# VİDEO GÖRÜNTÜLERİNDE HAREKET YOĞUNLUĞUNUN VE YÖNÜNÜN TESPİTİ

**2011**
**YÜKSEK LİSANS TEZİ**
**ELEKTRONİK VE BİLGİSAYAR EĞİTİMİ**

**Şafak ALTAY**

# VİDEO GÖRÜNTÜLERİNDE HAREKET YOĞUNLUĞUNUN VE YÖNÜNÜN TESPİTİ

**Şafak ALTAY**

**Karabük Üniversitesi**

**Fen Bilimleri Enstitüsü**

**Elektronik ve Bilgisayar Eğitimi Anabilim Dalında**

**Yüksek Lisans Tezi**

**Olarak Hazırlanmıştır**

**KARABÜK**

**Haziran 2011**

Şafak ALTAY tarafından hazırlanan "VİDEO GÖRÜNTÜLERİNDE HAREKET YOĞUNLUĞUNUN VE YÖNÜNÜN TESPİTİ" başlıklı bu tezin Yüksek Lisans Tezi olarak uygun olduğunu onaylarım.

Yrd. Doç. Dr. Salih GÖRGÜNOĞLU

Tez Danışmanı, Bilgisayar Mühendisliği Anabilim Dalı

Bu çalışma, jürimiz tarafından oy birliği ile Elektronik ve Bilgisayar Eğitimi Anabilim Dalında Yüksek Lisans tezi olarak kabul edilmiştir. 21/ 06/ 2011

| Ünvanı, Adı SOYADI (Kurumu) | İmzası |
|---|---|
| Başkan : Doç. Dr. Raif BAYIR (KBÜ) | |
| Üye : Yrd. Doç. Dr. Salih GÖRGÜNOĞLU (KBÜ) | |
| Üye : Yrd. Doç. Dr. Baha ŞEN (KBÜ) | |

...../..../2011

KBÜ Fen Bilimleri Enstitüsü Yönetim Kurulu, bu tez ile, Yüksek Lisans derecesini onamıştır.

Doç. Dr. Nizamettin KAHRAMAN

Fen Bilimleri Enstitüsü Müdürü

*"Bu tezdeki tüm bilgilerin akademik kurallara ve etik ilkelere uygun olarak elde edildiğini ve sunulduğunu; ayrıca bu kuralların ve ilkelerin gerektirdiği şekilde, bu çalışmadan kaynaklanmayan bütün atıfları yaptığımı beyan ederim."*

Şafak ALTAY

# ÖZET

**Yüksek Lisans Tezi**

## VİDEO GÖRÜNTÜLERİNDE HAREKET YOĞUNLUĞUNUN VE YÖNÜNÜN TESPİTİ

**Şafak ALTAY**

**Karabük Üniversitesi**
**Fen Bilimleri Enstitüsü**
**Elektronik ve Bilgisayar Eğitimi Anabilim Dalı**

**Tez Danışmanı:**
**Yrd. Doç. Dr. Salih GÖRGÜNOĞLU**
**Haziran 2011, 58 sayfa**

Hareketin yoğun olduğu ortamlarda insan hareketlerini izlemek güvenliği sağlamak bakımından önemlidir. Bu çalışmada bir kamera ile alınan video görüntülerindeki hareket yoğunluğunun hangi bölgelerde fazla olduğu ve hareketin yönü tespit edilmektedir. Geliştirilen yazılım sayesinde görüntülerde hareket tahmini algoritmaları kullanılarak hareket vektörleri çıkartılmıştır. Hareket vektörlerine göre hareketin yoğun olduğu yerler, k-means, görüntü bölütleme, en yakın komşu algoritmaları ve yarışmacı öğrenme ağı kullanılarak belirlenmiştir. Sisteme eklenen IP (Internet Protocol) kamera, geliştirilen yazılım tarafından otomatik olarak yönlendirilmekte ve hareketli topluluğun izlenmesi sağlanmaktadır.

**Anahtar Sözcükler :** Hareket tahmini, hareket yoğunluğu, video işleme.
**Bilim Kodu :** 702.1.014

## ABSTRACT

**M.Sc. Thesis**

**DETERMINATION OF MOTION INTENSITY AND DIRECTION ON VIDEO SEQUENCES**

**Şafak ALTAY**

**Karabük University**
**Graduate School of Natural and Applied Sciences**
**Department of Electronic and Computer Education**

**Thesis Advisor:**
**Assist. Prof. Dr. Salih GÖRGÜNOĞLU**
**June 2011, 58 pages**

Watching the human movements in environments where high motion intensity exists is important to provide safety. In this study, regions where high motion intensity exists and motion direction on video sequences taken by camera are determined. Using developed software, motion vectors have been created by using motion estimation algorithms on video sequences. According to motion vectors, regions where high motion intensity exists are determined by using k-means algorithm, image segmentation, nearest neighbour algorithm and competitive learning network. IP cam is driven automatic to track moving group of people by developed software.



**Key Words :** Motion estimation, motion intensity, video processing.

**Science Code :** 702.1.014

## TEŞEKKÜR

Bu tez çalışmasının yürütülmesinde ve oluşumunda ilgi ve desteğini esirgemeyen, bilgi ve tecrübelerinden yararlandığım sayın hocam Yrd. Doç. Dr. Salih GÖRGÜNOĞLU' na teşekkürlerimi sunarım.

Manevi destekleriyle beni yalnız bırakmayan, babam Sırrı ALTAY' a, annem F.Nilgün ALTAY' a ve kardeşim Şule ALTAY' a teşekkür ederim.

# İÇİNDEKİLER

**Sayfa**

## ŞEKİLLER DİZİNİ

**Sayfa**

## ÇİZELGELER DİZİNİ

**Sayfa**

## SİMGELER VE KISALTMALAR DİZİNİ

### SİMGELER

Σ : toplam fonksiyonu
μm : mikrometre
μs : mikrosaniye
d : öklid uzaklık değeri
K : kırmızı parlaklık seviyesi
k : küme sayısı
M : mavi parlaklık seviyesi
MB : makro blok
ms : milisaniye
n: : ağırlık azaltma oranı
N : makro blok boyutu
P : gri seviye değeri
w : ağırlık
Y : yeşil parlaklık seviyesi

### KISALTMALAR

ADALINE : adaptive linear neuron (adaptif doğrusal nöron)
AVI : audio video interleave (ses görüntü birleşimi)
HKT : hata kareleri toplamı
IP : internet protocol (internet protokol)
KYM : kırmızı-yeşil-mavi
MADALINE : multiple adaline (çoklu adaline)
MFT : mutlak fark toplamı
MPEG : moving pictures experts group (hareketli görüntü uzmanlar birliği)
PDA : personal digital assistant (kişisel dijital asistan)

# BÖLÜM 1

# GİRİŞ

Görüntü dizilerinde hareket tahmini algoritmaları kullanarak hareketli nesnelerin takip edilmesi bilgisayarlı görme uygulamalarındaki önemli konulardan biridir. Örneğin askeri uygulamalar kapsamında hareketli bir hedefin takip edilerek imha edilmesi ulusal güvenlik için akıllı silahların geliştirilmesi açısından büyük bir önem taşımaktadır. Benzer şekilde insanların toplu halde bulunduğu metro yada terminal gibi güvenliğin önemli olduğu ortamlarda insan aktivitelerinin otomatik olarak yorumlanabilmesi, insanları algılama ve takip etme yeteneğine sahip görmeye dayalı, sağlam ve güvenilir bir sistemin kurulmasıyla sağlanabilir. Etkileşimli çoklu ortam sistemleri geliştirebilmek amacıyla insan bilgisayar etkileşimi için görmeye dayalı arabirimler de insan ve nesnelerin takip edilmesini gerektirir [1].

Görüntü dizilerinde çerçeveler arasındaki hareket bilgilerini tanımlamak ve hareket yönünü saptamak görüntü analizinin temel problemlerindendir. Bu gibi hareketli nesneleri bulunduran dizilerde bir çerçevedeki piksel değerleri, hareketin ardışıklığına bağlı olarak ikinci çerçevede farklı bir konuma transfer edilebilirler. Bu işlem piksel korunumu olarak bilinir ve bir çerçevedeki nesnelerin ikinci çerçevedeki pozisyonunu oluşturmak üzere değişik yönlere hareket etmesi anlamına gelir. Böylece, birbirini takip eden çerçeveler arasındaki anlamlı değişimler belirlenerek hareketin tahminine olanak sağlanır. Genelde uygulamalarda çerçeve içerisindeki tek bir objenin ileri-geri ya da yukarı-aşağı hareketi kestirilir ve hareket izlenmeye çalışılır. Fakat bir çerçeve içerisinde birden fazla hareketli nesne olan karmaşık görüntü dizilerinde tek bir objeye ilişkin hareketin izlenmesi pek pratik değildir. Bunun için birden fazla hareketli objeyi bulunduran dizilerde objelerin hareketleri hakkında daha fazla bilgi sağlayabilecek algoritmalara ihtiyaç duyulur. Diğer bir sorun nesnenin hareketindeki değişiklikler nedeniyle kameraya daha önce yansımayan nesnenin ya da başka nesnelere ait bazı yeni bölümlerinin ortaya çıkmasıdır. Bu da ikinci çerçevede büyük değişimlerin olmasına neden olabilir.

Ayrıca aynı çerçeve içerisinde farklı yönlere hareket edebilen cisimler de bulunabilir ve eğer hareket yönleri belirlenmemişse hareketin hangi cisme ait olduğu karmaşıklığına neden olabilir. Bu nedenle hareketli objelerde, hareketin yönünü belirleyen vektörlerin ortaya konulması ve çerçeveler arasındaki değişimlerin bu vektörler doğrultusunda izlenmesi yanılgı payını büyük ölçüde azaltacaktır [2].

Hareket analizi ve hareket tahmini ile ilgili birçok akademik çalışma yapılmıştır. Bu çalışmalardan bazıları sadece tek bir insanın yaptığı hareketleri incelemiştir. Koşmak ve yürümek gibi hareketleri inceleyen çalışmaların yanında tek bir insanın yaptığı spor aktivitelerinin şeklini inceleyen çalışmalarda yapılmıştır. İnsanları grup olarak ele alıp, bu bir grup insanın hareketini değerlendirip, takip eden çalışmalarda bulunmaktadır [3-5].

Ayrıca trafikdeki araç hareketlerini izleyerek trafiğin nerelerde, ne kadar yoğun olduğunu bulmaya çalışan çalışmalarda yapılmaktadır [6-8]. Araçların havadan lazer darbeleri kullanılarak elde edilen görüntüleri üzerinden hareket tahmini yapan bir çalışmada bulunmaktadır [9]. Pasha, dinamik bir uyarlanabilir kestirim modeli olan etkileşimli çoklu model algoritmasını derlemiş ve benzetimi yapılan hareketli sivil hava trafik hedefine uygulamıştır. Benzetimi yapılan harekete en yakın kestirimin etkileşimli koordineli dönüş çoklu modeli olduğunu ortaya koymuştur [10].

Hareket tahmini tıp ile ilgili alanlarda da yaygın olarak kullanılmaktadır. Ultrasonik görüntüleme ve manyetik rezonans görüntüleri için hareket tahmini yapan çalışmalar bulunmaktadır [11-13]. İnsan sağlığı ile ilgili yapılan çalışmaların yanında, kamera ve mekanik sistemler kullanılarak hareketli cisimleri izleme uygulamaları da bulunmaktadır [14].

Ogawara ve arkadaşları uzun görüntü dizilerinde tekrar eden hareket örüntülerini hareket yoğunluğu kullanarak tespit eden bir çalışma yapmışlardır. Bir kaç farklı nesne kullanarak oluşturdukları hareketleri çalışmalarında kullanmışlardır. Dört farklı veri seti için değerlendirmelerini sunmuşlardır [15].

Telatar görüntü dizilerindeki çoklu obje hareketini kestirmede ve algılamada kullanılabilecek ayrıt sezmeye dayalı hızlı bir algoritma sunmuştur. Yöntem çerçeveler arasında görüntünün satır ve sütunları boyunca piksel değerlerindeki kaymalarına bağlı olarak vektör yansımalarının hesaplanması ve bu yansımaların çerçevelere yansıtılması ile ilgilidir ve beş kat daha hızlı sonuç vermektedir [16].

Katadioptrik gibi farklı kamera çeşitlerini kullanan çalışmaların yanında birden fazla kamera kullanarak hareket tahmini yapan çalışmalarda bulunmaktadır [17-19].

Görgünoğlu ve arkadaşları tüm arama algoritması kullanarak buldukları hareket vektörlerini k-means algoritması ile kümelendirip baskın hareket yönünü tahmin eden bir çalışma yapmışlardır. Kullanılan algoritmaları süre bakımından değerlendirmişlerdir [20].

Bu tez çalışmasında video görüntülerindeki birkaç insanın ya da insan topluluklarının hareketi algılanmış ve takip edilmiştir. Bu işlemler için öncelikle görüntü dizilerindeki hareket vektörleri oluşturulmuş ve bunlara göre hareket yoğunluğunun fazla olduğu bölgeler tespit edilmiştir. Geliştirilen sistemde hareket yoğunluğunun fazla olduğu bölgeler, IP kamera odaklatılarak gerçek zamanlı olarak izlenebilmektedir.

Tezin ikinci bölümünde görüntü dizilerinin özelliklerinden ve hareket tahmininde kullanılan blok eşleştirmeye dayanan algoritmalar hakkında bilgi verilmektedir. Üçüncü bölümde hareket yoğunluğunun tespit edilmesi için gerekli olan ön işlemler, kümeleme yöntemleri ve yapay sinir ağları hakkında temel bilgi verilmektedir. Dördüncü bölümde gerçekleştirilen yazılım sistemin de hareket vektörlerinin elde edilmesi, hareket yoğunluğunu belirleme, IP kameranın kontrolü ve yazılımın ara yüzü hakkında bilgi verilmektedir.

Son bölümde geliştirilen yazılımda kullanılan algoritmaların ortalama işlem süreleri verilmektedir. Hareket yoğunluğunu belirlemek için kullanılan algoritmalar performans bakımından değerlendirilmektedir. Ayrıca hareket yoğunluğunu ve yönünü tespit etme konusunda yapılacak çalışmalar hakkında öneriler de sunulmaktadır.

