Gazi Üniv. Müh. Mim. Fak. Der.
Cilt 24, No 2, 245-255, 2009

J. Fac. Eng. Arch. Gazi Univ.
Vol 24, No 2, 245-255, 2009

VİDEO DİZİLERİNDE AYRIT SEZMEYE DAYALI HIZLI HAREKET KESTİRİMİ

Ziya TELATAR
Elektronik Mühendisliği Bölümü, Mühendislik Fakültesi, Ankara Üniversitesi, 06100-Beşevler,Ankara
telatar@eng.ankara.edu.tr

(Geliş/Received: 10.09.2008 ; Kabul/Accepted: 25.02.2009)

ÖZET

Bu makalede video dizilerindeki çoklu obje hareketini kestirmede ve algılamada kullanılabilecek hızlı bir algoritma sunulmuştur. Yöntem çerçeveler arasında imgenin satır ve sütunları boyunca piksel değerlerindeki kaymalarına bağlı olarak vektör yansımalarının hesaplanması ve bu yansımaların çerçevelere yansıtılması ile ilgilidir. Ardışık çerçevelere dayalı yansımalar kaymaların ayrı ayrı yatay ve düşey yönlerde algılanması ile belirlenirler. Bu kaymalar için çerçeveler arası ilişkilerin belirlenmesinde öbek eşleme tabanlı ortalama mutlak fark yöntemi kullanılmıştır. Her bir çerçevedeki pikseller arasındaki keskin geçişler veya kısaca kenar detayları, hareket bilgilerinin belirlenmesinde ve algoritmanın hesap yükünü azaltmada etkili bir yöntem olarak sunulmuştur. Deneysel çalışmalar algoritmanın objeleri algılama ve izleme yeteneğinin yanında işlem süresinde de bir azalma sağladığını göstermiştir.

Anahtar Kelimeler: Hareket kestirimi, nesne hareketi arama, öbek eşleme, ayrıt sezme.

FAST MOTION ESTIMATION IN VIDEO SEQUENCES BY EDGE DETECTION

ABSTRACT

In this paper, a fast approach to detect and estimate the multiple object motion in video sequences is presented. The method is based on transforming each image in the sequence of frames into vector projections formed by accumulating pixel shift value along the rows and columns of the image. The projections corresponding to consecutive frames are in turn used to detect and estimate the individual horizontal and vertical components of the possible shifts. For that reason, block matching and maximum absolute difference (MAD) algorithms were used in detecting and tracking the moving objects in sequences. Sharp transitions or shortly edges in each image frame are also considered as an efficient method in the algorithm for decreasing the computational costs. Experimental results show that, usefulness of the method is presented not only in the context of motion analysis and tracking ability of objects, but also for a decrease in process time required for detecting and tracking.

Keywords: Motion estimation, object motion search, block matching, edge detection.

1. GİRİŞ (INTRODUCTION)

Hareket analizi, imge dizileri içerinde statik hareket ve obje veya kameranın hareketi sonucu oluşan değişimlerin tanımlanması ve algılanmasına dayanır. İnsan görme sistemi için hareket algılama doğası gereği yapılmaktadır. Bilgisayar ortamında ise ardışıl iki çerçeve arasında hareket eden bir objenin hareketinin algılanması ve izlenmesi bazı güçlükleri de beraberinde getirir. Hareket yönünün değişimi, eğimlerin değişimi, gölgenin takip edilmesi, karmaşık arkaplan ve gürültü bu zorluklardan bazıları olarak verilebilir. Tanımlanan etkiler genellikle nesne veya kamera hareketi yada elektronik sistemlerin çalışmalarından kaynaklanırlar. Bunlardan birincisi sabit obje sınırları ile analiz yapılması ve diğerleri de obje ve kamera hareketi nedenleriyle ortaya çıkabilirler. Hareket kestirimi, bir video dizisi içerisinde ardışıl çerçevelerde hareket eden objelerin hareket bilgilerinin tanımlanmasıdır. Sonuçta her bir durum için hareket analizi, hareketi kestirme ve izleme için çerçeveler arasındaki eşleşen noktaları belirlemek ve hareketin dinamiğini ortaya koymaktır.

İmge dizilerinde ardışık çerçeveler arası ilişkinin ortaya konulması için çeşitli yöntemler geliştirilmiştir. Bu yöntemler arasında en yaygın ve en kullanışlı olanı öbek eşlemedir [1, 2, 3, 4, 8]. Öbek eşleme, sadeliği, donanım karmaşıklığının daha az olması, hareket vektörlerinin kodlanmasındaki verimliliği gibi avantajlarından ötürü ITU-T H.261 (ITU-T SG15 1993), H.263 (ITU-T SG15 1996), ISO MPEG–1 (ISO/IEC JTC 1/SC29 1993), MPEG–2 (ISO/IEC JTC 1/SC29 1996), MPEG–4 (ISO/IEC 14496–2 1998) gibi birçok uluslararası video kodlama standartları tarafından da benimsenmiştir. Hareket kestirimi için de önemli uygulamaları olan obek eşleme için imge öbeklere ayrılır ve imge içerisindeki karmaşık hareketler basit küçük öbekler yardımıyla belirlenmeye çalışılır. Burada amaç ilk çerçeveden seçilen bir öbeğe en iyi karşılık gelen sonraki çerçevedeki öbeği bulmaktır. Bunun için referans öbeğinin bir sonraki çerçeve içerisinde yer değiştirmesi eşleme kriterleri yardımıyla bulunur. Burada yer değiştirme çerçeveler arasındaki farka bakılarak hareket yönünü veren vektör yansımaları ile belirlenir ve değişimler bu vektörler yardımıyla izlenir. Demir vd. [1] zorlamalı 1 bit dönüşüm ile öbek eşleme bir yöntem önermişlerdir. Öbek eşleme doğruluğunu artırmak için aynı genlik değerine sahip zıt yönlerdeki piksellerin farklı sınıflara atanma hatasını azaltacak bir yöntem üzerinde çalışmışlardır. Dufaux vd. [2] üç aşamalı hiyerarşik bir yöntemi ortaya koymuşlardır. Bunun için önce düşük-geçirgen bir görüntü piramidi oluşturmuşlar ve bir başlangıç dönüşüm parametresi ile kestirime başlamışlardır. Son aşamada ise azalan dik yokuş (Gradient descent) algoritması ile çerçeveler arasında hareket kestirimi yapılmaya çalışılmıştır. Namuduri [3] öbek eşlemeye dayalı olarak monotonik hata yüzeyini araştırarak zamansal-mekansal komşuluk bilgisinden hareket vektörleri için en iyi kestirimi yapmaya çalışmıştır. Traver vd. [4] log-polar imgelerdeki çerçeveler arası öbeklerin zamansal bölge yansımalarından hareket kestirimi yapmıştır. Telatar[5] Öbek eşleme tabanlı çerçeveler arası hareket vektörlerinden tam arama algoritması ile yön ve hız kestirme çalışmasını sunmuştur.

