayısal bir imgenin histogramı, imge hakkında genel bir bilgi verir. Eğer histogramı dar ise imgenin zayıf bir görünüme sahip olduğu, geniş ise de imgede daha fazla gri değer bulunduğu, imgenin karşıtlığının ve görünürlüğünün iyi olduğu anlaşılır.

Şekil 3.2. (a) Asıl İmge (RGB)

Şekil 3.2. (b) Gri İmge



Şekil 3.2. (c) Gri İmge Histogramı

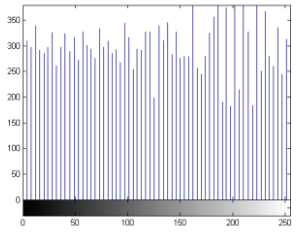
Şekil 3.2 (a)'da RGB biçimindeki asıl imge gösterilmiştir. Asıl imgenin $0-255$ arasında değerlere sahip olduğu, gri imgeye dönüştürülmüş hali Şekil 3.2 (b)'de, gri imgenin histogramı ise Şekil 3.2 (c)'de verilmiştir. Histogramda yatay eksen gri renk skalasını gösterirken, düşey eksen her bir gri renk değerinin imgedeki toplam sayısını göstermektedir.

3.1.2. Histogram eşitleme

Histogram eşitleme, imge güçlendirme amacıyla yoğun olarak kullanılan bir işlemdir. Histogram eşitlemede, imgedeki gri renk değerlerinin renk skalası üzerindeki dağılımı düzgün dağılıma yaklaştırılarak daha iyi bir karşıtlık elde edilmeye çalışılır. Şekil 3.3 (a)'da verilen asıl gri imge, histogramı eşitlendikten sonra daha yüksek karşıtlığa kavuşmuştur. Histogram eşitlemesinden sonra imge Şekil 3.3 (b)'de, histogramı ise 3.3 (c)'de verilmiştir. Bu yeni histogramda, Şekil 3.2 (c)'deki histograma göre daha az sayıda gri renk değerinin bulunduğu görülmektedir.

Şekil 3.3. (a) Asıl Gri İmge

Şekil 3.3. (b) H. Eşitlenmiş İmge



Şekil 3.3. (c) Yeni Histogram

3.2. İmge Aritmetiği ve Noktasal İşlemler

İmgeler üzerinde matematiksel olarak birçok işlem uygulanabilir. Bunların başında dört işlem gelir. İmgeler üzerindeki toplama, çıkarma, çarpma ve bölme işlemleri, matematiksel olarak imgeleri temsil eden matrisler üzerinde aynı işlemlerin gerçekleştirilmesidir. Bu işlemlerden, imge toplama iki veya daha fazla imgeyi üst üste bindirme amacıyla kullanılır. İmge çarpma, imgelerden birini maske olarak kullanmak amacıyla yapılmakla birlikte, bunun için *mantıksal* VE işlemi daha uygundur. İmge çarpma, daha yoğun olarak frekans alanında imgeler üzerinde filtre uygulamak amacıyla kullanılır. İmge çıkarma ve imge bölme ise çoğunlukla iki imge arasındaki farkı ya da değişikliği tespit etmek amacıyla kullanılır.

İmgeler arasındaki değişikliğin tespiti, özellikle hareket yakalama probleminde sıkça başvurulan bir yöntemdir. İmgeler ile sabit sayılar arasında da matematiksel işlemler uygulanabilir. Bu işlemler, genellikle imge parlaklığını artırmak ya da azaltmak için kullanılır. Sabit $k = 1.2$ sayısı ile bir imge arasındaki çarpma işleminin etkisi Şekil 3.4.'de gösterilmiştir.

Şekil 3.4. (a) Asıl İmge

Şekil 3.4. (b) 1.2 ile Çarpılmış İmge

İmgeler üzerinde yapılabilecek diğer işlemler arasında döndürme, kırpma, boyutlandırma ve dönüşüm uygulama sayılabilir. Bu işlemler çoğunlukla, imgelerin istenen biçime getirilerek diğer işlemlere hazır olması için yapılır.

3.3. İmge Filtreleme ve Matematiksel Morfoloji

İmge filtreleme, imgedeki tek bir piksel yerine birbirine komşu olan bir grup piksel üzerinde yapılan işlemler için kullanılan bir tanımlamadır. Filtrelemenin arkasındaki temel fikir, bir imgedeki her pikseli filtreleme çekirdeğindeki katsayılarla çarpıp toplayarak, aynı pikselin yeni değerini hesaplamaktır. Filtreleme işlemlerinin arkasındaki fikir basit olmasına rağmen, filtreleme, imge işlemede çok kullanışlı bir yöntemdir. Filtreleme, imgelerde keskin geçişleri azaltma, gürültüleri temizleme, kenar ve köşeler gibi imgenin önemli özelliklerini tespit etme amacıyla kullanılır.