# BÖLÜM 2

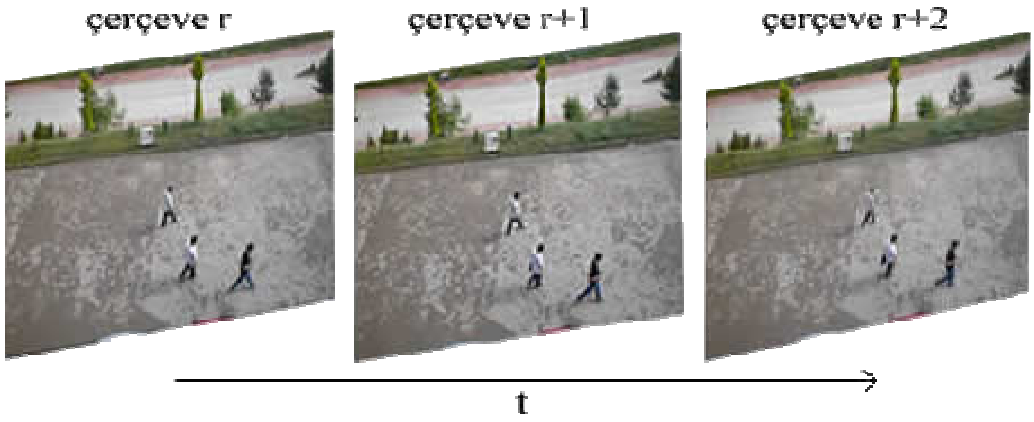
# HAREKET ANALİZİ

Hareket analizi, görüntü dizileri içinde statik hareket ve obje veya kameranın hareketi sonucu oluşan değişimlerin tanımlanması ve algılanmasına dayanır. Hareket tahmini, bir video görüntüsü içerisinde ardışık çerçevelerde hareket eden objelerin hareket bilgilerinin tanımlanmasıdır. Sonuçta hareket analizi, hareket tahmini ve izleme için çerçeveler arasındaki eşleşen noktaları belirlemek ve hareketin dinamiğini ortaya koymaktır [16].

## 2.1. GÖRÜNTÜ DİZİLERİ

Sayısal bir görüntü nesneler tarafından yansıtılan ışık enerjisinin bir algılayıcı tarafından öngörülen elektromanyetik aralıkta algılanarak sayısal sinyal haline dönüştürülmesi ile oluşturulur. Bir görüntünün temel bileşeni pikseldir. Sayısal bir görüntü mxn boyutlu piksellerden oluşan bir matris ile ifade edilir. Gri tonlu görüntülerde, görüntü farklı gri ton değerlerinden oluşur. Gri değer aralıkları (0,1,2…255) şeklinde ifade edilir. Bunun anlamı, bir gri tonlu görüntüde 256 tane farklı gri ton değeri bulunabilir. Elektromanyetik spektrumda 0,4-0,5 μm dalga boyu mavi renge; 0,5-0,6 μm dalga boyu yeşil renge; 0,6-0,7 μm dalga boyu kırmızı renge karşılık gelir. Bu dalga boylarında elde edilmiş üç gri düzeyli görüntü bilgisayar ekranında sırası ile kırmızı-yeşil-mavi kombinasyonun da üst üste düşürülecek olursa renkli görüntü elde edilmiş olur [21].

Görüntü dizileri bir kamera yardımıyla elde edilen ardışık görüntüler bütünüdür. Görüntü dizileri MPEG (Moving Pictures Experts Group) veya AVI (Audio Video Interleave) gibi formatlara sahiptir. Şekil 2.1'de bir görüntü dizisinin yapısı gösterilmektedir. Burada t zamanı, r görüntü dizisinin hangi çerçevede olduğunu temsil etmektedir.




Şekil 2.1. Görüntü dizisi.

## 2.2. HAREKET TAHMİNİNDE KULLANILAN BLOK EŞLEŞTİRMEYE DAYANAN ALGORİTMALAR

Hareket tahminin de kullanılan birçok farklı algoritma vardır. Bu algoritmalardan bir kısmı blok eşleştirmeye dayanmaktadır. Blok eşleştirmeye dayanan algoritmalar basit olmaları nedeni ile en çok kullanılan hareket tahmini algoritmalarıdır.

Blok eşleştirmeye dayanan algoritmalar da ilk önce karşılaştırılacak görüntüler bloklara ayrılmaktadır. Bu bloklara "makro blok" denmektedir. Blok eşleştirme işleminde bu bloklar kullanılmaktadır. İlk görüntüdeki eşleştirilecek makro blok, diğer görüntüde arama bölgesi içindeki bloklarla karşılaştırılır. En uygun makro blok bu karşılaştırma işlemi sonunda tespit edilir.

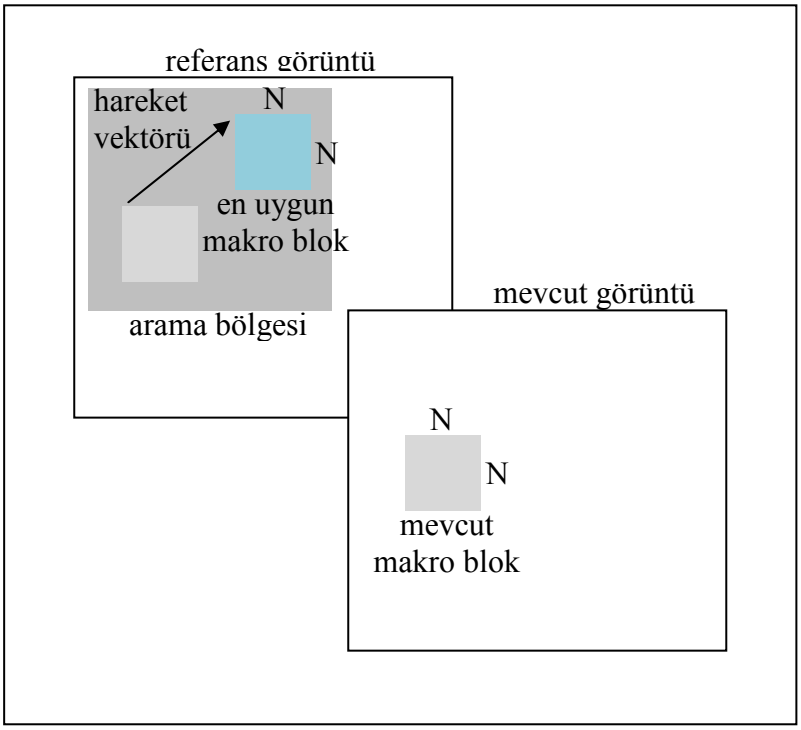
Blok eşleştirmeye dayanan algoritmalarda, karşılaştırma işlemi için farklı matematiksel yöntemler kullanılmaktadır. En basit yöntem piksel farkları toplamıdır. Makro blokdaki her bir piksel değeri, karşılaştırılan alandaki aynı pozisyondaki piksel değerinden çıkartılarak, tüm farkların toplamı hesaplanmaktadır. Sonucu 0'a en yakın olan karşılaştırılan alan, en çok benzerlik taşımaktadır. Bu yönteme Mutlak Fark Toplamı (MFT) denir. NxN boyutların da ki iki makro blok arasında MFT Eşitlik 2.1' de verildiği gibi hesaplanır. Eşitlikte ki $MB_1$ ilk görüntüde ki karşılaştırılacak makro bloğu, $MB_2$ ikinci görüntüde ki arama bölgesindeki bloğu göstermektedir. Makro bloğun piksellerinin koordinatları (x,y) olarak belirtilmektedir.

$$\mathrm{MFT} = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} |\mathrm{MB}_1(x, y) - \mathrm{MB}_2(x, y)| \quad (2.1)$$

Arama bölgesinde yapılan arama işleminin başarılı olması için bir çok farklı arama algoritması geliştirilmiştir. Bu algoritmalardan bir kısmı tüm arama, üç adımlı arama, baklava biçimli arama ve hiyerarşik arama algoritmalarıdır.

### 2.2.1. Tüm Arama Algoritması

Uzaysal düzlemde hareket tahmini yapan bu algoritmada, mevcut makro blok arama bölgesindeki en uygun makro blokla eşleştirilir ve hareket vektörü oluşturulur. Şekil 2.2'de tüm arama algoritmasının işleyişi gösterilmektedir.



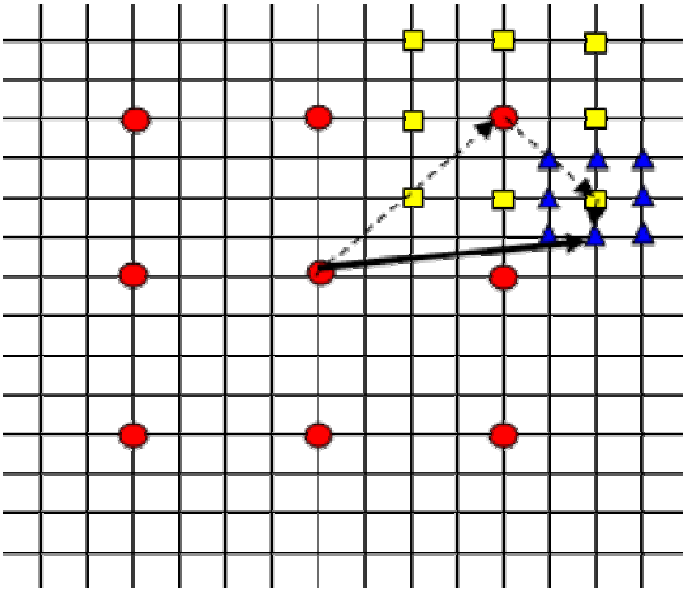

Şekil 2.2. Tüm arama algoritması [22].

Arama bölgesindeki her bir olasılık için karşılaştırma işlemi yapıldığından çok sayıda matematiksel işlem gerektirir. Nesnenin hareketi görüntüleri alan kameraya göre yatay veya dikey düzlemde olduğunda iyi sonuç elde edilir. Görüntü orijini etrafında dönmesi durumunda veya kameraya yaklaşması durumunda arama penceresine göre açısı değişmiş veya büyümüş olacağından referans görüntü

üzerindeki alan ile aynı sayıda piksel değerlerine sahip olmaz bu da algoritmayı yanıltır. Bu yüzden arama uzayını açıya bağlı olarak ve z ekseni boyunca da genişletmek gerekir. Bu da karşılaştırma sırasındaki matematiksel işlemleri daha da karmaşık hale getirmektedir [22].

### 2.2.2. Üç Adımlı Arama Algoritması

Üç adımlı arama hızlı blok eşleştirme algoritmalarının en eskilerindendir. Ortaya çıkma tarihi 1980'li yıllara dayanmaktadır. Üç adımlı arama algoritmasının genel yapısı ve oluşan hareket vektörü Şekil 2.3'de gösterilmektedir [23].

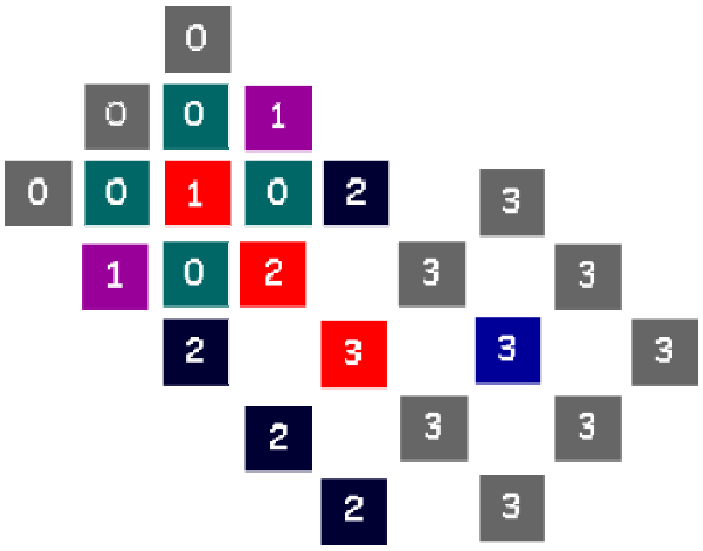


Şekil 2.3. Üç adımlı arama algoritması [23].

Üç adımlı arama algoritması arama işlemine arama alanının merkezinden başlar. İlk adımdan sonra en uygun makro bloğun çevresinde arama yapar. Son adımda ikinci adımda bulduğu en uygun makro bloğun çevresinde arama yapar.

### 2.2.3. Baklava Biçimli Arama Algoritması

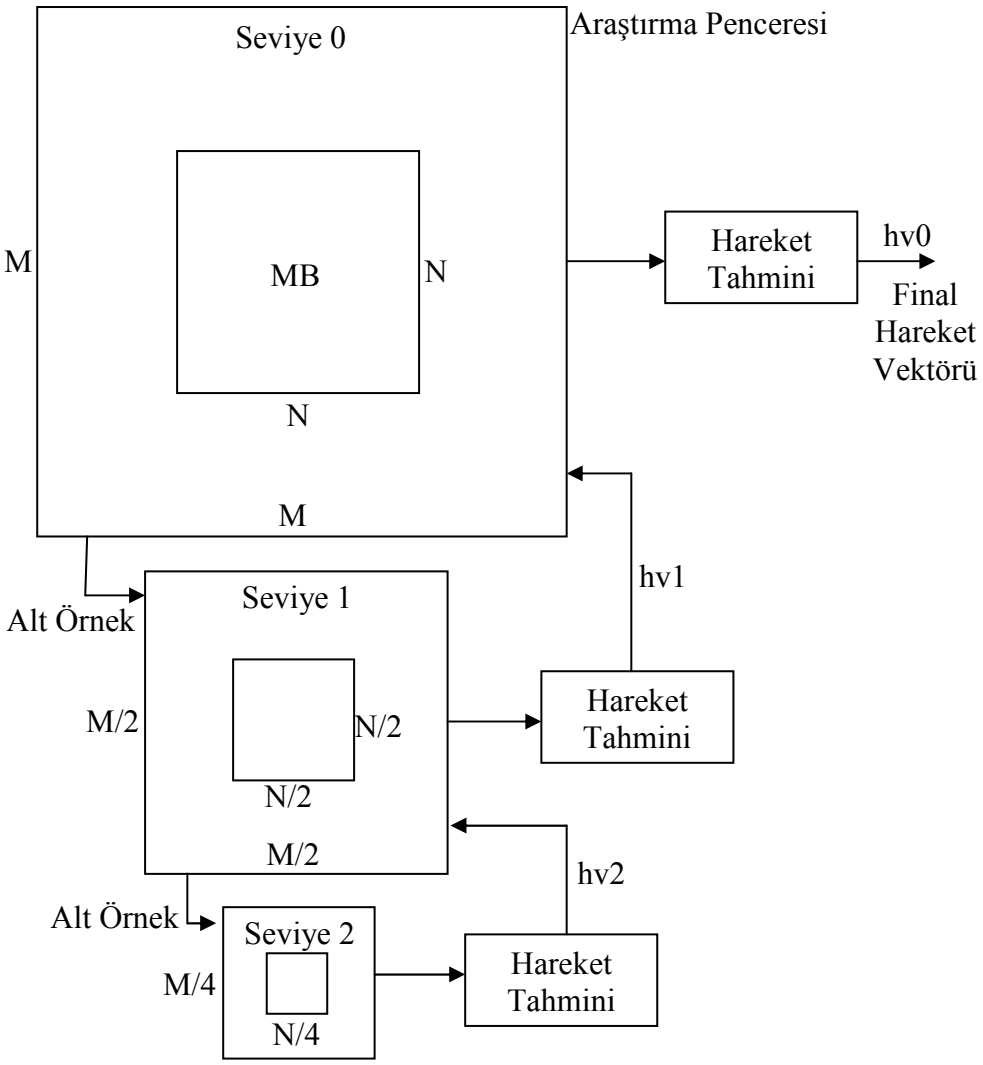
Baklava biçimli arama algoritması üç adımlı arama algoritmasına benzemektedir. Diğer algoritmalardan en önemli farklarından biri adım sayısında bir sınırlandırma olmamasıdır. Son adımın diğer adımlardan farkı o adımda daha fazla alanın taranmasıdır. Son karşılaştırma işleminden sonra belirlenen en düşük karşılaştırma kriteri değerine sahip pozisyon vektör yönünü belirler. Algoritmanın işlem adımları Şekil 2.4'da gösterilmektedir. Şekilde ki 0 adımı son adımdır [24].