Referans öbeğini bir sonraki çerçeve içerisinde arama çabuk ve etkili eşleme için önemli kriterlerden birisini oluşturur. Fakat burada önemli problemlerden biri, özellikle video dizilerinde çerçevelerin çok hızlı hareket etmesi ve algoritmaların çerçevelerin geçiş hızına göre yavaş kalmasıdır. İdeal olarak çerçeveler arasında bir kayıp olmaması için algoritmaların yaklaşık olarak 1/24 saniyede sonuç üretmesi beklenir. Ancak, pratikte bu mümkün değildir ve mevcut algoritmalar ile bazı kayıplarla çerçeve atlayarak hareket tahmini yapmak mümkün olabilmektedir. Bu nedenle arama algoritmalarının kısmi bir arama bölgesi içerisinde eşleme hata kriteri hesaplama ve karşılaştırması prensibine göre çalışması düşünülmüştür. Bu algoritmaların etkinliği arama bölgesi sayıları ile yakından ilgili olabildiği gibi arama bölgesinin seçimi ile de ilgilidir. Fazla sayıda arama bölgesinin iyi sonuç vereceği kesindir. Ancak gereksiz yapılan aramalar algoritmanın hızında önemli miktarda yavaşlamalara sebep olabilir. Bu nedenle en az sayıda öbek ile en iyi performansın yakalanması çok önemlidir. Literatürde bu amaç için kullanılan değişik algoritmalar vardır. İki ardışıl çerçevenin tamamında arama yapan (Full Search) algoritmalarda [5, 6] doğrudan pikseller eşleştirildiğinden hata oranı az olmakla birlikte arama süreleri uzun olmaktadır. Zhu vd. [7] hızlı arama yapabilmek için öbek eşleme tabanlı eşkenar yönde arama (diamond search) algoritmasını önermiştir. Bu algoritma hızlı eşleme yapmakla birlikte hata oranı tam arama kadar küçük olamamaktadır. Chau vd [8] modifiye 3 adımda arama (three step search) aramasını geliştirmiş ve ilk adımda küçük bir eşkenar boyunca herhangi bir sınırlama olmaksızın arama yapmıştır. Daha sonrasında merkez etrafında aramayı devam ettirmiştir. Koç vd. [9] her bir yöndeki hareketi kestirebilmek için değiştirilmiş alt uzay yönalgılama algoritmasını sunmuşlardır. Liu vd. [10] çerçeveler içinde her bir alt öbekteki dik yokuş genliklerinden eşik seviyesi ile bir indirgeme sıralaması yaparak en iyi eşleşenleri belirlemişlerdir. Bu yöntemde hızlı sonuç için ek olarak hızlı yaklaşık hareket kestirimi hata distorsiyonu şemasını kullanmışlardır.

Hareket kestirimi ve arama algoritmalarının iyi sonuç üretme ile hızlı çalışması arasında bir tercih yapılması düşünülebilir. Her ikisinin bir arada olamayacakları kesindir. Çok iyi sonuçlar üreten bir algoritmanın işlem yükünün fazla olması nedeniyle yavaş çalışacağı açıktır. Yukarıda literatür özetinde verilen yaklaşımlarda da iyi sonuçların elde edilebilmiş oldukları gözlenmiştir. Ancak gerçek zamanlı hareket kestiriminde kaliteden çok az ödün vererek iyiye yakın hızlı çalışabilecek yeni algoritmaların geliştirilmesine ihtiyaç bulunmaktadır.

Bu çalışmada imge dizilerinde hareketli bir veya daha fazla nesnenin öbek eşleme temeline dayalı hareket kestirimi ve hareket vektörleri yansımalarının belirlenmesi konusu ele alınmıştır. Çerçeveler arasında çoklu değişen özellikler ve geçici oluşumlar tanımlanmış, bu bilgiler ile hareket değerlendirmesi yapılmıştır. Hareket kestiriminin video kalitesine yaklaştırılabilmesi amacıyla hareketli nesneye ait kenar sınırları bulunarak yalnızca bu bölgelerin daha sonraki çerçevelerde aranması gerçekleştirilmiştir. Bu olası hareketli bir nesnenin tüm çerçevede aranması yerine sadece nesnenin bulunduğu yerin belirlenmesini ve daha sonraki çerçevede de yalnızca bu öbeğin yakınlarında aranmasına olanak sağlar. İkinci bölümde çalışmada önerilen yöntem için materyal ve metot verilmiştir. Bu başlık altındaki alt bölümlerde öbeklere bölerek ayrıt bulma ve ayrıta

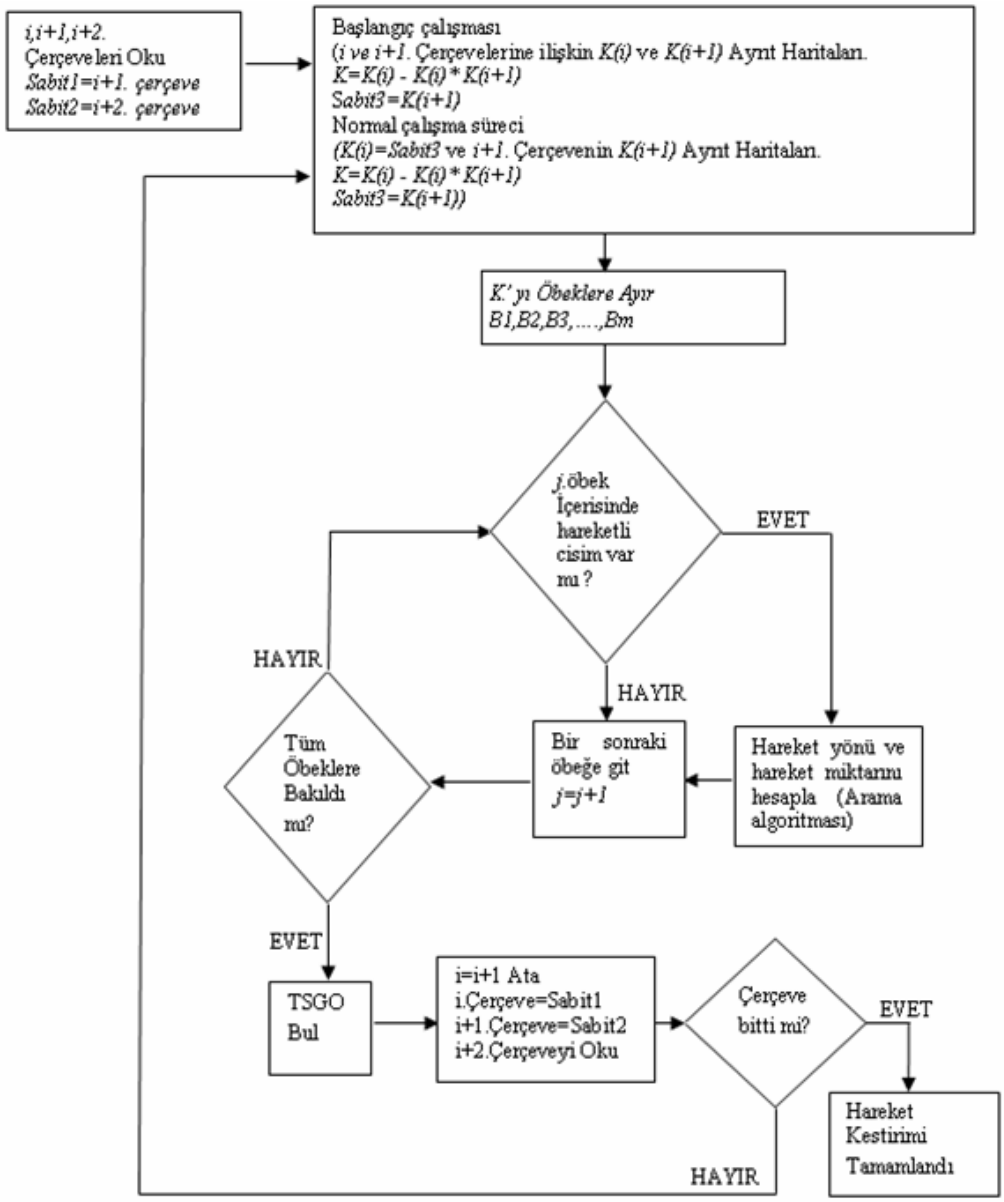
dayalı hareket tahmini için geliştirilen algoritma detayları açıklanmıştır. Üçüncü bölümde deneysel sonuçlar sunulmuş olup dördüncü bölümde ise genel çıkarımlar verilmiştir.