Dilimizdeki karşılığı "Biçim bilim" olan "Morfoloji", canlıların şekil ve yapıları ile ilgilenen, biyolojiye ait bir alt bilim dalıdır. İmge işlemede ise morfolojik işlemler, filtreleme, inceltme, budama, kenar ya da iskelet elde etme, gürültü temizleme ve bölütleme amacıyla kullanılır. Morfolojik işlemler, gri renkli imgeler üzerinde de uygulanabilir olmakla birlikte, çoğunlukla siyah-beyaz imgeler üzerinde uygulanır. Morfolojik işlemler, kesişim, birleşim, fark, tümleme gibi temel küme işlemlerine dayanır. Aşındırma (erosion), genişletme (dilation), açma (open), kapama (close) en yaygın olarak kullanılan morfolojik işlemler arasında yer alır.

3.3.1. Aşındırma

Aşındırma işlemi, siyah-beyaz (mantıksal) imgeler üzerindeki bölgeleri bir yapılandırma elemanı ile kenarlarından inceltme işlemi olarak tanımlanır. Yapılandırma elemanı, çoğu zaman daire, kare, çokgen gibi ilkel şekillerin döndürülmüş ya da boyutu değiştirilmiş halleri şeklinde kabul edilir. Aşındırma işlemi ile imgelerdeki istenmeyen küçük bölgeler temizlenebilir. Şekil 3.5'de siyah-beyaz asıl imge (a) ve aşındırma işleminin etkisi (b) gösterilmiştir.

Şekil 3.5. (a) Asıl İmge

Şekil 3.5. (b) Aşındırılmış İmge

3.3.2. Genişletme

Genişletme işlemi, aşındırma işleminin tersidir. Siyah-beyaz imgeler üzerindeki bölgeleri bir yapılandırma elemanı ile kenarlarından kalınlaştırma işlemi olarak tanımlanır. Bu işlem kullanılarak, imgelerde bulunmak istenen bölgeler belirgin hale getirilebilir. Şekil 3.6'da siyah-beyaz asıl imge (a) ve genişletme işleminin etkisi (b) gösterilmiştir.

Şekil 3.6. (a) Asıl İmge

Şekil 3.6. (b) Genişletilmiş İmge

3.3.3. Açma

Morfolojik açma işlemi aslında, aynı yapılandırma elemanı ile aşındırma ve genişletme işleminin peş peşe uygulanmasıdır. Açma işlemi ile imgelerde istenmeyen küçük bölgeler temizlenebilir. İmgelerdeki küçük bölgeleri temizlemek için *bwareaopen* komutu da uygulanabilir. Bu işlem ile komşuluk sayısı belli bir değerden az olan piksellerin temizlenmesi mümkündür. Şekil 3.7'de siyah-beyaz asıl imge (a) ve açma işleminin etkisi (b) gösterilmiştir.

Şekil 3.7. (a) Asıl İmge

Şekil 3.7. (b) Açma Uygulanmış İmge

3.3.4. Kapatma

Morfolojik kapatma, açma işleminin tersidir. Kapatma işleminde, aynı yapılandırma elemanı ile genişletme ve aşındırma işlemleri peş peşe uygulanır. Kapatma işlemi ile imgelerdeki bölgeler içinde bulunan boşluklar doldurulmak suretiyle bölgeler daha belirgin hale getirilebilir. İmgelerde boşluk doldurmak amacıyla kullanılabilecek diğer bir komut da *imfill* komutudur. Bu işlem ile komşuluk sayısı belli bir değerden az olan boşluklar kapatılabilir. Şekil 3.8'de siyah-beyaz asıl imge (a) ve kapatma işleminin etkisi (b) gösterilmiştir.

Şekil 3.8. (a) Asıl İmge

Şekil 3.8. (b) Kapatma Uygulanmış İmge

3.3.5. Kenar temizleme

Kenarlar çoğunlukla hareket tespiti amacıyla ardışık imgeler arasında uygulanan çıkarma işlemi sonrasında oluşur. İmge işlemede, önemli beklentilerden birisi de istenmeyen imge kenarlarının temizlenmesidir. *imclearborder* komutu kullanılarak komşuluk sayısı belli bir değerden az olan kenar bölgeleri imgeden çıkartılabilir. Şekil 3.9'da siyah-beyaz asıl imge (a) ve kenar temizleme işleminin etkisi (b) gösterilmiştir.