Şekil 2.4. Baklava biçimli arama algoritması [24].

### 2.2.4. Hiyerarşik Arama Algoritması

Hiyerarşik arama algoritması daha az sayıda arama bölgesi kullanarak ve mutlak fark toplamı metodunu daha az sayıda piksel üzerinde kullanarak tüm arama algoritmasına göre hareket tahmini işlemini hızlandırmıştır. Algoritma arama bölgesindeki piksellerin sayısını, makro bloğu ve arama alanını alt kümeleyerek azaltır. 3-seviye bir hiyerarşik arama algoritması Şekil 2.5'de gösterilmektedir [25].




Şekil 2.5. Hiyerarşik arama algoritması [25].

# BÖLÜM 3

# HAREKET YOĞUNLUĞUNUN TESPİTİ

Hareket yoğunluğunun tespiti hareket vektörlerinin konumuna göre yapılmaktadır. Hareket vektörlerinin oluşturulmasının hızlandırılması için görüntü dizilerine öncelikle bazı ön işlemler uygulanmalıdır. Hareket yoğunluğunun görüntünün neresinde fazlalaştığının tespiti için uygun kümeleme yöntemi uygulanabilir.

## 3.1. ÖN İŞLEMLER

Görüntünün işlenmesini kolaylaştırmak için uygulanan işlemlere ön işlemler denir. Bunlardan bazıları gri tonlama, kenar bulma, eşikleme ve gürültülerin temizlenmesidir.

### 3.1.1. Gri Tonlama

Renkli görüntünün her bir pikselinin kırmızı, yeşil ve mavi değerlerinin belli bir sabitle çarpılıp görüntünün gri tonlu hale getirilmesidir. Eşitlik 3.1'de, P dönüştürülmüş gri seviye değerini, K kırmızı parlaklık seviyesini, Y yeşil parlaklık seviyesini ve M mavi parlaklık seviyesini göstermektedir [26].

$$P=0,299xK+0,587xY+0,114xM \tag{3.1}$$

Şekil 3.1 a'da orijinal görüntü ve Şekil 3.1 b'de gri tonlu hale çevrilmiş görüntü gösterilmektedir.

(a) (b)

Şekil 3.1. Gri tonlu hale çevirme a) orijinal görüntü ve b) gri tonlu hale dönüştürülmüş görüntü.

### 3.1.2. Kenar Bulma

Görüntüye uygulanan çeşitli filtreler sayesinde kenar bulma işlemi gerçekleştirilmektedir. Bu işlem için çeşitli operatörler kullanılmaktadır. Sobel operatörleri de bunlardan biridir. Kenar bulma işleminde operatördeki katsayılar piksel komşuluklarına göre doğrudan kendi konumlarına denk gelen değerler ile çarpılmaktadır. Şekil 3.2 a'da piksel komşulukları, Şekil 3.2 b'de sobel yatay operatörü ve Şekil 3.2 c'de sobel dikey operatörü gösterilmektedir.

| | | |
|---|---|---|
| $z_1$ | $z_2$ | $z_3$ |
| $z_4$ | $z_5$ | $z_6$ |
| $z_7$ | $z_8$ | $z_9$ |

(a)

| | | |
|---|---|---|
| 1 | 0 | -1 |
| 2 | 0 | -2 |
| 1 | 0 | -1 |

(b)

| | | |
|---|---|---|
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

(c)

Şekil 3.2. Kenar bulma işleminde kullanılanlar a) piksel komşulukları, b)sobel yatay operatörü ve c) sobel yatay operatörü.

Pikselin x yönündeki gradyan $G_X$ değeri ve y yönündeki gradyan $G_Y$ değeri sırası ile Eşitlik 3.2'de ve Eşitlik 3.3'de verildiği gibi hesaplanır. Piksele atanacak gradyan değeri G Eşitlik 3.4'de verildiği gibi hesaplanır.

$$Gx = Z_1 + 2Z_4 + Z_7 - Z_3 - 2Z_6 - Z_9 \quad (3.2)$$

$$Gy = Z_1 + 2Z_2 + Z_3 - Z_7 - 2Z_8 - Z_9 \quad (3.3)$$

$$G = \sqrt{{G_x}^2 + {G_y}^2} \quad (3.4)$$

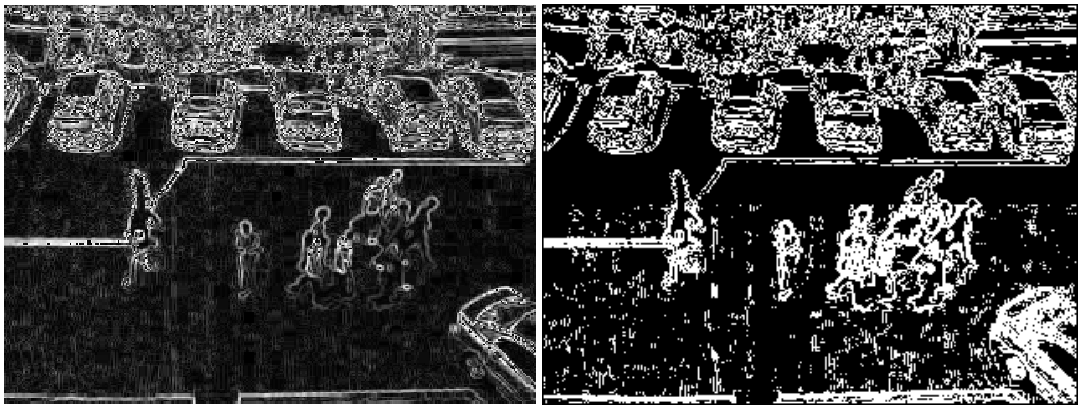
Şekil 3.3 a'da orijinal görüntü ve Şekil 3.3 b'de kenarların bulunduğu görüntü gösterilmektedir.

(a) (b)

Şekil 3.3. Kenar bulma işlemi a) orijinal görüntü ve b) kenarların bulunduğu görüntü.

### 3.1.3. Eşikleme

Gri tonlu hale dönüştürülmüş görüntünün her bir piksel renk değeri belli bir eşik değerle karşılaştırılır. Bu eşik değerden düşük olan piksellere 0, yüksek olanlara 255 değeri verilerek ikili görüntü oluşturulmuş olur. Bu işleme eşikleme (thresholding) denir. Eşik değerinin belirlenmesi için farklı yöntemler kullanılmaktadır. Rastgele bir değer seçilebileceği gibi otsu eşikleme yöntemi de kullanılabilir. Piksellerin hesaplanan ortalama değeri de eşik değer olarak kullanılabilir. Şekil 3.4 a'da orijinal görüntü ve Şekil 3.4 b'de eşiklenmiş görüntü gösterilmektedir.

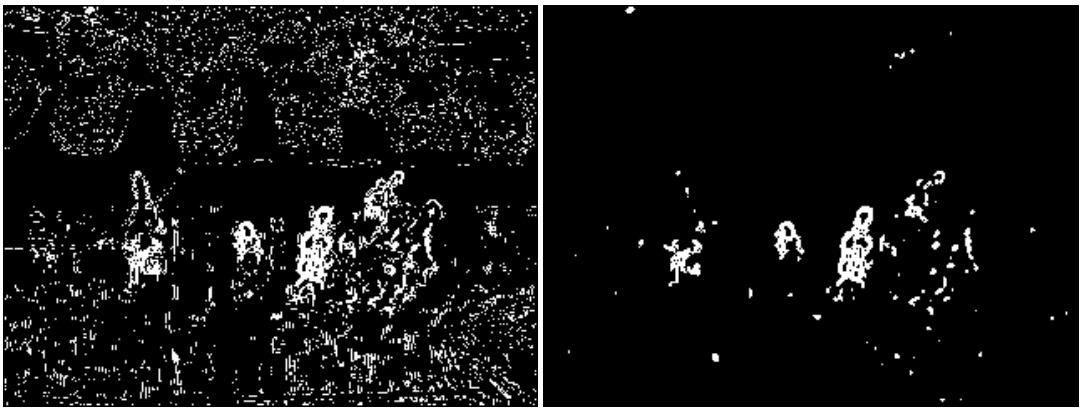


(a) (b)

Şekil 3.4. Eşikleme işlemi a) orijinal görüntü ve b) eşiklenmiş görüntü.

### 3.1.4. Gürültülerin Temizlenmesi

Görüntülerde istenmeyen gürültüler oluşabilmektedir. Bu gürültüleri temizlemek için farklı filtreleme yöntemleri kullanılmaktadır. Bu tez çalışmasında gürültüleri temizlemek için her bir beyaz pikselin çevresindeki kendisi de dahil dokuz piksele bakılmıştır. Bu dokuz pikselden dört tanesi veya daha azı beyaz ise o piksel siyah yapılmıştır. Böylelikle gürültüler büyük ölçüde temizlenmiştir. Şekil 3.5 a'da orijinal görüntü ve Şekil 3.4 b'de gürültüleri temizlenmiş görüntü gösterilmektedir.



(a) (b)

Şekil 3.5. Gürültülerin temizlenmesi a)orijinal görüntü ve b) gürültüleri temizlenmiş görüntü.

## 3.2. KÜMELEME YÖNTEMLERİ

Kümeleme birbirine benzeyen veri parçalarını ayırma işlemidir. Kümele yöntemlerinin çoğu veriler arasındaki uzaklıkları kullanır. Örneğin Öklid, Manhattan ve Minkowski bağlantıları kümeleme işleminde alt işlem olarak uzaklık hesaplamada kullanılmaktadır. Kümeleme yöntemleri hiyerarşik olanlar ve olmayanlar olarak iki bölüme ayrılır [27].

Hiyerarşik kümeleme yöntemleri, verileri ağaç yapısı şeklindeki gruplar içerisinde kümeleyerek çalışırlar. Hiyerarşik yöntemler küme sayısını belirten k değerine ihtiyaç duymazlar. Onun yerine ağaç yapısını oluşturma işleminin ne zaman durduracağını belirten bir eşik değerini bilmeye ihtiyaç duyarlar. Hiyerarşik olmayan kümeleme yöntemleri veri tabanında bulunan n adet veriyi belirlenmiş olan k sayıda kümeye ayırmaya çalışmaktadır [28].

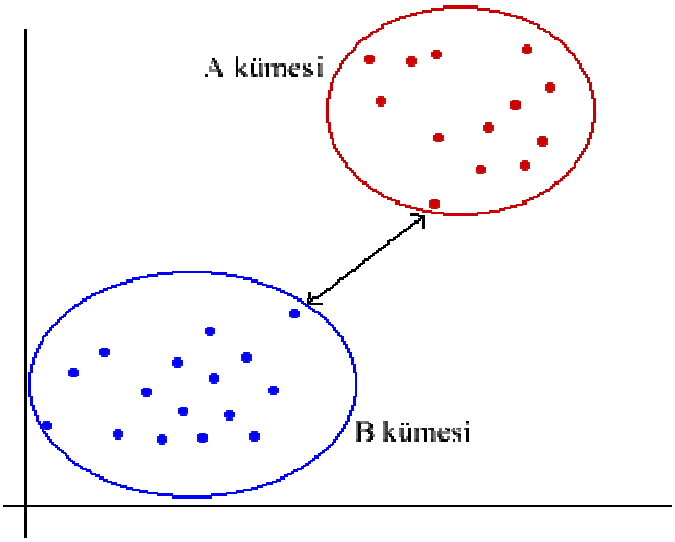
### 3.2.1. En Yakın Komşu Algoritması

En yakın komşu algoritması hiyerarşik kümeleme yöntemlerinden biridir. En yakın komşu yöntemine "tek bağlantı kümeleme yöntemi" adı da verilmektedir. Başlangıçta tüm gözlem değerleri birer küme olarak değerlendirilir ve adım adım bu kümeler birleştirilerek yeni kümeler elde edilir.

Bu yöntemde öncelikle gözlemler arasındaki uzaklıklar belirlenir. i ve j gözlemleri arasındaki uzaklıkların belirlenmesinde Eşitlik 3.5' de verilen Öklid uzaklık bağıntısı kullanılabilir. Eşitlik de ki p gözlem değerlerinin boyutunu, d(i,j) i ve j gözlemleri arasındaki uzaklığı belirtmektedir.

$$d(i,j) = \sqrt{\sum_{k=1}^{p}(x_{ik} - x_{jk})^2} \tag{3.5}$$

Uzaklıklar göz önüne alınarak minimum d(i,j) değeri seçilir. Söz konusu uzaklıkla ilgili gözlemler birleştirilerek yeni bir küme elde edilir [27].




Şekil 3.6. En yakın komşu algoritması [27].

Şekil 3.6.'da gösterildiği gibi en yakın komşu algoritmasında iki kümenin birbirine en yakın gözlemleri arasındaki uzaklık iki kümenin birbirine olan uzaklığı olarak değerlendirilir.

### 3.2.2. K-means Algoritması

Hiyerarşik olmayan kümeleme yöntemlerinden biridir. En iyi bilinen ve yaygın kullanılan algoritmalardan biri olan k-means, verileri sınıflandıran bir kümeleme algoritmasıdır. Verilen nesneleri özelliklerine göre k adet sınıfa ayırmak amacıyla kullanılır. Sınıflandırma, verilerin en yakın veya benzer oldukları küme merkezleri etrafına yerleştirilmesi ile gerçekleştirilir. MacQueen tarafından 1967 yılında geliştirilmiştir. Bu yöntem yıllardır bilimsel ve endüstriyel uygulamalarda kullanılmaktadır. Küme sayısı k ile gösterilir ve elemanlarının birbirlerine olan yakınlıklarına göre oluşacak grup sayısını ifade eder. Buna göre k önceden bilinen ve kümeleme işlemi bitene kadar değeri değişmeyen sabit bir pozitif tamsayıdır [29].

K-means algoritması şu adımları içermektedir;

a) k (küme sayısı) değeri belirlenir.
b) Her bir kümenin merkezi rastgele belirlenir. Bunun için gözlem değerleri arasından k değeri kadar nokta seçilir.
c) Merkez değerleri ile gözlem değerleri arasındaki uzaklıklar hesaplanır. Bir gözlem değeri hangi merkeze yakın ise o kümeye dahil edilir.
d) Oluşan kümelerin yeni merkezleri o kümedeki tüm gözlem değerlerinin ortalama değeri olarak değiştirilir.
e) Kümelerin merkez noktaları değişmeyene kadar c ve d adımları tekrar edilir.