2. MATERYAL VE YÖNTEM (MATERIAL AND METHOD)

Bu çalışmada video dizilerinde çerçeveler arasında hareket eden nesneleri kestirip izleyebilecek bir algoritma önerilmiştir. Şekil 1'de önerilen yöntem için genel bir öbek şeması verilmiştir.

Şekil 1'de verilen şemada obje tanımlayıcı, imgenin bölgesel pixel değişimlerine duyarlı bir eşleme yapar. Bunun için bir çerçeve içerisindeki geri plandaki sabit olanlar dahil olası objeleri belirleyebilmek amacıyla obje ayrıtları (edges) hesaplanır ve hareketli obje tanımlama ile arka plan ve objenin ayrıştırılması sağlanır.

Şekil-1 de blok şeması verilen hareket kestirimi algoritmasının çalışması kısaca şöyle açıklanabilir. Başlangıçta ilk üç çerçeve okunarak bu çerçevelere ilişkin *K(i)* ayrıt haritaları hesaplanır. Bölüm-2.1 içerisinde Eşitlik-2 ve 3 te verildiği gibi yalnızca hareketli bölgeleri ihtiva eden *K* bulunur. Bu işlem algoritmanın normal çalışma süreci içerisinde başlangıç koşulu olarak kabul edilir. Aynı işlemler normal çalışma süreci içerisinde bir sonraki çerçeve ile güncel hesaplanmış çerçeve arasında olmak üzere tekrarlanır. Çerçeveler öbeklere bölünerek her bir öbek içerisinde hareketin olup olmadığına bakılır ve bu işlemlere bir döngü içerisinde son çerçeveye kadar devam edilir. Burada amaç tüm çerçeve içerisinde arama işlemini sadece hareketin bulunacağı öbeklere indirgemektir.



Şekil-1. Hareket kestirimi algoritması akış diyagramı (Motion estimation algorithm block diagram)

2.1 Nesne Kenarlarının ve Hareketli Nesnelerin Bulunması (Fındıng Object Boudarıes and Movıng Objects)

Bir çerçeve genellikle oldukça düzgün arka plan ile arka plandan farklı olan objeleri ihtiva eder. İkisi arasındaki geçiş bölgelerinde imgenin fiziksel yapısında da önemli değişiklikler oluşur. Arka plan ve objeler arasındaki keskin geçişler doğası gereği insan görme sistemi tarafından kolaylıkla ayırt edilebilirler. Ancak bilgisayar ortamında bu bölgelerin algılanması ancak onların iyi bir şekilde tanımlanmasıyla olabilir. Obje sınırlarını tanımlamanın en basit ve en iyi yolu imge ayrıtlarını hesaplamaktır. Bu çalışmada imge ayrıtları için Gradient tabanlı ve Laplacian tabanlı yöntemler ele alınmıştır ve farklı durumlar için en iyi performansı sağlayan ayrıt algoritmaları belirlenmiştir.

Gradient yöntemi imge içerisindeki satır ve sütünların her iki yöndeki türevleri (farklar) olarak ifade edilir [11].

$$\nabla f(x,y) = \frac{\partial f(x,y)}{\partial x} + \frac{\partial f(x,y)}{\partial y} \qquad (1)$$

Laplacian Yöntemi ise satır ve sütün yönlerinde ikinci türeve karşılık olup sıfır geçişlerini gösterir ve Eşitlik-2 de verildiği gibi ifade edilir [11].

$$\nabla^2 f(x,y) = \nabla(\nabla f(x,y)) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} \qquad (2)$$

Ayrıt sezme için Sobel, Roberts ve Log operatörleri kullanılmıştır. Ayrıt sezici çıkışında elde edilen bütün noktalar ile bu noktaların belirlediği sabit ve hareketli obje sınırları hesaplanır. Sabit ve hareketli objelerin ayrıştırılabilmesi için güncel çerçevede ve bir önceki çerçevede hesaplanan ayrıt haritaları çarpılır (Şekil-2).

$$K_1 = K(i) * K(i-1) \qquad (3)$$

Eşitlikle *K(i)* ler güncel ve bir önceki çerçevelerin ayrıt haritalarını, K_1 de çıkış imgesini verir. Bu sonuç hareketli pixel'lerin çarpımları sıfır olacağından dolayı yalnızca ayrıt haritasında elde edilen sabit arkaplana ve sabit objelere ait bilgileri ihtiva eder. Daha sonra hareketli bölgeler ise;