Şekil 3.9. (a) Asıl İmge

Şekil 3.9. (b) Kenarı Temizlenmiş İmge

EK 4. Veri Birleştirmeye Dayalı Parçacık Filtreleme Yöntemi Kaynak Kodu

Bu kısımda, gerçek zamanlı hareket izleme probleminin çözümü için, veri birleştirmeye dayalı parçacık filtreleme SIR algoritması ile MATLAB R2013a ortamında geliştirilen yazılımın kaynak kodu, her adımdaki açıklamalarıyla birlikte verilmiştir. Kaynak kod içinde, başlangıç koşullarının oluşturulması, parçacık üretme, hareket ve gözlem modellerinin işletilmesi, veri birleştirme ve yeniden örnekleme adımlarında kullanılan yöntemlerin detayları verilmemiş, yalnızca yöntemlere ait giriş ve çıkış parametreleri paylaşılmıştır.

```
% Başlangıç düzeni oluşturuluyor...
[RGBbg,Depthbg] = SFParticleFilter.AuxiliaryOperations.GetBackgrounds();
[vidRGB,vidIR] = SFParticleFilter.Kinect.ConfigureKinect();
% Parçacıklar için başlangıç noktası belirleniyor...
Z0 = SFParticleFilter.SetInitialBelief();
% Parçacık Filtresi başlangıç koşulları oluşturuluyor...
[NoP,X0,dyneq,varX,varK,varR,varD] = SFParticleFilter.AuxiliaryOperations.Initialize(Z0);
% Parçacık filtresi işletiliyor...
%% ADIM 1
    % Parçacıklar üretiliyor...
    X = SFParticleFilter.CreateParticleSet(NoP,X0);
    % Arama uzayı daraltılıyor...
    SFParticleFilter.AuxiliaryOperations.ShrinkSearchSpace();
    T = SFParticleFilter.AuxiliaryOperations.SimulationTime();
while (k<T)
%% ADIM 2
    % Adım 2.1 - Parçacıklar dinamik hareket denklemine göre hareket ettiriliyor...
    X = SFParticleFilter.UpdateParticles(dyneq,varX,X);
    % Adım 2.2 - Kinect cihazı yardımıyla ölçümler yapılıyor.
    SFParticleFilter.Kinect.Trigger([vidRGB vidIR]);
    % Her iki video aygıtından eş zamanlı veri alınarak kaydediliyor...
    [Color] = SFParticleFilter.Kinect.getdata(vidRGB);
    [Depth] = SFParticleFilter.Kinect.getdata(vidIR);
    % Kinect eklem noktaları alınıyor...
    Zk = GetJointCoordinates(Color,Depth);
    % Adım 2.3 - Gözlem modeli işletiliyor...
    % Aktif bölge(RGB) 141x141 px olarak oluşturuluyor...
    SFParticleFilter.ObservationModel.InitWindow(Zk);
    % Renk & Derinlik modelleri işletiliyor...
    handR = SFParticleFilter.ObservationModel.ColorBasedHandDetect(Zk,RGBbg);
    handD = SFParticleFilter.ObservationModel.DepthBasedHandDetect(Zk,Depthbg);
    % El avuç içi merkez koordinatları elde ediliyor...
    [Zr,Zd] = SFParticleFilter.ObservationModel.GetLargestAreaCentroid(handR,handD); %
    Uzaklıklar hesaplanıyor - Kinect, Renk, Derinlik
    dist = SFParticleFilter.ObservationModel.CalculateDistances(Zk,Zr,Zd,X);
    % Uzaklıklar (toplamı 1 olacak şekilde) normalize ediliyor...
    ndist = SFParticleFilter.NormalizeWeights(dist);
    % Olabilirlik değerleri hesaplanıyor - Kinect, Renk, Derinlik
    [lhK,lhR,lhD] = SFParticleFilter.ObservationModel.CalcLikelihood(dist,varK,varR,varD);
    % Olabilirlik değerleri normalize ediliyor...
    [nlhK,nlhR,nlhD] = SFParticleFilter.NormalizeWeights(lhK,lhR,lhD);
    % Veri birleştirme yapılıyor...
    lh = nlhK .* nlhR .* nlhD;
    % Parçacık ağırlıkları güncelleniyor...
    X.weight = lh .* X.weight;
    % Ortak olabilirlik değeri normalize edililiyor...
    nlh = SFParticleFilter.NormalizeWeights(X.weight);
    % Parçacık filtresi kestirimi hesaplanıyor - Robust Mean: en iyi %10 parçacığın
    ortalaması
    Ep = SFParticleFilter.Estimation.CalcPRMean.(X);
%% ADIM 3
    % Ağırlık dejenerasyonuna yönelik olarak artık terimli yeniden örnekleme yapılıyor...
    X = SFParticleFilter.Resampling.Residual(X,NoP,nlh);
    % Hareket takibi görselleştiriliyor...
    SFParticleFilter.AuxiliaryOperations.DrawSkeleton(Zk,Zr,Zd,Ep,X);
end
```

ÖZGEÇMİŞ

Tuğrul TAŞCI, 25.07.1978 tarihinde Gümüşhane'de doğdu. İlk, orta ve lise öğrenimini bu ilde tamamladı. 1996 yılında başladığı Sakarya Üniversitesi Bilgisayar Mühendisliği bölümünden 2001 yılında mezun oldu. 2001-2004 yılları arasında Sakarya Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar ve Bilişim Mühendisliği anabilim dalında yüksek lisansını tamamladı. 2006 yılında aynı anabilim dalında doktora öğrenimine başladı ve halen öğrenimine devam etmektedir. Tuğrul TAŞCI, 2001 yılından itibaren sırasıyla Enformatik Bölümü, Uzaktan Eğitim Araştırma ve Uygulama Merkezi ve Bilgisayar Bilişim Fakültesi Bilgisayar Mühendisliği Bölümünde araştırma görevlisi olarak çalıştı. Halen Uzaktan Eğitim Araştırma ve Uygulama Merkezi'nde Müdür Yardımcısı olarak görev yapmaktadır.