K-means algoritmasının zayıf yanları şu şekilde özetlenebilir [28,29];

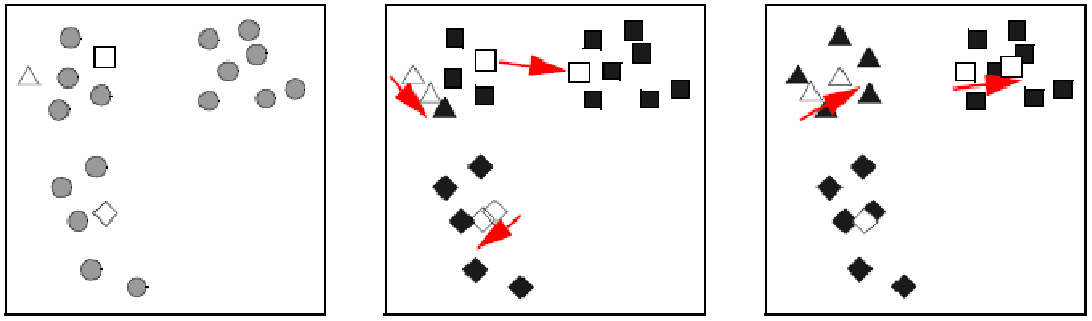
- Algoritmanın başında giriş parametresi olarak bir k sayısına ihtiyacı vardır. Elde edilecek sonuçlar k sayısına göre değişkenlik gösterebilmektedir.
- Aşırı gürültü ve istisna veriler algoritmayla hesaplanan ortalamayı değiştirdiği için k-means algoritması gürültü ve istisnaya aşırı duyarlıdır.
- K-means algoritması sadece sayısal veriler ile kullanılabilmektedir. Kategorik verilerin kümelenmesi için k-means algoritması bir çözüm sunamamaktadır.
- Çakışan kümelerde iyi sonuç vermez.
- Her eleman aynı anda verilen bir kümenin içindedir veya dışındadır.

K-means kümeleme yönteminin değerlendirilmesinde farklı yöntemler kullanılır. En yaygın olanı Hata Kareleri Toplamı (HKT) yöntemidir. En küçük Hata Kareleri Toplamı değerine sahip kümeleme en iyi sonucu verir. Eşitlik 3.6'da gösterildiği gibi hesaplanır. Eşitlik de ki $x_j$ değeri $Q_i$ kümesine ait olan j. veridir. $M_i$, i. kümenin merkez noktasıdır.

$$HKT = \sum_{i=1}^{k} \sum_{x_j \in Q_i} \left| x_j - M_i \right|^2 \tag{3.6}$$

Şekil 3.7'de K-means algoritmasının çalışma şekli gösterilmektedir. Şekil 3.7 a'da beyaz renkli simgeler rastgele seçilen küme merkezlerini temsil etmektedir. Sekil 3.7

b'de geri kalan siyah renkli noktalar aynı şekilli ve beyaz renk olan küme merkezlerine dahil edilerek ilk kümeler oluşturulur. Bu işlem sonunda küme merkezleri her kümedeki elemanların ortalaması dikkate alınarak tekrar hesaplanır. Değişen küme merkezleri sekil 3.7 b'de kırmızı oklar ile gösterilmiştir. Sekil 3.7 c'de aynı işlem tekrar edildiğinde küme merkezlerinin değişimi görülmektedir. Bu şekilde başlangıç durumunda rastgele seçilen küme merkezleri, sürekli yinelemeler ile gerçek kümelenme alanlarının ortasına doğru yaklaşır. Bu işleme merkeze yakınsama denir. Merkeze yakınsama minimum seviyeye geldiğinde veya durduğunda kümeleme işlemi sona erer [29].



(a) (b) (c)

Şekil 3.7. K-means algoritması a) ilk küme merkezleri, b) ilk ortalama hesabı ve c) merkez yakınsama [30].

## 3.3. YAPAY SİNİR AĞLARI

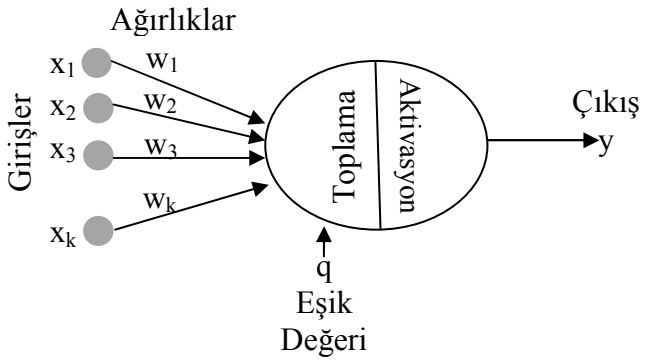
Yapay sinir ağları insan beyninin bir benzetimi olarak ortaya çıkmıştır. Yapay sinir ağlarının çalışma şekli insan beyninin çalışma şekline benzetilmeye çalışılmıştır.

### 3.3.1. Yapay Sinir Ağlarının Yapısı

İnsan beyninde olduğu gibi birbirine farklı şekillerde bağlı nöronlardan oluşmaktadırlar. Bu nöronlara yapay sinir ağı hücresi adı verilmektedir. Farklı şekillerde birbirine bağlanan yapay sinir ağı hücreleri ağları oluşturmaktadır.

İnsanlarda öğrenme yaşayarak ve tecrübe ederek olmaktadır. Bu süreç içinde beyin hücreleri arasındaki bağlantılar ayarlanır ya da yeni bağlantılar oluşur. Yapay sinir

ağlarında ise oluştuğu yapay sinir ağı hücreleri arasındaki bağlantı ağırlıkları güncellenerek öğrenme işlemi gerçekleştirilmektedir. Şekil 3.8'de yapay sinir ağı hücresi gösterilmektedir.

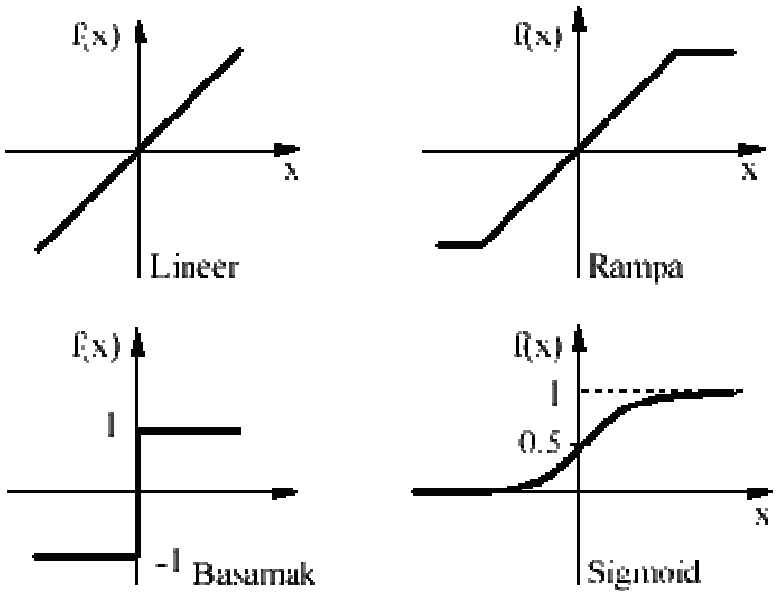



Şekil 3.8. Yapay sinir ağı hücresi.

Girişler çevreden aldığı bilgiyi yapay sinir ağı hücresine getirir. Girişler kendinden önceki yapay sinir ağı hücresinden veya dış dünyadan yapay sinir ağı hücresine gelebilir. Bir yapay sinir ağı hücresi genellikle bir çok girdiyi alır.

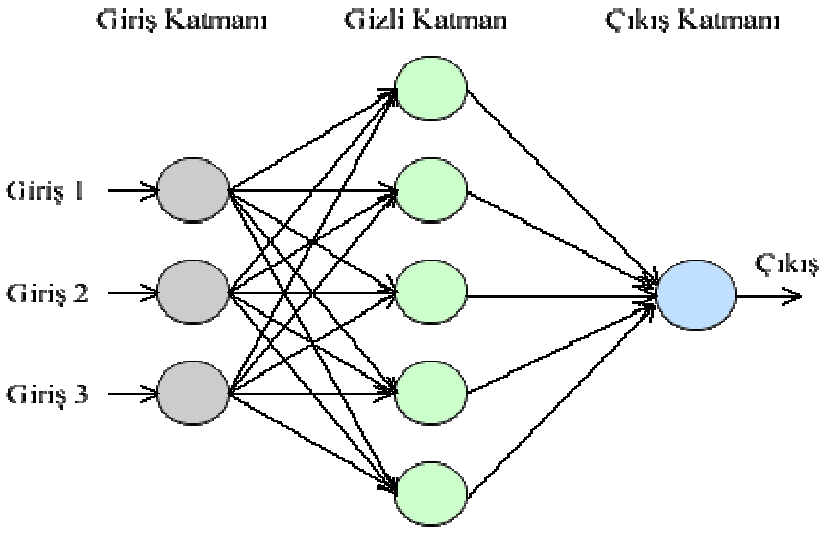
Ağırlıklar girişlerin yapay sinir ağı hücresi üzerindeki etkisini belirleyen uygun katsayılardır. Her bir giriş kendine ait bir ağırlığa sahiptir. Bir ağırlığın değerinin büyük olması, o girişin yapay sinir ağı hücresine güçlü bağlanması ya da önemli olması, küçük olması zayıf bağlanması ya da önemli olmaması anlamına gelmektedir.

Toplama fonksiyonu sinirde her bir ağırlığın ait olduğu girişlerle çarpımının toplamlarını eşik değeri ile toplayarak aktivasyon fonksiyonuna gönderir. Toplama fonksiyonunun sonucu aktivasyon fonksiyonundan geçirilip çıkışa iletilir. Şekil 3.9'da örnek aktivasyon fonksiyonları gösterilmektedir [31].




Şekil 3.9. Aktivasyon fonksiyonları.

Yapay sinir ağları çok sayıda yapay sinir ağı hücresinin bir araya gelmesi ile oluşur. Şekil 3.10'da üç girişli tek çıkışlı çok katmanlı bir yapay sinir ağı gösterilmektedir.




Şekil 3.10. Yapay sinir ağı.

### 3.3.2. Yapay Sinir Ağları Türleri

Yapay sinir ağları, ağ hesaplamaları, ağ yapıları ve ağ öğrenme dinamikleri açısından öğreticili ve öğreticisiz yapay sinir ağları olmak üzere iki sınıfa ayrılabilirler.

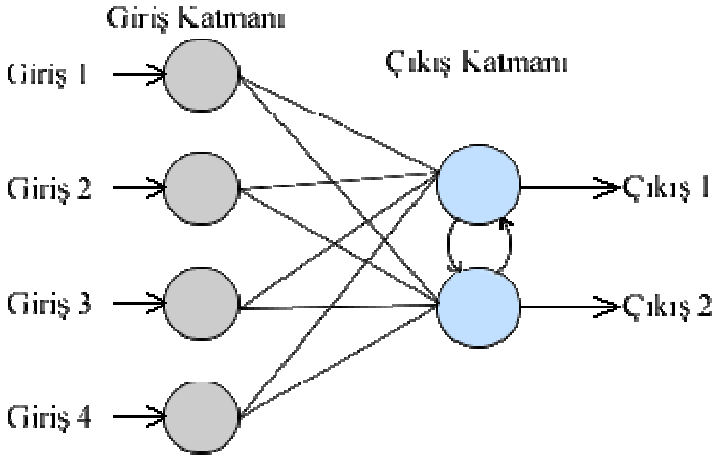
Öğreticisiz yapay sinir ağları öğreticisiz öğrenme kurallarını kullanan sinir ağlarıdır. Bunlar arasında en yaygın olarak kullanılanlar, Adaptif rezonans teorisi, Hopfield ağları, Kohonen ağı ve Sayaçlı yayılma ağıdır. İkili değerleri kullanan Adaptif rezonans teorisini Carpenter ve Grossberg (1986) geliştirilmiştir. Bu ağ giriş katmanından çıkış katmanına iki yönlü bağlantılar ile bağlıdır. Ayrıca çıkış katmanındaki nöronlar kendi içlerinde geri besleme bağlantılarına sahiptirler. Yarışmacı öğrenme kuralını kullanır. Basit bir eşik fonksiyonuna sahiptir. Bilgiler hem kısa dönemli hemde uzun dönemli hafızada saklanmaktadır. Kesikli Hopfield ağı, Hopfield (1987) tarafından geliştirilmiştir. Bu ağ tek katmanlıdır, ve simetrik, yönsüz, katmaniçi ve geri besleme bağlantılarına sahiptir. Hebian öğrenme kuralı ve adım fonksiyonu kullanır. Eşik fonksiyonu işlemci fonksiyonudur. Sürekli Hopfield ağı, Hopfield ve Tank (1984) tarafından geliştirilmiştir. Bu ağ gezgin satıcı problemi gibi en iyiyi bulma problemlerinde yaygın olarak kullanılmaktadır. İşlemci fonksiyonu sigmoid ve tanjant hiperbolik fonksiyonudur. Kohonen ağı, Kohonen (1982) tarafından geliştirilmiştir. Giriş ve çıkış nöronlarından oluşmaktadır. Nöronlar katmaniçi bağlantılar ile birbirine bağlıdır. Yarışmacı öğrenme kuralı ve bu kurala ait aktivasyon fonksiyonunu kullanır. Ağırlıkların değiştirilmesinde ise komşuluk ilişkilerini dikkate alır. Sayaçlı yayılma ağı Hecht-Nielsen (1987) tarafından geliştirilmiştir. Kohonen ve Grossberg ağlarının bir birleşimidir. Hebian öğrenme kuralı ile Kohonen vektör boyutlandırıcı öğrenme kurallarının bir kombinasyonudur [32].

Öğreticili yapay sinir ağları arasında Perceptron, ADALINE (Adaptive Linear Neuron)/MADALINE (Multiple Adaline), Geriyayılım ve Boltzmann makinası en çok kullanılan ağlardır. Perceptron Rosenblatt (1985) tarafından geliştirilen iki katmanlı bir sinir ağıdır. Katmanlararası bağlantılar içerir. Hata düzeltme öğrenme kuralını ve adım fonksiyonunu kullanmaktadır. Bir sinir hücresinin birden fazla girdiyi alarak bir çıktı üretmesi prensibine dayanır. ADALINE Widrow ve Hoff

(1960) tarafından geliştirilmiştir. Delta kuralını ve delta fonksiyonu veya sigmodial fonksiyonu kullanır. Doğrusal dağılmış desenleri iki sınıf içinde başarı ile sınıflandırır. MADALINE Widrow tarafından geliştirilmiştir. Bir orta katman eklenmiş ADALINE ağıdır. Doğrusal olmayan desenlerin sınıflandırılması için kullanılır. Geriyayılım ağı temelde katmanlararası bağlantılar içeren üç katmanlı perceptron yapısındadır. Genelleştirilmiş delta kuralı kullanır. Lojistik veya sigmoid aktivasyon fonksiyonu içerir. Boltzmann makinası Hinton, Ackley ve Sejnowski (1984) tarafından geliştirilen üç katmanlı, simetrik ve katmanlararası yönsüz bağlantılar içerir. Lojistik aktivasyon fonksiyonu ve tavlama benzetiminin bileşimi olan bir aktivasyon fonksiyonunu kullanır. Öğrenme kuralı stokastikdir [32].