$$K = K_1 - K(i) \qquad (4)$$

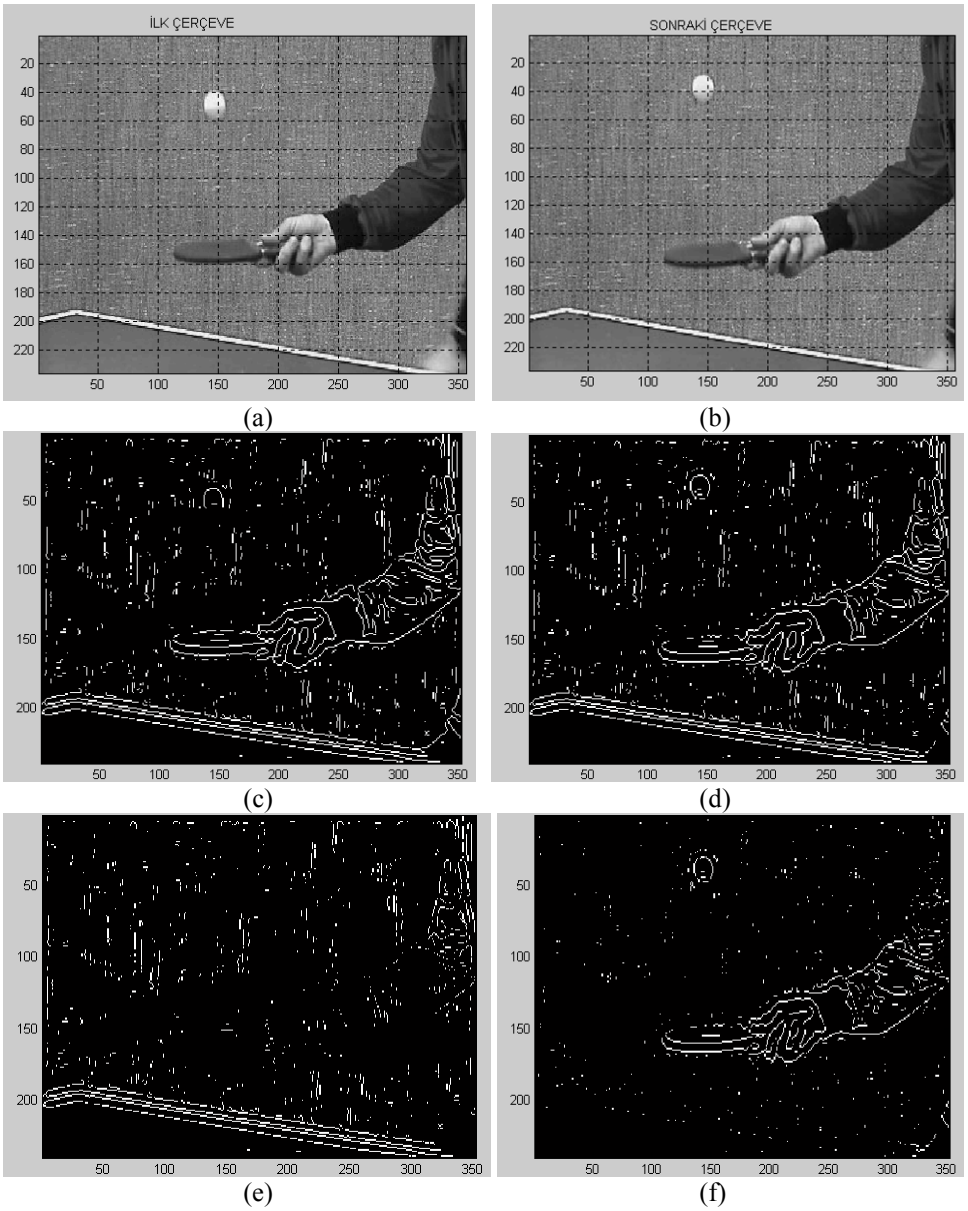
eşitliği ile elde edilir. *K* imgesi sadece hareket ihtiva eden nesnelerin bir imgesi olacaktır. Şekil-2 de obje tanımlama işlemi sonucu elde edilen imge gösterilmiştir. Şekil-2.a ve Şekil-2.b bir video dizisi içerisindeki ardışık iki çerçeveyi göst. Bu imgeler içerisinde sabit ve hareketli objeler karmaşık olarak bulunmaktadırlar. Şekil-2.e, (c) ve (d) deki imgelerin ermektedir. Şekil-2.c, (a) daki imgenin ve Şekil-2.d ise, (b) deki imgenin kenar haritalarını göstermektedir çarpımı sonucu elde edilmiş olup yalnızca iki çerçeve arasındaki hareketsiz bölgeleri ve objeleri göstermektedir. Şekil-2.e imgesi Şekil-2.c imgesinden çıkarıldığında, Şekil-2.f imgesinde gözlediğimiz gibi el, top ve rakete ait sadece haraket olan kısımlar elde edilmiştir. Bu işlem algoritmanın toplam işlem süresinde önemli ölçüde tasarruf sağlamaktadır. Şekil-2.f imgesi elde edildikten sonra, bir sonraki çerçevede hareketin yönü ve büyüklüğü belirlenir. Bu amaçla, Şekil-2.f deki çerçeve öbeklere ayrılır. Öbek içerisindeki kenar piksellerinin yoğunluğuna bakılır. Gürültüden dolayı ve Şekil-2.e imgesi Şekil-2.c imgesinden çıkarılması nedeni ile Şekil-2.f de görüldüğü üzere öbekler içinde az sayıda ayrıt bulunur. Bu nedenle öbekler içerisindeki ayrıt değerleri belirlenen bir eşik değeri ile karşılaştırılır. Ayrıt haritası içerisindeki piksel değerleri bu eşik değerinin üzerindeyse bu öbek içerisinde hareketli bir objenin olduğuna karar verilir. Bu işlemin bir sonucu olarak, ayrıt haritasında sadece eşik seviyesi üzerinde kalan pikseller değerlendirildiğinden ve diger noktalar sabit bölge olarak işlem dışında bırakıldığından dolayı gürültüye karşı duyarlılık azaltılmış olmaktadır. Ayrıt haritasında referans çerçevedeki öbekler içerisinde belirlenen objenin, sonraki çerçevedeki yeri arama algoritması ile bulunur [7].

2.2. Öbeklere Ayırma (Dıvıdıng Blocks)

İmge üzerinde pikseller ile tüm imge boyunca işlem yapmak hafıza gereksinimleri ve işlem sürelerinin uzun olmaları nedeniyle araştırmacılar tarafından tercih edilmez. Bunun yerine imge alt öbeklere ayrılarak benzer işlemler alt öbeklerde gerçekleştirilir. İmge dizilerinde de (Videolarda) aynı şekilde öbeklere ayırma, işlem yükü açısından önemli ölçüde tasarruf sağlar ve çerçeveler arası ilişkiler daha küçük alanlar içerisinde aranır. Bir başka deyişle, öbek eşleme için, işlenecek olan çerçeve *KxK* boyutundaki alt öbeklere bölünür ve bu alt öbekler G_i çerçevesi içerisinde alt öbek matrisleri şeklinde gösterilirler. Böylece G_i, *M/K* ve *N/K* alt öbek matrisler olarak ifade edilirler.

Bir imge dizisi Eşitlik-5 deki gibi ifade edilebilir.

$$G_i(x,y,t) = G_0(x,y,t_0) + G_1(x,y,t_1) + \ldots\ldots + G_n(x,y,t_n) \qquad (5)$$



Şekil-2. a. Dizi içerisindeki referans çerçeve, b. bir sonraki çerçeve, c. (a)' nın kenar haritası, d. (b)' nin kenar haritası, e. (c) ve (d) imgeleri çarpımı, f. (c) ve (e) imgeleri farkı (hareketli noktalar) (a. Reference frame in sequence, b. Next frame, c. Edge map of (a), d. Edge map of (b), e. Multiplication of images in (a) and (b), f. Difference between images (a) and (b)).

Bu ifadede $G_i(x,y,t)$ herhangi bir t anındaki çerçevedeki pixel değerini tanımlar. Çerçeveleri öbeklere ayırmadan önce ayrıt algılama algoritması uygulayarak çok daha hızlı ve etkili bir biçimde hareket kestirimi yapılması sağlanmıştır. Ayrıt bulma algoritmasının imge üzerindeki etkisi şu şekilde açıklanabilir:

1. Geri Plan Bölgesi: Hareketin olması için hareket eden bir obje gerekir.Geri planda herhangi bir obje olmadığından dolayı, ayrıt haritasında hiçbirşey görünmez ve geri plan olan bölge 0 ile gösterilir. Arama algoritmasıyla bu bölgede gereksiz arama yapılmamış olur. Bu işlem zamandan önemli ölçüde tasarruf sağlar.

2. Sabit Nesneler Bölgesi: Kenar bulma algoritması sonucunda çerçeve içerisindeki objelerin ayrıtları bulunur.Bu sabit objeler içinde geçerlidir. Ancak sabit objeler veya arka plan birbirini izleyen her çerçevede (çekim boyunca) aynı kaldığından burada da arama yapılmasına gerek yoktur. Ayrıtları bulunmuş ilk ve devam eden çerçevelerin çarpımlarıyla elde edilen sonuç (Eşitlik-3 deki K_1) ile ayrıt haritası hesaplanmış ilk çerçevenin farkı hesaplanarak (Eşitlik-4 deki K) hareketsiz objelerin olduğu bölge işlem dışına atılmış olur.

3. Hareketli Objeler Bölgesi: Hareketli objenin olduğu bölge olup, objenin ayrıtları, ayrıt haritasında 1 ile gösterilir. Objenin bir sonraki çerçevedeki yeri tesbit edilerek hareket vektörleri oluşturulur.

Bu çalışmada imge dizilerinde hareketli bir veya daha fazla objenin öbek eşleme temeline dayalı hareket kestirimi ve hareket vektörlerini belirlenmesi konusu ele alınmıştır. Çerçeveler arasında çoklu değişen özellikler ve geçici oluşumlar tanımlanmış, bu bilgiler ile hareket değerlendirmesi yapılmıştır.

Hareket kestiriminin video kalitesine yaklaştırılabilmesi amacıyla referans çerçevede hareketli objeye ait sınırlar bulunarak yalnızca bu bölgelerin sonraki çerçevede aranması gerçekleştirilmiştir. Bu başlangıçta olası hareketli bir objenin tüm çerçevede aranması yerine sadece objenin bulunduğu yerin belirlenmesini ve daha sonraki çerçevede de yalnızca bu öbeğin yakınlarında aranmasına olanak sağlar.