### 3.3.3. Yarışmacı Öğrenme Ağı

Yarışmacı öğrenme öğreticisiz bir başka ifade ile danışmansız öğrenme türlerinden biridir. Sisteme hedef değerler verilmemektedir. Genellikle sınıflandırma problemlerinde kullanılmaktadır. Bütün çıkış birimleri giriş birimleri ile bağlantılıdır. Çıkış birimleride birbirleriyle bağlantılıdır. Sadece tek bir çıkış birimi aktive olur ve ağırlıkları değişir. Kazanma kuralı amaca göre değişmektedir. Şekil 3.11'de dört giriş iki çıkışlı bir yarışmacı öğrenme ağının modeli gösterilmektedir.




Şekil 3.11. Yarışmacı öğrenme ağı.

Yarışmacı öğrenmede kazanan çıkışın ağırlığının güncellenmesi için Eşitlik 3.7 kullanılmaktadır.

$$w_{k+1} = w_k + n(x - w_k) \quad (3.7)$$

Eşitlik 3.7'de ki $w_{(k+1)}$ yeni ağırlık değerini, $w_{(k)}$ eski ağırlık değerini, x giriş değerini ve n ağırlık azaltma oranını göstermektedir. Ağırlık azaltma oranı arttıkça salınım da artmaktadır.

Yarışmacı öğrenme ağının işleyişi şu adımları içermektedir;

a) Giriş ve çıkış sayısına karar verilir.
b) Öğrenme hızı ve iterasyon sayısı belirlenir.
c) İlk giriş değeri için kazanan çıkış kazanma kuralına göre tespit edilir.
d) Sadece kazanan çıkışın ağırlıkları Eşitlik 3.7' de ki gibi güncellenir.
e) İterasyon sayısı kadar bütün giriş değerleri için c ve d adımları tekrarlanır.
f) Ağırlıkları güncellenen çıkışların ağırlıkları oluşan kümelerin merkezlerini belirttiği kabul edilir.

Öğrenme işlemi sırasında gözlemlenen önemli özellikler;

- Bir birimin etkinliği onun uzaklığı ile ters orantılıdır. Bir birimin bir örneğe daha yakın olması, onun etkinliğinin daha yüksek olması demektir.
- Bir birim öğrendiği zaman, o birim öğreniyor olduğu örneğe daha yakın bir noktaya hareket eder. Birimin gittiği mesafe öğrenme hızı ile belirlenir. Yüksek bir öğrenme hızı, birimin örneğe doğru daha çabuk hareket edeceği anlamına gelir.
- Yarışmacı öğrenim tanımına göre, bir örneğe en yakın olan birim yalnızca o örneği öğrenecektir.
- Örneklerin arasındaki mesafe onların benzerliğini gösterir. Çok yakın iki örnek oldukça benzerdir, bunun yanında, iki uzak örnek benzer değildir [31].

Yarışmacı öğrenim daha büyük bu sistemin parçasını oluşturmakta kullanılabilmektedir. Yarışmacı öğrenimi kullanan ağlar, homojen ağlardan daha hızlı problemi çözdükleri için geri yayılım algoritması gibi başka algoritmalara ek

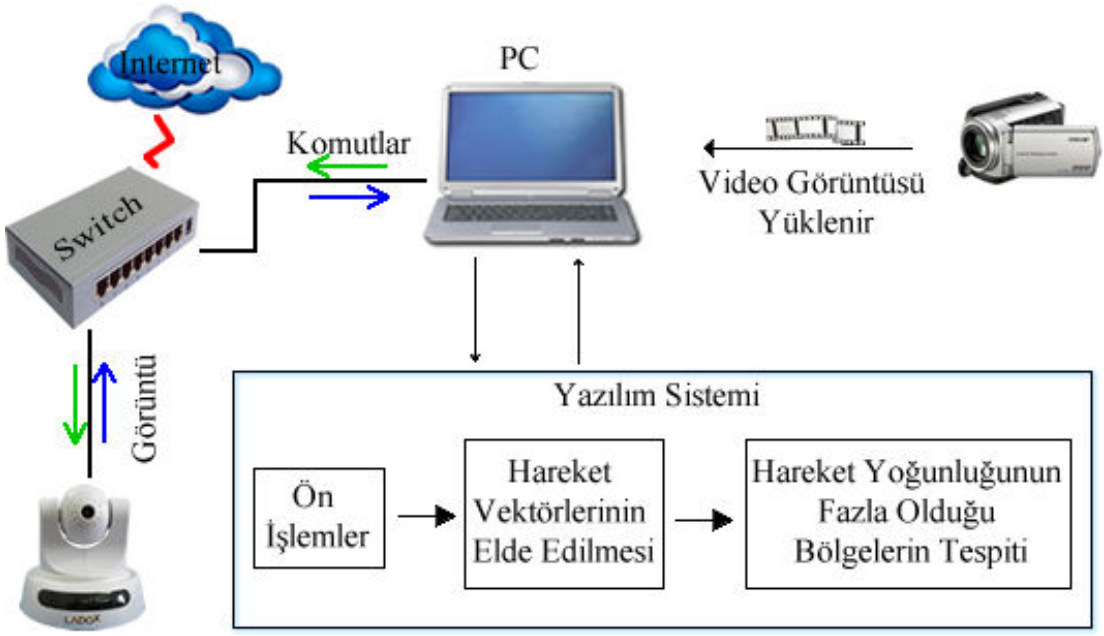
olarak da kullanılabilmektedir. Ayrıca yarışmacı öğrenim, bilgi sıkıştırmada ve sıkıştırılan bilgilere ulaşmada da kullanılabilmektedir [31].

# BÖLÜM 4

# HAREKET YOĞUNLUĞUNUN VE YÖNÜNÜN TESBİTİ İÇİN GERÇEKLEŞTİRİLEN YAZILIM SİSTEMİ

Gerçekleştirilen yazılım sisteminde hem video görüntüleri hem de IP kameradan alınan görüntüler kullanılabilmektedir. Kullanıcı isterse hazır video görüntüsünü sisteme yükleyip, bu video görüntüsü üzerinde hareket analizi yapabilmektedir yada sisteme bağlı olarak çalışan ip kamerasını başlatıp, bu kameradan aldığı görüntüleri aynı anda hem izleyip hem de işleyebilmektedir. Ayrıca kullanıcı IP kamerayı isterse sistemden kendisi yönlendirebilmekte ya da yazılım kendisi hareket ettirebilmektedir.

Video görüntüsü sisteme yüklendikten ya da IP kameradan görüntü alınmaya başlandıktan sonra görüntüler bazı ön işlemlerden geçirilmektedir. Daha sonra yazılım blok eşleştirmeye dayalı bir hareket tahmini algoritmasını kullanarak hareket vektörlerini çıkarmaktadır. Sonrasında bu hareket vektörleri değerlendirilir ve farklı algoritmalar kullanılarak hareket yoğunluğunun fazla olduğu bölgeler tespit edilir. Görüntüler IP kameradan alınıyor ise yazılım IP kamerayı otomatik olarak hareket yoğunluğunun olduğu bölgeye yönlendirebilmektedir. Şekil 4.1'de yazılım sisteminin çalışma şekli gösterilmektedir.




Şekil 4.1. Yazılım sisteminin çalışma şekli.

## 4.1. HAREKET VEKTÖRLERİNİN ELDE EDİLMESİ

Hareket vektörlerinin elde edilmesi için blok eşleştirme algoritmalarından yararlanılmıştır. Bunlar tüm arama algoritması, üç adımlı arama algoritması, baklava biçimli arama algoritması ve hiyerarşik arama algoritmasıdır. Kullanıcı istediği algoritmayı yazılım sisteminden seçip kullanabilmektedir.

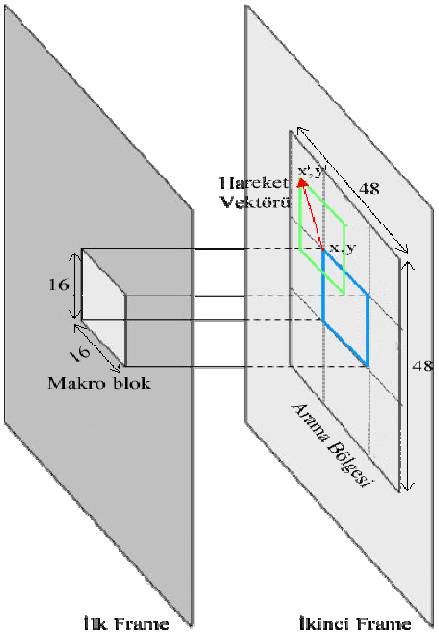
Video görüntüsünden ya da IP kameradan alınan görüntüler hareket vektörleri elde edilmeden önce bir takım ön işlemlerden geçmektedir. Bu ön işlemler sırasıyla;

- Gri tonlu hale çevirme
- Kenar bulma
- Eşikleme
- Gürültülerin temizlenmesi

Görüntü ilk önce gri tonlu hale çevrilmektedir. Ayrıca bu işlem sırasında görüntü yatayda ve dikeyde iki olarak, dört bölgeye ayrılmakta ve her bölgenin piksel değerlerinin ortalaması hesaplanmaktadır. Daha sonra sobel yatay ve dikey

operatörleri kullanılarak görüntüdeki kenar noktalar ortaya çıkarılmaktadır. Son olarak da her bir bölge için hesaplanan ortalama değer kullanılarak eşikleme işlemi yapılmaktadır. Eşikleme işleminden sonra görüntü ikili bir yapı almaktadır. Kenarların olduğu bölgeler beyaz, diğer bölgeler siyahtır. Aralarında hareket tahmini yapılacak olan iki görüntü bu işlemlerden geçtikten sonra ikili olarak karşılaştırılır. İlk görüntüde beyaz ikinci görüntüde siyah olan bölgeler beyaz, diğer yerler siyah olarak kabul edilir ve yeni bir ikili görüntü oluşturulur [16]. Son olarak bu ikili görüntüye filtreleme uygulanır böylece gürültüler temizlenmiş olur. Bu ikili görüntüde beyaz renkli piksellerin fazla olduğu bölgeler tespit edilir ve aralarında hareket tahmini yapılacak orijinal görüntülerde bu makro bloklar incelenir. Böylelikle görüntüdeki belli makro bloklar için algoritma çalıştırılır böylece işlem zamanı kısaltılmış olur.

Kullanılan görüntüler 320x240 piksel boyutlarındadır. Kullanıcının isteğine göre bu görüntüler 10x10, 11x11, 12x12, 14x14, 16x16, 18x18, 20x20, 22x22, 24x24, 26x26, 28x28 ve 30x30 piksel boyutlarında makro bloklara ayrılmaktadır. Blok eşleştirme algoritması olarak Tüm Arama algoritması seçilmiş ise ön işlemler sonucunda hareket olduğu düşünülen makro bloklar, çevresindeki 8 bloğu kapsayan alanda aranmaktadır. Bu bölge 3x3 makro blokluk bir alana karşı gelmektedir. Makro blok bu bölge içerisinde x ve y yönlerinde 1 piksellik adımlarla ilerletilerek bütün bölgenin içinde arama işlemi yapılmıştır. En uygun bloğu bulmak için Mutlak Fark Toplamı yöntemi kullanılmıştır. Makro blokdaki toplam piksellerin renk değerinden, karşılaştırılan bölgenin renk değerleri çıkarılmış ve tek tek toplanarak mutlak fark toplamı bulunmuştur. Mutlak fark toplamı en düşük çıkan bölge en uygun makro blok olarak kabul edilmektedir. En uygun bloğun yerine göre de hareket vektörü oluşturulmaktadır. Şekil 4.2'de tüm arama algoritması ile oluşturulan hareket vektörü gösterilmektedir. Mavi blok aranan makro bloğu, yeşil blok en uygun makro bloğu ve kırmızı ok da hareket vektörünü göstermektedir.



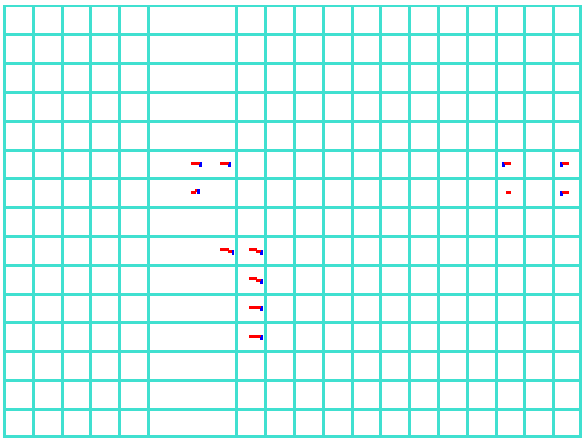

Şekil 4.2. Tüm arama algoritması ile oluşturulan hareket vektörü.