Şekil 3.a da bir önceki çerçeveden seçtiğimiz referans öbeğinin güncel çerçeve (Şekil 3.b) içerisinde yer değiştirmesini bulmak amacıyla güncel çerçeve ayrıt bilgisini de kullanarak Şekil 3.c'deki gibi öbeklere ayrılmıştır. Her bir öbek için bir önceki çerçevede belirlenen kısıtlı arama bölgesi sınırları içerisinde, kısmi tam arama ile öbek eşleme yapılmış ve eşleme için ortalama mutlak fark (MAD) yöntemi uygulanarak hareket vektörleri bulunmuştur (Şekil 3.d).

Ayrıt haritası ile belirlenen hareketli objelerin bulunduğu öbekler arama öbeklerini belirler ve sonraki çerçevede yalnızca bu öbeklerin etrafındaki öbeklerde arama gerçekleştirilir. Hareket halindeki objenin yön ve hareket bilgisi ilk çerçevedeki öbeğin ikinci çerçevedeki vektör yansımasının bulunması ile elde edilir. Bunun için *x* ve *y* yönlerinde ayrı ayrı tarama yapılır. *x* ve *y* yönündeki taramalar için aşağıdaki ifadeler yazılabilir;

$$S_{i-1,i}(x,t) = \sum_{t=1}^{j}\sum_{y} m^{(j)}(x_{(t)} - x_{(t-1)}, y_{(t)} - y_{(t-1)}) \quad (6)$$

$$S_{i-1,i}(y,t) = \sum_{t=1}^{j}\sum_{x} m^{(j)}(x_{(t)} - x_{(t-1)}, y_{(t)} - y_{(t-1)}) \quad (7)$$

Hareket vektörü konumu ve boyu ise Eşitlik-8 e göre bulunabilir.

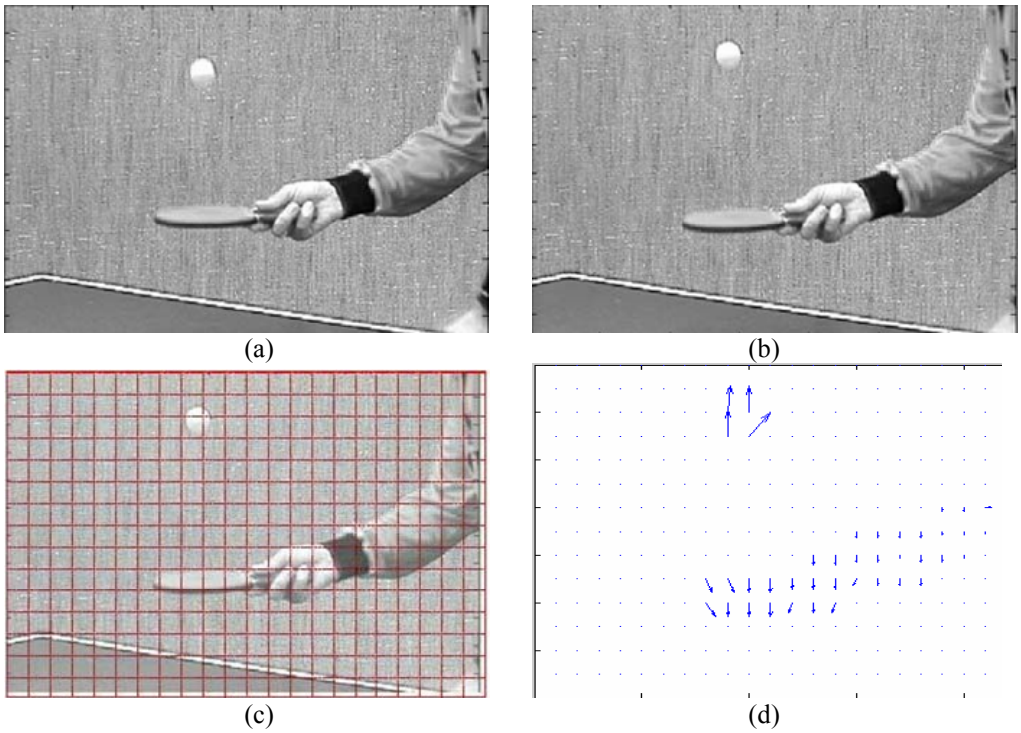
$$OMF(dx,dy) = \frac{1}{mn}\sum_{i=-n/2}^{n/2}\sum_{j=-m/2}^{m/2}\left|G_1(i,j) - G_2(i+dx, j+dx)\right| \quad (8)$$

Burada, $G_1(i,j)$ ilk çerçevedeki makro öbek, $G_2(i,j)$ ikinci çerçevedeki makro öbek, (dx,dy) arama konum hareket vektörünü belirtirler. Tüm algoritmanın işleyişini gösteren akış şeması Şekil-1 de verilmiştir.

İki ardışık çerçeve arasında hesaplanan hareket vektörlerinin bulunmasında oluşan hatanın tepe sinyal-gürültü oranı (TSGO) Eşitlik-9 a göre hesaplanmıştır.

$$TSGO = 10\log_{10}\frac{(\sum\sum G_i^k)^2}{\sum\sum (G_i^k - G_i)^2} \quad (9)$$

Tepe sinyal gürültü oranı hesaplamaları dB cinsinden yapılmıştır.



(a) (b) (c) (d)

Şekil-3. a. Bir önceki çerçeve b. güncel çerçeve c. Öbeklere ayrılmış güncel çerçeve d. Hareket vektörleri (a. Previous frame, b. Current frame, c. Current frame divided blocks, d. Motion vectors)

3. SONUÇLAR (RESULTS)

Bu çalışma 1.7 GHz işlemci ve 512 Mbyte RAM özelliklerine sahip bir bilgisayarda MATLAB ortamında gerçekleştirilmiş ve 240x352 piksel büyüklüğünde 100 çerçeveden oluşmuş Tennis ve Diskus imge dizileri ile deneyler yapılmıştır. Deneylerde öbek büyüklüğü 16x16 piksel olarak alınmıştır. Deneylerde ayrıt sezme algoritmasının kullanılması ve kullanılmaması durumlarında algoritmanın toplam süreler yönünden performansı test edilmiştir.

Çizelge-1'de algoritmanın kenar algılama algoritması uygulanması ve uygulanmaması durumunda değişik arama algoritmaları için hareket kestirimi sürelerinin bir karşılaştırması verilmiştir. Çizelge-1 de kullanılan bazı kısaltmaların tanımları:

Durum A: Ayrıt algılaması kullanılmıyor. Şekil-2.b deki güncel çerçeve öbeklere ayrılır.

Durum B: Ayrıt algılaması kullanılıyor. Şekil-2.e'deki gibi hareketsiz sabit bölgeler işlem dışı bırakılmıştır. Şekil 2.d'deki ayrıt haritası bulunarak güncel çerçeve öbeklere ayrılır. Eğer öbek içerisinde hareketli obje olduğu tespit edilirse, referans çerçevede arama bölgesi içerisindeki en iyi öbek eşlemeyi bulmak için aktüel çerçevedeki o öbeğin kendisinin değeri kullanılır (ayrıt eşleme yapılmaz).

Durum C: Ayrıt algılaması kullanılıyor. Durum B'den farkı, Şekil-2.b'deki çerçeve öbeklere ayrılır.

Arama süreleri göz önüne alındığında, ayrıt algılama algoritması uygulayarak Durum C'de sürenin azalmış olduğu, sabit bölgeler ayrıştırılarak işlem dışında bırakıldığında yani Durum B'de ise bu sürelerde biraz daha daha azalma sağlandığı gözlenmiştir. Ayrıt algılama algoritması uygulandığında, seçilen arama yöntemine göre arama sürelerinde değişiklikler olmaktadır.