Bu tez çalışmasında kullanılan Tüm Arama algoritması şu şekildedir;

*Basla*
*Karşılaştırılacak frameleri NxN boyutlarında makro bloklara ayır*
*Arama bölgesini 3x3 makro blok olarak belirle*
*For ilk framedeki her bir $MB_1(x,y)$için*
*{*
*For her bir i ϵ { x-N, x-(N-1),..., x+N}*
*{*
*For her bir j ϵ { y-N, y-(N-1),..., y+N}*
*{*
*İkinci framede ki $MB_2(i,j)$ ile $MB_1(x,y)$ arasındaki MFT değerini hesapla*
*Eğer i=x-N ve j=y-N ise En küçük değer=MFT*
*Eğer MFT<En küçük değer ise { En küçük değer=MFT, $x'=i, y'=j$ }*
*}*
*}*
*(x,y) ve $(x', y')$ noktaları arasında hareket vektörünü oluştur*
*}*
*Son*

(a)

(b)

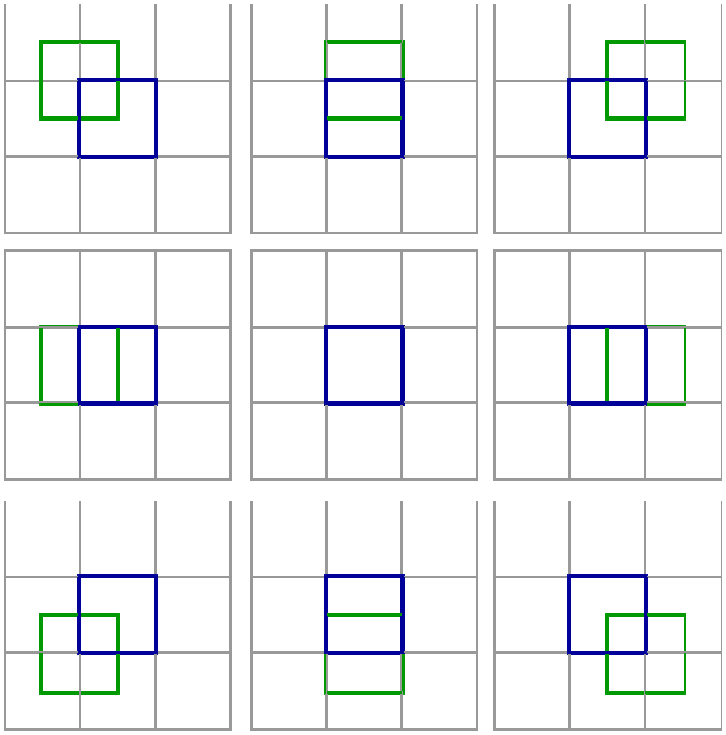


(c)

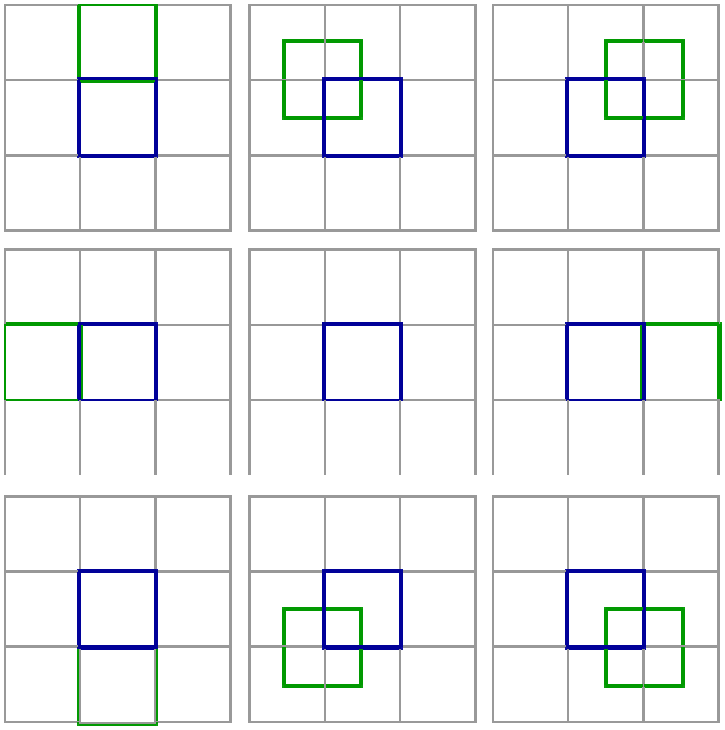
Şekil 4.3. İki görüntü arasında oluşan hareket vektörleri a) ilk görüntü, b) beş frame sonraki görüntü ve c) hareket vektörleri.

Şekil 4.3 a'da ilk görüntü, Şekil 4.3 b'de beş frame sonraki görüntü ve Şekil 4.3 c'de geliştirilen yazılımda Tüm Arama algoritması kullanılarak bu iki görüntü arasında oluşturulan hareket vektörleri gösterilmektedir. Görüntüler 16x16 piksel boyutlarında makro bloklara ayrılmıştır.

Blok eşleştirme algoritması olarak Üç Adımlı Arama ya da Baklava Biçimli Arama algoritması seçilmiş ise ön işlemler sonucunda hareket olduğu düşünülen makro bloklar, çevresindeki 9 makro blok da aranmaktadır. En uygun makro blok bulunduğunda bu bloğun çevresindeki 9 makro blok taranmaktadır. Bu işlem son bir kez daha yapılıp en uygun makro blok bulunmaktadır. En uygun bloğu bulmak için yine Mutlak Fark Toplamı yöntemi kullanılmıştır. En uygun bloğun yerine göre de hareket vektörü oluşturulmaktadır. Şekil 4.4'de Üç Adımlı Aramanın Şekil 4.5'de Baklava Biçimli Aramanın ilk arama adımları gösterilmiştir. Her bir çerçevenin bir kenarı seçilen blok boyutundadır. Mavi çerçeve aranan bloğu, yeşil çerçeve karşılaştırılan makro bloğu göstermektedir.



Şekil 4.4. Üç adımlı aramanın ilk arama adımları.



Şekil 4.5. Baklava biçimli aramanın ilk arama adımları.

Bu tez çalışmasında kullanılan Üç Adımlı Arama algoritması şu şekildedir;

*Basla*
*Karşılaştırılacak frameleri NxN boyutlarında makro bloklara ayır*
*For ilk framedeki her bir $MB_1(x,y)$için*
*{*
*tx=x ve ty=y*
*For t=0 dan t<3 olduğu sürece*
*{*
*For her bir i $\epsilon$ { tx-(N/2), tx, tx+(N/2)}*
*{*
*For her bir j $\epsilon$ { ty-(N/2), ty, ty+(N/2)}*
*{*
*İkinci framede ki $MB_2(i,j)$ ile $MB_1(tx,ty)$ arasındaki MFT değerini hesapla*
*Eğer i=tx-(N/2) ve j=ty-(N/2) ise En küçük değer=MFT*
*Eğer MFT<En küçük değer ise { En küçük değer=MFT, $x'=i, y'=j$ }*
*}*
*}*
*tx= $x'$ ve ty= $y'$*
*}*
*(x,y) ve ($x'$, $y'$) noktaları arasında hareket vektörünü oluştur*
*}*
*Son*

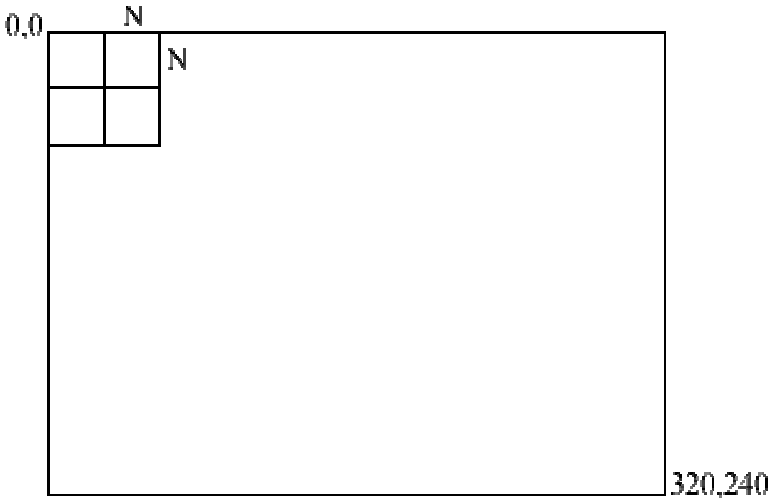
Bu tez çalışmasında kullanılan Baklava Biçimli Arama algoritması şu şekildedir;

*Basla*
*Karşılaştırılacak frameleri NxN boyutlarında makro bloklara ayır*
*For ilk framedeki her bir $MB_1(x,y)$için*
*{*
*tx=x ve ty=y*
*For t=0 dan t<3 olduğu sürece*
*{*
*Dizi[9,2]={(tx,ty-N),(tx-(N/2),ty-(N/2)),(tx+(N/2),ty-(N/2)),*
*(tx-N,ty),(tx,ty),(tx+N,ty),*
*(tx-(N/2),ty+(N/2)),(tx+(N/2),ty+(N/2)),(tx,ty+N)}*

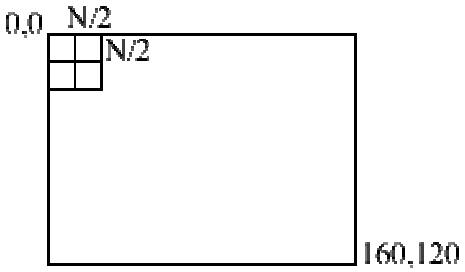
*For her bir i ϵ { 0,1,...,8}*
*{*
*İkinci framede ki $MB_2$(dizi[i,0],dizi[i,1]) ile $MB_1$(tx,ty) arasındaki MFT değerini hesapla*
*Eğer i=0 ise En küçük değer= MFT*
*Eğer MFT<En küçük değer ise*
*{ En küçük değer=MFT, $x'$=dizi[i,0], $y'$=dizi[i,1] }*
*}*
*tx= $x'$ ve ty= $y'$*
*}*
*($x'$, $y'$) noktaları arasında hareket vektörünü oluştur*
*}*
*Son*

Blok eşleştirme algoritması olarak Hiyerarşik Arama algoritması seçilmiş ise ilk olarak görüntüler belli işlemlerden geçirilerek dört de bir boyutuna düşürülür. Daha sonra küçültülmüş görüntüde ön işlemlerden sonra Tüm Arama algoritması uygulanır. Bulunan hareketli makro blok normal görüntüdeki koordinatlarına göre tekrar Tüm Arama algoritmasından geçirilir. En son oluşturulan hareket vektörü gerçek hareket vektörü olarak kabul edilir. Bu yöntemde de en uygun bloğu bulmak için Mutlak Fark Toplamı yöntemi kullanılmıştır. Şekil 4.6'da Hiyerarşik Arama

algoritmasında kullanılan orijinal boyutlarında ki görüntü, Şekil 4.7'de küçültülmüş görüntü gösterilmektedir. N blok boyutudur.




Şekil 4.6. Orijinal boyutlarındaki görüntü.

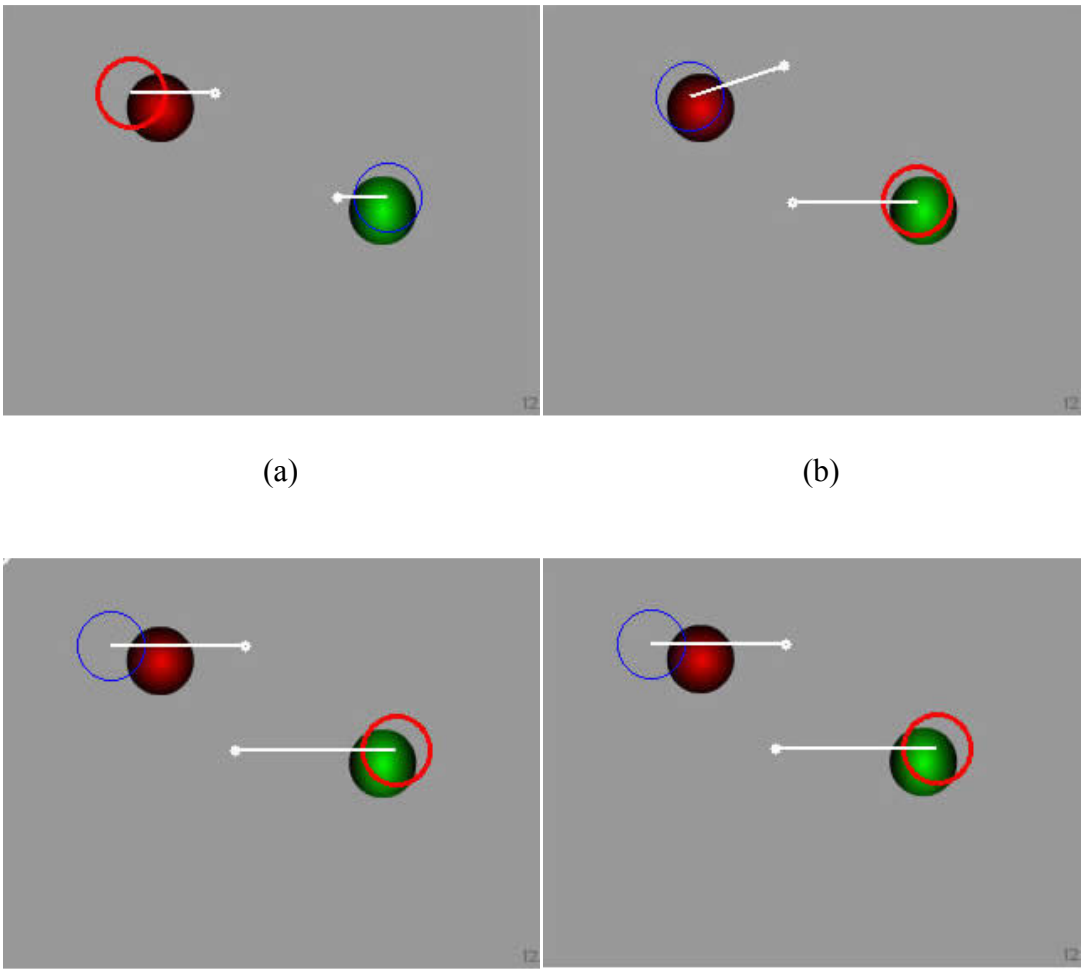



Şekil 4.7. Küçültülmüş görüntü.

Bu tez çalışmasında kullanılan Hiyerarşik Arama algoritması şu şekildedir;

*Basla*
*Karşılaştırılacak framelerden 1. seviye alt örnekler oluştur.*
*Alt örnekleri N/2xN/2 boyutlarında makro bloklara ayır*
*Arama bölgesini 3x3 makro blok olarak belirle*
*For ilk framemin alt örneğindeki her bir $AMB_1(i,j)$için*
*{*
*İkinci framemin alt örneğinde Tüm arama algoritması kullanılarak en uygun makro blok $AMB_2$ $(i',j')$ tespit edilir.*
*$AMB_2(i', j')$ nin konumuna gore ilk frame de Tüm arama algoritması kullanılarak $MB_1(x,y)$ ye en uygun blok $MB_2(x', y')$ tespit edilir.*
*(x,y) ve $(x', y')$ noktaları arasında hareket vektörünü oluştur*
*}*
*Son*

Kullanılan blok eşleştirme algoritmalarının tespit ettiği hareket yönünü incelemek için bir test video görüntüsü oluşturulmuştur. Bu görüntüde kırmızı top sağa, yeşil top sola doğru düz bir şekilde hareket etmektedir. Şekil 4.8 a'da tüm arama, Şekil 4.8 b'de hiyerarşik arama, Şekil 4.8 c'de üç adımlı arama ve Şekil 4.8 d'de baklava biçimli arama algoritmaları kullanılarak elde edilen toplam hareket vektörleri beyaz bir çizgi halinde gösterilmektedir. Çizginin kalın kısmı hereket vektörünün yönünü göstermektedir. Kırmızı ve mavi daireler yazılımın tespit ettiği hareket yoğunluğunun fazla olduğu bölgeleri göstermektedir. Kırmızı daireler mavi dairelere gore hareketin daha fazla olduğu bölgeyi göstermektedir. Bu yüzden toplam hareket vektörleri bu bölgelerde daha büyük çıkmıştır.



(a) (b)

(c) (d)

Şekil 4. 8. Toplam hareket vektörünün a) Tüm arama algoritması ile, b) hiyerarşik arama algoritması ile, c) üç adımlı arama algoritması ile ve d) baklava biçimli arama algoritması ile tespiti.

## 4.2. HAREKET YOĞUNLUĞUNU BELİRLEME

Geliştirilen yazılım sisteminde hareket yoğunluğunun fazla olduğu yerleri tespit etmek için kümeleme yöntemlerine ve yapay sinir ağlarına dayanan beş farklı algoritma kullanılmıştır. Bu algoritmalar hem video görüntülerine hem de IP kameradan elde edilen görüntülere uygulanabilmektedirler Kullanıcı istediği algoritmayı seçebilmektedir.