Tam arama ile deneylerde kullanılan diğer üç arama algoritması arasında arama süreleri açısından bir fark gözlenmiştir. Ayrıt algılama algoritmasından yararlanarak tam arama yapılması veya ayrıt algılama algoritmasından yararlanmayarak tam arama yapılması durumları için arama süreleri ve toplam süre açısından büyük bir fark olmasına karşın, üç adım arama (TSS), eşkenar arama (DS) ve 2boyutlu (2D) arama algoritmalarında ayrıt algılama algoritması kullanılması durumunda sadece arama süresi

Çizelge-1. Algoritmanın arama ve toplam süreler açısından karşılaştırılması (Table-1 Comparisons for search and total computation times of algorithm)

		ARAMA SÜRESİ	TOPLAM SÜRE			ARAMA SÜRESİ	TOPLAM SÜRE
TENNIS DURUM A	TSS	26,92	149,58	DISKUS DURUM A	TSS	32,97	177,38
	DS	17,38	138,14		DS	29,00	174,41
	2D	21,62	142,14		2D	28,13	176,14
	FULL	250,33	370,59		FULL	300,19	442,89
DURUM B (TSS)	LOG	12,86	166,45	DURUM B (TSS)	LOG	17,49	213,06
	SOBEL	9,16	154,83		SOBEL	11,35	189,37
	ROBERTS	7,78	149,70		ROBERTS	9,07	179,20
DURUM B (DS)	LOG	10,49	164,56	DURUM B (DS)	LOG	18,86	208,89
	SOBEL	7,89	151,17		SOBEL	12,11	190,53
	ROBERTS	6,85	150,61		ROBERTS	9,18	180,92
DURUM B (2D)	LOG	11,55	164,78	DURUM B (2D)	LOG	16,50	211,06
	SOBEL	8,50	147,97		SOBEL	10,75	187,13
	ROBERTS	7,28	144,53		ROBERTS	8,82	183,62
DURUM B (FULL)	LOG	102,51	259,05	DURUM B (FULL)	LOG	142,67	337,91
	SOBEL	66,69	207,55		SOBEL	79,76	261,24
	ROBERTS	51,82	190,13		ROBERTS	59,47	230,97
SABİT BÖLGELER DEVREDE				SABİT BÖLGELER DEVREDE			
DURUM C (TSS)	LOG	16,29	172,28	DURUM C (TSS)	LOG	21,31	211,81
	SOBEL	11,08	152,22		SOBEL	13,86	188,02
	ROBERTS	9,51	145,47		ROBERTS	10,86	183,69
DURUM C (DS)	LOG	11,51	168,44	DURUM C (DS)	LOG	20,03	221,75
	SOBEL	8,41	147,58		SOBEL	12,90	198,95
	ROBERTS	7,53	142,75		ROBERTS	9,97	179,89
DURUM C (2D)	LOG	13,70	166,37	DURUM C (2D)	LOG	18,57	213,50
	SOBEL	9,57	151,19		SOBEL	12,78	187,06
	ROBERTS	8,60	148,11		ROBERTS	9,92	183,97
DURUM C (FULL)	LOG	139,45	292,39	DURUM C (FULL)	LOG	180,76	367,61
	SOBEL	83,86	228,69		SOBEL	109,39	280,91
	ROBERTS	70,43	213,13		ROBERTS	77,70	253,13

açısından bir fark olmakta, toplam süre açısından ise yakın hesaplama süreleri elde edilmiş ve büyük oranlarda değişiklik gözlenmemiştir. Deneylerde kullanılan ayrıt algoritmaları içerisinde Roberts yönteminin kullanılması durumunda diğerlerine göre daha kısa hesaplama zamanları elde edilmiştir.

Bu çalışmada geliştirilen algoritmanın ayrıt haritası kullanıldığı durum için video kalitesinde (24 çerçeve/saniye) hareket takibi yapabileceğini görebilmek amacıyla 30 çerçeve için deneyler yapılmış ve ayrıt haritası kullanılmadığı durum için elde edilen süreler arasındaki farklar Çizelge-2 de sunulmuştur. Çizelge-2 de verildiği gibi algoritma 30 çerçeveyi toplam olarak 9.43 saniyede okumuş, tüm çerçeveleri okunma işlemi bittikten sonra ayrıt algoritması kullanılması durumu için 6.34 saniye içerisinde tüm çerçeveleri işleyerek sonuç üretmiştir. Yani yaklaşık olarak 1 saniye içerisinde 5 çerçeve işlenebilmektedir. Bu sonuç saniyede 24 çerçeve olması gereken video kalitesinden biraz düşüktür. Ancak çerçeveler arasında obje hareketinin izlenebilmesi için elde edilen 5 çerçeve/saniye kaliteyi biraz düşürmekle birlikte yaklaşık her 5 çerçevede bir çerçeve işlenmesi ile izlenebilirliği sürdürmek mümkün olabilmektedir. Çalışmanın gerçekleştirildiği bilgisayar özellikleri geliştirildiğinde işlem sürelerin daha da kısalacağı düşünülmektedir.

Çizelge-2. Algoritmanın uygulanmasından elde edilen ölçüm sonuçları (Measurement results obtained from application of edge and non edge version of algorithm)

Ayrıt Algılama Algoritması Uygulandığında		Ayrıt Algılama Algoritması Uygulanmadığında	
Toplam Çerçeve Okuma Süresi (Saniye)	Toplam Algoritma İşleme Süresi (Saniye)	Toplam Çerçeve Okuma Süresi (Saniye)	Toplam Algoritma İşleme Süresi (Saniye)
9,437	6,842	0,504	82,668
Σ=16,279		Σ=83,172	

Durum B ile Durum C arasındaki arama süresi açısından meydana gelen farklara karşılık ($T_{DurumB}>T_{DurumC}$), TSGO değerleri açısından Şekil 4'de görüldüğü gibi logaritmik ayrıt algılaması dışında Sobel ve Roberts ayrıt algılama algoritmalarının TSGO değerleri benzerlik göstermektedir. Şekil-4'de logaritmik ayrıt algılama algoritması uygulanması durumunda, sabit bölgelerin elemine edilmesi veya edilmemesi koşulları için TSGO değerleri arasındaki fark görülmekte iken, Sobel ve Roberts yöntemlerindeki farkı görebilmek için Şekil 4'deki görüntü TSGO açısından küçük bir bölge için daha ayrıntılı ölçeklendirerek Şekil-5 te gösterilmiştir. Şekil-6'da TSGO değerlerinin ayrıt algılama algoritması uygulanmadığı durumda arama algoritmaları için bir performans karşılaştırması gösterilmiş olup, uygulanan arama algoritmaları için TSGO değerleri karşılaştırıldığında;

$$TSGO_{TamArama}>TSGO_{Eşkenar}>TSGO_{ÜçAdım}>TSGO_{2BoyutluLog.}$$

şeklinde bir sıralama yazılabilir. Şekil-6'da 'tennis' çerçevesinde tam arama, eşkenar biçimli arama, üç adımda arama ve 2 boyutlu logaritmik arama algoritmalarına ait sonuçlar, sırasıyla daire, eşkenar, üçgen ve yıldız karakterleri ile işaretlenmiş Şekil-7 de ise ayrıt algılama algoritması uygulanmaması, Logaritmik, Sobel, Roberts ayrıt algılama algoritmaları uygulanması durumları için TSGO değerleri görünmektedir.