### 4.2.1. K-Means Algoritması ile Hareket Yoğunluğunu Belirleme

Hareket yoğunluğunu belirlemek için k-means algoritmasına dayalı iki farklı yöntem kullanılmıştır. Birinci yöntemde kullanıcı hareketin yoğun olduğu bölgeleri isteğine göre 1, 2 yada 3 kümeye ayırabilmektedir. Bu yöntem daha çok bir ya da birkaç kişinin hareket halinde olduğu görüntülerde daha iyi sonuç vermektedir.

Öklid uzaklığı temel alınarak kümeleme işlemi yapılmıştır. Kullanılan formül Eşitlik 4.1'de gösterilmektedir. MB makro bloğu, C küme merkezini, (x,y) küme merkezlerinin koordinatlarını, $d_{jk}$ j. makro bloğun k. küme merkezine olan uzaklığını göstermektedir. Ayrıca j makro blok sayısı kadar değer alırken k, {1,2,3} değerlerinden birini almaktadır.

$$d_{jk} = \sqrt{(MB_{jx} - C_{kx})^2 + (MB_{jy} - C_{ky})^2} \tag{4.1}$$

Çalışmada kullanılan bu yöntemin algoritması şu adımları içermektedir;

*Başla*
*Oluşturulacak küme sayısı k değerini belirle*
*k değeri kadar rastgele küme merkezi belirle ($C_1$,$C_2$, …, $C_k$)*
*Do*
*{*
*Mevcut merkezleri sakla*
*($tempC_1 = C_1$, $tempC_2 = C_2$, … $tempC_k = C_k$ )*
*Her bir hareketli makro blok ile küme merkezleri arasındaki Öklid uzaklığını hesapla*
*Hareketli makro blok hangi merkeze yakın ise o kümeye dahil et*
*Her bir kümedeki makro blokların orta merkezini bul($C_1$,$C_2$, …, $C_k$)*
*}*
*While ($tempC_1 \neq C_1$, $tempC_2 \neq C_2$, … $tempC_k \neq C_k$ )*
*Son*

Şekil 4.9'da birinci yönteme göre hareket yoğunluğunun fazla olduğu bölgelerin belirlendiği bir görüntü gösterilmektedir. Kullanıcı grup sayısını 3 olarak belirlemiştir. Hareket tahmini algoritması olarak da tüm arama algoritması kullanılmıştır. Kırmızı daire hareket yoğunluğunun en fazla olduğu bölgenin merkezini, mavi daireler diğer hareketli bölge merkezlerini göstermektedir.

Şekil 4.9. K-means algoritması birinci yöntem ile hareket yoğunluğunu belirleme.

İkinci yöntem de oluşturulan hareket vektörleri k-means algoritması ile kümelendikten sonra kümelerin merkez noktaları çevresindeki 5x5 blokluk alandaki 25 adet makro blok incelenmektedir. Bu 25 adet makro bloğun en az 6 tanesinde hareket vektörü oluşmuşsa bu merkezin çevresinde bir topluluk hareketi olduğu kabul edilmiştir. Kullanıcı hareketin yoğun olduğu bölgeleri isteğine göre 1, 2 yada 3 kümeye ayırabilmektedir. Bu yöntem bir insan topluluğunun birlikte hareket ettiği görüntülerde daha iyi sonuç vermektedir. Bu yöntemin algoritması şu şekildedir;

*Basla*

*K-means yöntemine göre k sayıda ki C merkez noktaları hesapla*

*For her bir $C_k$ (x,y) merkez noktası için*

*{*

*$T_k$=0 olsun*

*For her bir i ϵ { x-2N, x-N,..., x+2N}*

*{*

*For her bir j ϵ { y-2N, y-N,..., y+2N}*

*{*

*MB(i,j) de hareket varsa $T_k$ yı 1 artır*

*}*

*}*

Eğer $T_k$ >6 ise $C_k$ merkezinde toplu bir hareket olduğunu kabul et

*}*

*Son*

Şekil 4.10'da bu yönteme göre hareket yoğunluğunun fazla olduğu bölgelerin tespit edildiği bir görüntü gösterilmektedir. Kullanıcı grup sayısını 2 olarak belirlemiştir. Hareket tahmini algoritması olarak da tüm arama algoritması kullanılmıştır. Kırmızı daire, hareket yoğunluğunun mavi dairenin olduğu bölgeye göre daha fazla olduğu bölgedir.

Şekil 4.10. K-means algoritması ikinci yöntem ile hareket yoğunluğunu belirleme.

### 4.2.2. Görüntü Bölütleme Algoritması ile Hareket Yoğunluğunu Belirleme

Bu yöntem de video görüntüsünden ya da IP kameradan alınan görüntü öncelikle 64x48 piksel olarak 25 bölgeye ayrılmaktadır. Daha sonra her bir bölgenin içindeki makro bloklarda bulunan hareket vektörlerinin sayısı belirlenmektedir. En fazla hareket vektörü hangi bölgede bulunmaktaysa o bölge hareketin en yoğun olduğu bölge olarak kabul edilmektedir. Bu yöntemin algoritması şu şekildedir;

*Basla*
*Görüntüyü t=25 tane M bölgesine ayır*
*For her bir $M_t$ bölgesi için*
*{*
*İçindeki hareketli makro blokların sayısı $T_t$ 'yi hesapla*
*Eğer t=1 ise*
*{*
*Maksimum=$T_t$*
*$M_t$ hareketin en yoğun olduğu bölgedir*
*}*
*Eğer $T_t$>Maksimum ise*
*{*
*Maksimum=$T_t$*
*$M_t$ hareketin en yoğun olduğu bölgedir*
*}*
*}*
*Son*

Şekil 4.11'de görüntü bölütleme algoritması ile hareket yoğunluğunun fazla olduğu bölgelerin tespit edildiği bir görüntü gösterilmektedir. Hareket tahmini algoritması olarak tüm arama algoritması kullanılmıştır. Kırmızı daire hareket yoğunluğunun en fazla olduğu bölgeyi göstermektedir.

Şekil 4.11. Görüntü bölütleme algoritması ile hareket yoğunluğunu belirleme.

### 4.2.3. En Yakın Komşu Algoritması İle Hareket Yoğunluğunu Belirleme

Bu yöntem oluşturulan hareket vektörlerinin en yakın komşu algoritması ile kümelenmesinden oluşmaktadır. Yöntemde hareketin yoğun olduğu bölgeler 2 kümeye ayrılmaktadır. Öklid uzaklığı temel alınarak kümelendirme yapılmıştır. Kümeler arasındaki uzaklığın bulunmasında iki kümenin bir birine en yakın değerleri dikkate alınmıştır. Bu yöntemin algoritması şu şekildedir;

*Başla*
*Her bir hareketli makro bloğu bir küme olarak kabul et*
*Toplam küme sayısı t yi hesapla*
*Do*
*{*
    *Her bir kümenin diğer kümelere olan Öklid uzaklığını hesapla*
    *En yakın olan kümeleri birleştir*
    *Küme sayısını t=t-1 yap*
*}*
*While (t > 2)*
*Son*

Şekil 4.12'de en yakın komşu algoritması ile hareket yoğunluğunun fazla olduğu bölgelerin tespit edildiği bir görüntü gösterilmektedir. Hareket tahmini algoritması olarak tüm arama algoritması kullanılmıştır. Kırmızı daire hareket yoğunluğunun en fazla olduğu bölgenin merkezini, mavi daire diğer hareketli bölge merkezini göstermektedir.

Şekil 4.12. En yakın komşu algoritması ile hareket yoğunluğunu belirleme.

### 4.2.4. Yarışmacı Öğrenme Ağı İle Hareket Yoğunluğunu Belirleme

Bu yöntem de video görüntüsünden ya da IP kameradan alınan görüntüdeki hareket vektörleri tespit edildikten sonra bu hareket vektörlerinin sayısı kadar girişe sahip bir yarışmacı öğrenme ağı oluşturulmaktadır. Ağın çıkış sayısı üç olarak ayarlanmıştır. Her bir hareket vektörünün çıkışlara olan öklid uzaklığı hesaplanmakta ve en küçük öklid uzaklığına sahip çıkış kazanmaktadır. Bu çıkışın ağırlığı güncellenirken diğerlerinin ki değişmemektedir

Çıkış sayısı üç olarak belirlenmesine rağmen yarışmacı öğrenme ağı hareket vektörlerini en uygun sayıda gruba ayırmaktadır. Hareket analizi yapılan bir video görüntüsünde ya da IP kameradan alınan görüntüde hareket vektörlerinin konumlarına göre 1, 2 ya da 3 grup oluşturabilmektedir. Yöntemin algoritması şu şekildedir;

*Başla*

*Toplam hareket vektörlerinin sayısı t yi hesapla*

*Ağın giriş sayısını t, çıkış sayısını 3 olarak kabul et*

*İterasyon sayısını i_sayısı belirle*

*Do*

*{*

*For her bir $HV_t$ hareket vektörü için*

*{*

*$HV_t$'nin çıkışlara olan Öklid uzaklığını hesapla*

*$HV_t$'ye en yakın olan çıkışın ağırlıklarını güncelle*

*}*

*}*

*While (i_sayısı>0)*

*Ağırlığı güncellenen çıkışların ağırlık değerlerini kümelerin merkez noktası olarak kabul et*

*Son*

Şekil 4.13'de ağırlıkları azaltma oranı 0.5, iterasyon sayısı 30 olarak kabul edilmiş yarışmacı öğrenme ağı kullanarak hareket yoğunluğu fazla olduğu bölgeler tespit edilen bir görüntü gösterilmektedir. Hareket tahmini algoritması olarak tüm arama algoritması kullanılmıştır. Kırmızı daire hareket yoğunluğunun en fazla olduğu bölgenin merkezini, mavi daireler diğer hareketli bölge merkezlerini göstermektedir.

Şekil 4.13. Yarışmacı öğrenme ağı ile hareket yoğunluğunu belirleme.

## 4.3. IP KAMERADAN GÖRÜNTÜNÜN ALINMASI VE IP KAMERANIN YÖNLENDİRİLMESİ

IP kameralar ağ üzerinden canlı görüntü alabileceğimiz kameralardır. Ayrıca alınan görüntüler kayıt edilebilmektedir. Özellikle güvenliğin önemli olduğu yerleri izlemek için yaygın olarak kullanılmaktadır.

Gerçekleştirilen yazılım sayesinde kullanıcı yerel ağa bağladığı bir IP kameradan görüntü alabilmektedir. Yapılan deneysel çalışmalarda Ladox Pan Tilt LD-3710-P/PW IP kamerası kullanılmıştır. Şekil 4.14'de kullanılan IP kamera gösterilmektedir. Özellikleri Çizelge 4.1'de gösterilmektedir.


Şekil 4.14. IP kamera.

Çizelge 4.1. Kullanılan IP kameranın özellikleri.

| Özellikler |
|---|
| 802.11g kablosuz fonksiyon |
| 1/4" renk cmos |
| 640x480 piksel |
| f:6.0 mm, F: 1.8 |
| Jpeg format |
| Pan: $-170^{o}$~ $+170^{o}$ |
| Tilt: $+45^{o}$ ~ $-90^{o}$ |
| Saniyede 24 frame |

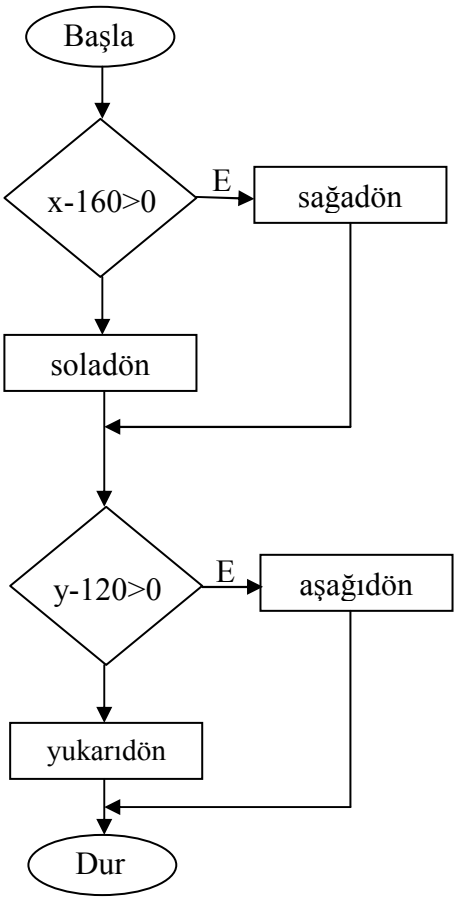
IP kameradan görüntü alma işlemi için öncelikle yerel ağa bağlanan IP kameranın IP numarasını bilmek gerekmektedir. Daha sonra yazılımla bu IP adresine bir "istek" gönderilmektedir. Buna karşılık olarak da IP kamera yazılıma bir "yanıt" göndermektedir. Bu yöntemle IP kameradan görüntü alınabilmektedir.

Hareket yoğunluğunun fazla olduğu bölgeye doğru IP kameranın yönlendirilmesi IP kameranın kendi programın da ki javascript kodları kullanılarak yapılmıştır. Bu javascript kodları düzenlenmiş ve fonksiyonlar halinde bir html dosyasına kaydedilmiştir.

Html dosyası yazılıma yerleştirilen bir web browser aracı ile ilişkilendirilmiştir. Bu html dosyasından fonksiyonlar yazılımda tek tek çağrılarak gerektiği yerlerde kullanılmıştır.

Hareket yoğunluğunun fazla olduğu bölgenin merkez noktası, görüntünün merkez noktası ile karşılaştırılarak IP kameranın ne tarafa dönmesi gerektiği kararlaştırılmaktadır. Görüntüde birden fazla hareketli nokta tespit edilmiş ise hareket yoğunluğunun en fazla olduğu nokta ile merkez nokta karşılaştırılmıştır. Hareket yoğunluğunun fazlalığı bu noktaların çevresindeki hareketli blokları sayarak belirlenmiştir. Yazılım sistemi bu işlemi otomatik olarak kendisi gerçekleştirmektedir. Gerektiği şekilde IP kamerayı sağa, sola, yukarı ya da aşağıya yönlendirebilmektedir.