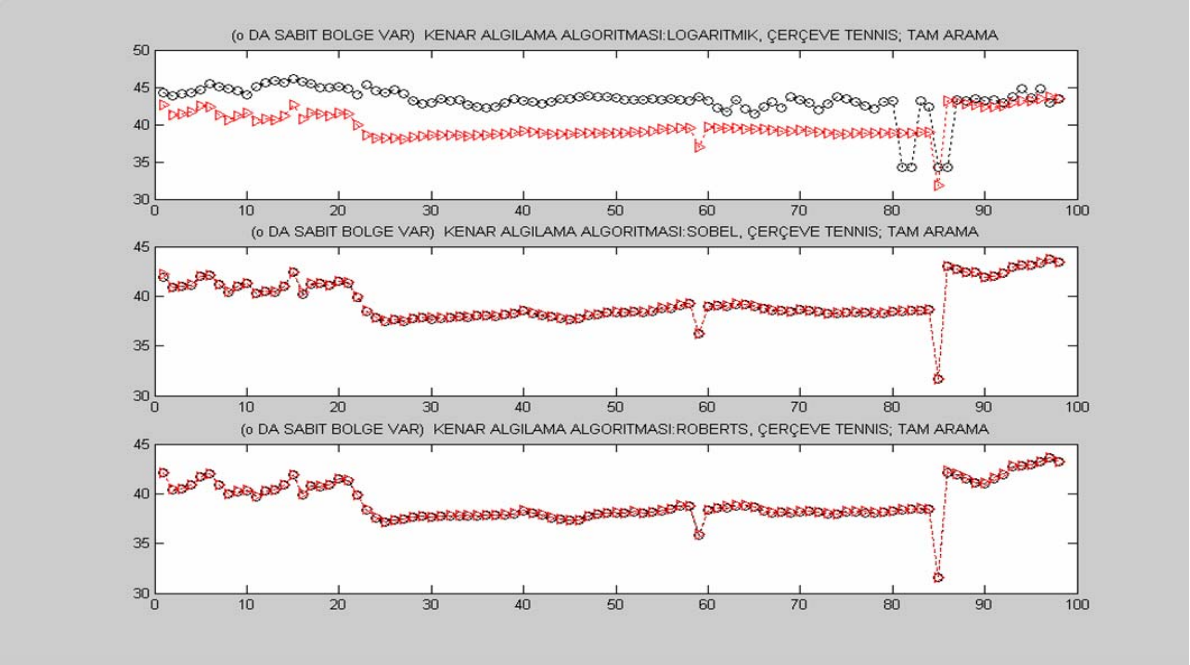
Şekil-7'de de görüldüğü üzere, ayrıt algılama algoritması uygulanmaması (Durum A) ile ayrıt algılama algoritması uygulanması (Durum B) durumunda ve ayrıt algılama algoritması uygulanması durumunda uygulanan yönteme göre TSGO karakteristikleri tam arama, eşkenar biçimli arama, 3 adımda arama ve 2 boyutlu logaritmik aramada benzerlik göstermektedir. Eğer tam arama algoritması kullanılırsa, ayrıt algılama yöntemlerinden Log yöntemi tercih edilebilir. Bunun sebebi, Log ayrıt algılama yöntemi arama ve toplam süre açısından avantaj sağlar ve TSGO açısından Sobel ve Roberts'den belirgin biçimde daha iyi olmasıdır.

Ayrıt bulma algoritması kestirim işlem süresini önemli ölçüde azaltmasının yanında algoritmanın performansına da önemli ölçüde katkı yaptığı gözlenmiştir. Şekil-7 de görüleceği gibi, ayrıt bulma algoritması ile hareketli bölgeler seçildiğinde hata azalmakta ve örnek için TSGO 39 dB seviyelerinde kalırken, uygulanmadığı durumda TSGO ortalama olarak 37,5 dB ve aşağısında kalmaktadır. Ayrıt algılama algoritması uygulandığında, eşik seviyesinin altında kalan ayrıt değerleri işleme katılmamaktadır ve genellikle de gürültü seviyeleri küçük değerlerle ifade edildiklerinden eşik seviyesinin altında kalmaktadır. Bu nedenle sinyal gürültü oranında yukarıda verilen oranda bir artış olduğu gözlenmiştir.

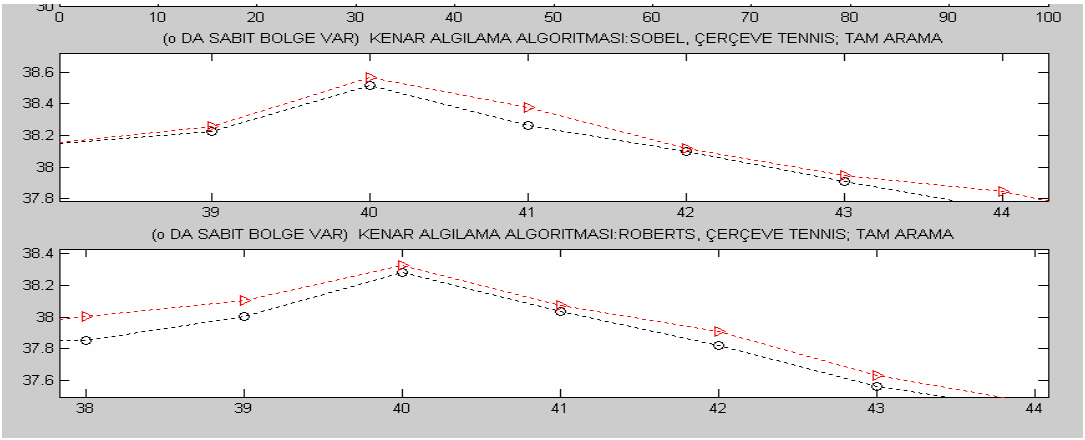
4. TARTIŞMA VE ÖNERİLER (DISCUSSION AND CONCLUSIONS)

Ayrıt bulma algoritması nedeniyle başlangıçta çerçeveleri okuma süresi uzun olmasına karşın, algoritma ayrıt bulma algoritması uygulanmadan elde edilen toplam süreye göre 5 kat daha hızlı sonuç vermektedir. Ayrıca ayrıt bulma algoritması ile birlikte hareket kestiriminde sadece hareket eden nesneye ait öbekler belirlendiğinden, hatalı kestirilen vektörler de azalmıştır. Ancak bu çalışma için ilk çerçevede seçilen öbeğin, ikinci çerçevede yalnızca çevresindeki öbekler araştırıldığından çok hızlı hareket eden objelerde hata oluşabilir. Ancak algoritmanın hızından bir miktar fedakarlıkta bulunularak hareketli cisim çevresindeki arama bölgesini geniş tutulabilir

∇ : Sabit bölgeler işlem dışı, o: Sabit bölgeler var



Şekil-4. Sabit bölgeler elemine edildiği ve edilmediği durum için TSGO karşılaştırılması (PSNR comparisons in the case of whether background elemination applied and not)



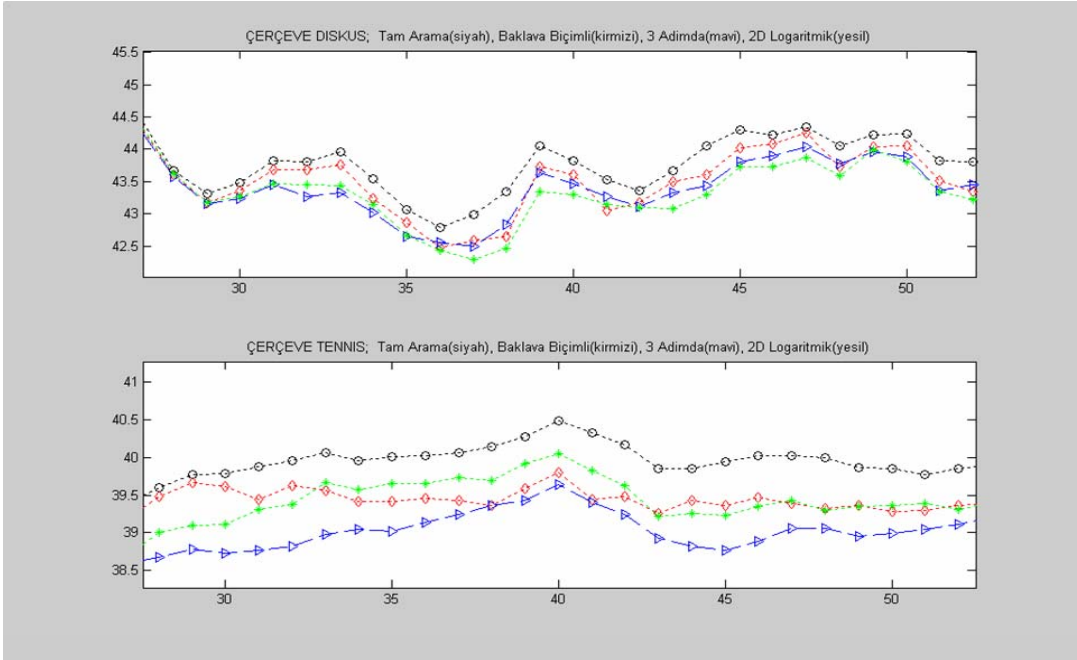
Şekil-5. Sabit bölge elemine edildiği ve edilmediği durum için TSGO karşılaştırılmasına (Şekil-4) detaylı bakış (Detailed view of Figure-4)

ve bu sorun ortadan kaldırılabilir. Ayrıca hareket halindeki cismin koordinatları saklı tutularak, eğer çerçevede hareketli yeni cisimler yoksa tarama yalnızca bu bölgeler içerisinde gerçekleştirilerek daha hızlı bir sonuç elde edilebilir.

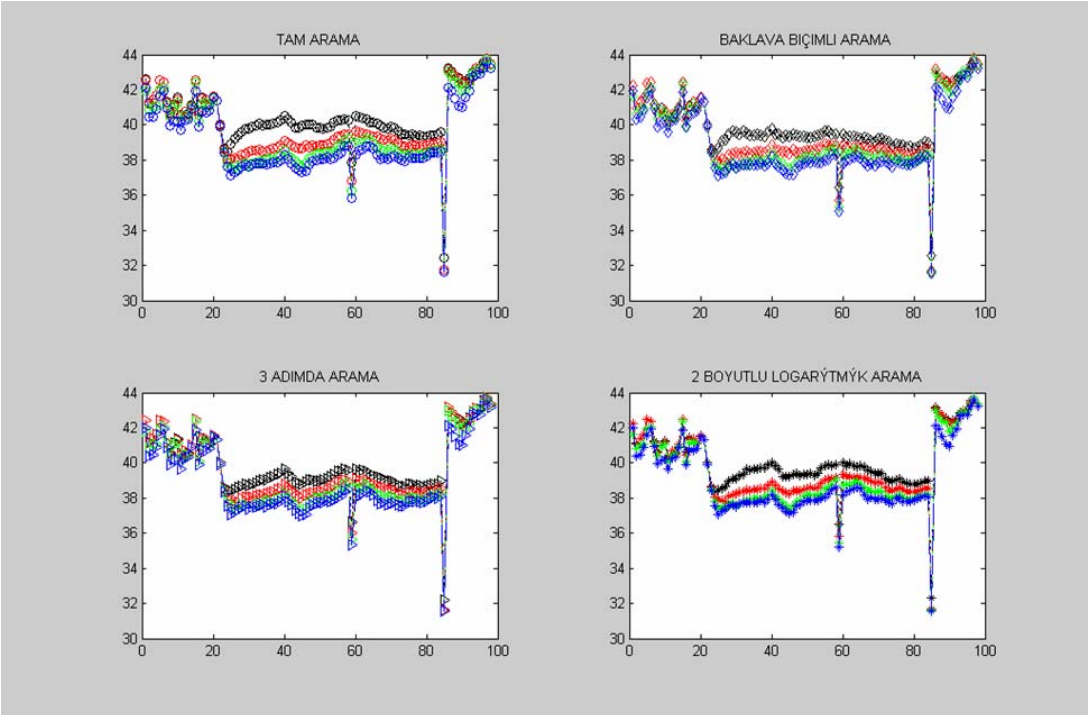
Uygulamada her bir çerçevenin ayrıt haritasının çıkartılması yerine, her iki çerçeveden 1 tanesinin ayrıt haritasının çıkartılarak ileri ve geri yönlü kestirimlerde bulunmak, böylece ayrıt algılama algoritmalarının toplam süreye getirdiği süre dezavantajını yarıya indirmek mümkün olabilir.

KAYNAKLAR (REFERENCES)

1. Urhan O., Erturk S., "Constrained One-Bit Transform for Low Complexity Block Motion Estimation", **IEEE Trans. on Circuits and Systems for Video Technology,** 17 (4), 478–482, 2007.
2. Dufaux F., Konrad J., 'Efficient, robust,and fast global motion estimation for video coding', **IEEE Trans. on Image Proc.,** 9(3), 497-504, 2000
3. Namuduri K.R., 'Motion estimation using spatio-temporal contextual information'**, IEEE Trans. on Circuit and Systems for video technology,** 14(8), 1111-1115, 2004



Şekil-6. Ayrıt algılama algoritması uygulanmadığı durumda TSGO değerlerinin karşılaştırılması (PSNR measurements when edge detection was not applied)



Şekil-7. TSGO değerlerinin ayrıt algılama uygulanması ve uygulanmaması durumu karşılaştırılması (PSNR comparisons in the case of whether edge detection applied and not)

4. Traver V.J., Pla F., 'Similarity motion estimation and active tracking through spatial-domain projections on log-polar images', **CVIU,** 97, 209-241, 2005
5. Telatar Z., "İmge Dizilerinde Objelerin Hareket Yönünün Kestirimi", IEEE SIU-2001 9. **Sinyal İşleme ve Uygulamaları Kurultayı,** S. 641-646, Gazimağusa-Kıbrıs, 2001.
6. Chen M. J., Chen L. G., Chiueh T. D., "One dimensional full search motion estimation algorithm for video coding", **IEEE Trans. Circuits Syst. Video Technol.,** 4, 378-385, 1994.
7. Zhu S., Ma K. K., "A new diamond search algorithm for fast block-matching motion estimation", **IEEE Trans. on Image Proc.,** 7(6), 287-290, 2000.
8. Chau L. P, Xuan J., "Efficient three-step search algorithm for block motion estimation in video coding", **IEEE International Conference on**

Acoustics, Speech, and Signal Processing Proc, (ICASSP '03), 3, 6-10, 2003

9. Koç M. A.., Telatar Z., "Alt-Uzay Yön Algılama ile Görüntü Dizilerinde Hareket Kestirimi", SIU'04 IEEE 12. **Sinyal İşleme ve İletişim Uygulamaları Kurultayı,** S.426-429, Kuşadası, Nisan-2004.
10. Liu S-W., Wei S-D., Lai S-H., "Fast optimal motion estimation based on gradient-based adaptive multilevel successive elimination" **IEEE Transactions on Circuits and Systems for Video Technology,** 18(2), 263-267, 2008.
11. Lim J.S., "**Two dimensional signal and image processing**",PrenticeHall,1990.