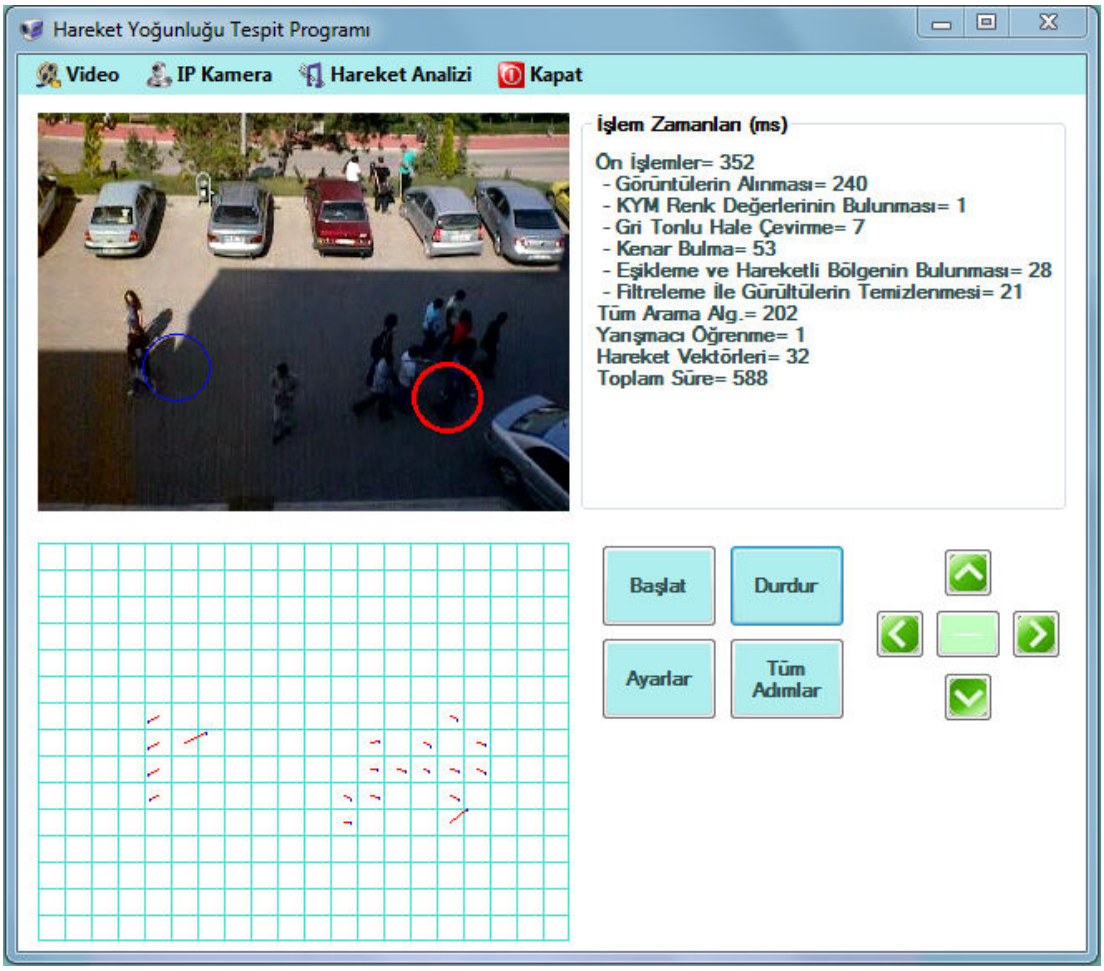
Ip kameranın hareketini gerçekleştiren program parçasının akış şeması Şekil 4.15'de gösterilmektedir. Geliştirilen yazılımda kullanılan görüntüler 320x240 piksel boyutlarında olduğu için merkez noktası (160,120) olarak kabul edilmiştir. Akış şemasında da görüldüğü gibi IP kamera hareket yoğunluğunun en fazla olduğu noktaya göre sola yada sağa ve yukarı yada aşağıya döndürülmektedir. Akış şemasındaki x ve y değerleri hareket yoğunluğunun en fazla olduğu noktanın koordinatlarını belirtmektedir. "Sağadön", "soladön", "aşağıdön" ve "yukarıdön" kameranın hareketini sağlayan fonksiyonlardır.




Şekil 4.15. IP kameranın hareketini gerçekleştiren program parçasının akış şeması.

## 4.4. YAZILIMIN ARAYÜZÜ

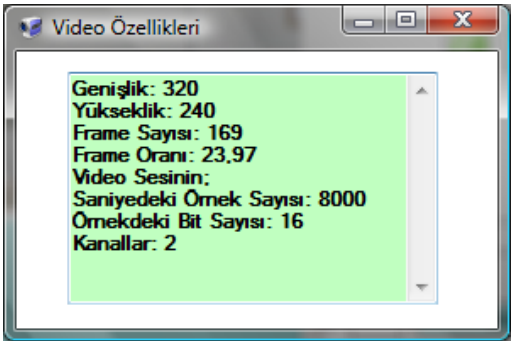
Yazılım Microsoft Visual Studio kullanılarak C# programlama dilinde Intel(R) Pentium(R) Dual T3400 2.16 GHz işlemcili, 3GB Ram belleğe sahip, Windows 7 işletim sistemi kurulu bir diz üstü bilgisayarda geliştirilmiştir. Yazılımın ana penceresi Şekil 4.16'da gösterilmektedir.




Şekil 4.16. Yazılımın ana penceresi.

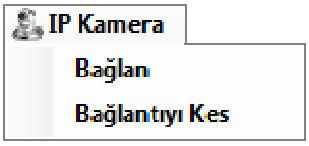
Ana pencerenin en üst kısmında ana menü yer almaktadır. Ana menü dört seçenekten oluşmaktadır. Bunlar Video, IP Kamera, Hareket Analizi ve Kapat menüleridir. Video menüsü Şekil 4.17'de gösterildiği gibi Aç, Oynat, Durdur ve Özellikler seçeneklerinden oluşmaktadır. Aç seçeneği kullanılarak bir video görüntüsü açılmaktadır. Oynat seçeneği video görüntüsünü oynatır. Durdur seçeneği video görüntüsünün oynatılmasını durdurur. Özellikler seçeneğinden açılan video görüntüsünün özellikleri görülmektedir. Şekil 4.18'de özellikler penceresi gösterilmektedir.


Şekil 4.17. Video menüsü.




Şekil 4.18. Özellikler penceresi.

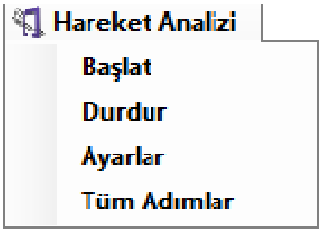
IP Kamera menüsü Şekil 4.19'da gösterildiği gibi Bağlan ve Bağlantıyı Kes seçeneklerinden oluşmaktadır. Bağlan seçeneği kullanılarak IP kameradan görüntü alınmaya başlanmaktadır. Bağlantıyı Kes seçeneği IP kameradan görüntü alınmasını durdurur.



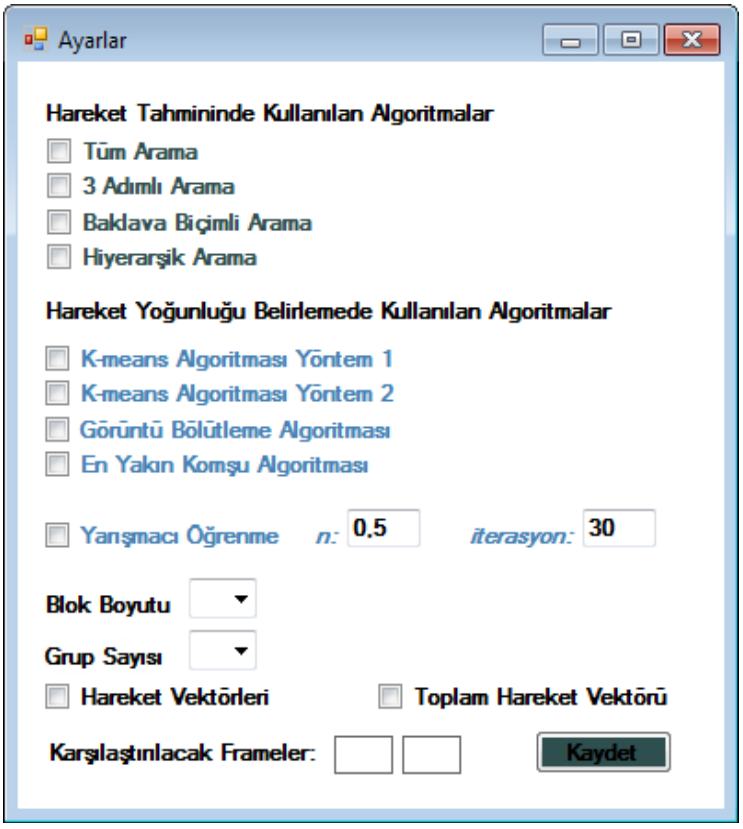

Şekil 4.19. IP kamera menüsü.

Hareket Analizi menüsü Şekil 4.20'de gösterildiği gibi Başlat, Durdur, Ayarlar ve Tüm Adımlar seçeneklerinden oluşmaktadır. Başlat seçeneği kullanılarak hareket analizi başlatılmaktadır. Durdur seçeneği kullanılarak hareket analizi durdurulmaktadır. Ayarlar seçeneğinden hareket tahmininde kullanılacak olan algoritma, hareket yoğunluğu belirlemede kullanılacak olan algoritma, yarışmacı öğrenme için ağırlık azaltma oranı ve iterasyon sayısı, blok boyutu, grup sayısı, hareket vektörlerinin ve toplam hareket vektörünün ekranda gösterilip

gösterilmeyeceği ve karşılaştırılacak frameler seçilmektedir. Şekil 4.21' de Ayarlar penceresi gösterilmektedir.

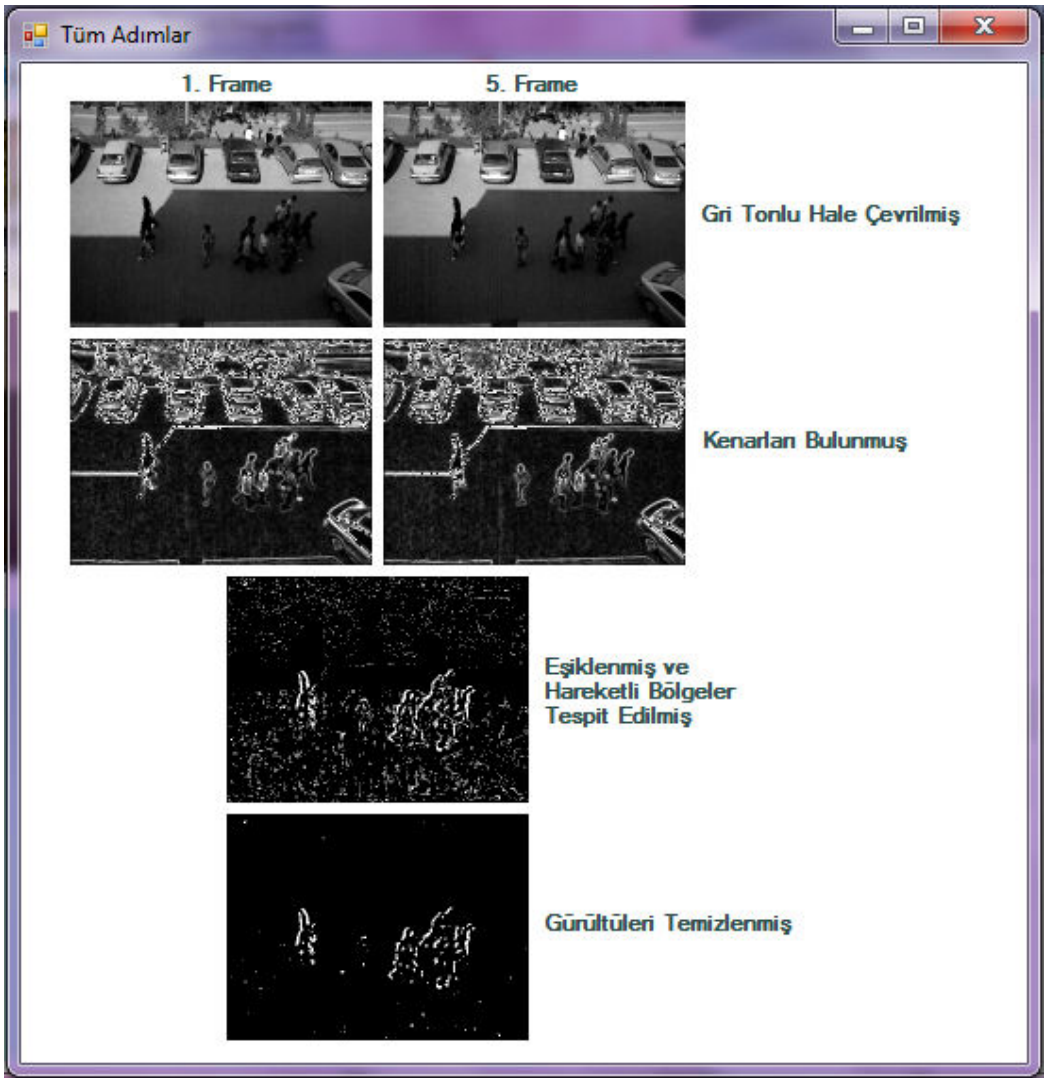



Şekil 4.20. Hareket analizi menüsü.




Şekil 4.21. Ayarlar penceresi.

Tüm Adımlar seçeneğinde karşılaştırılmak için seçilen framelerin geçirildiği ön işlemler gösterilmektedir. Bu ön işlemler gri tonlu hale çevirme, kenarların bulunması, eşikleme ve hareketli bölgelerin tespit edilmesi ve gürültülerin temizlenmesidir. Tüm Adımlar penceresi Şekil 4.22'de gösterilmektedir.




Şekil 4.22. Tüm adımlar penceresi.

Son olarak bulunan menü Kapat menüsüdür. Kapat menüsü kullanılarak yazılım ekranı kapatılmaktadır.

Ana pencerenin sol tarafında iki adet görüntü gösterme nesnesi bulunmaktadır. Bunlardan üsttekinde video görüntüsü ve IP kameradan alınan görüntüler gösterilmektedir. Alttakinde ise video görüntülerinde oluşan hareket vektörleri gösterilmektedir.

Ana pencerenin sağ üst köşesinde işlem zamanlarını gösteren bir nesne bulunmaktadır. Uygulanan işlemlerin ne kadar sürdüğünü milisaniye cinsinden göstermektedir.

Üstünde okların bulunduğu butonlar kullanıcının IP kamerayı hareket ettirmesini sağlamak içindir. IP kamerayı, üst de ki buton yukarıya, alt da ki buton aşağıya, sağ da ki buton sağa ve sol da ki buton sola çevirmektedir. Merkez de ki buton IP kamerayı başlangıç pozisyonuna getirmektedir.

Ekrandaki diğer butonlar hareket analizi menüsü ile aynı işlemleri yapmaktadırlar.

# BÖLÜM 5

# SONUÇLAR

Bu çalışmada video görüntülerinde hareket yoğunluğu ve yönü tespit edilmiştir. Hareketli bölgeyi bulmak için blok eşleştirmeye dayanan algoritmalardan tüm arama algoritması, üç adımlı arama algoritması, baklava biçimli arama algoritması ve hiyerarşik arama algoritması kullanılmıştır. Hareketin yoğun olduğu bölgeyi belirlemek için k-means kümeleme yöntemi, görüntü bölütleme algoritması, en yakın komşu algoritması ve yarışmacı öğrenme ağı kullanılmıştır.

C# görsel programlama dili kullanılarak hareket yoğunluğu ve yönü tespit eden bir yazılım geliştirilmiş ve bu yazılım sayesinde kullanılan algoritmalar hem zaman hem de performans bakımından değerlendirilmiştir. Deneysel çalışmalarda kullanılan görüntüler 320x240 piksel boyutlarındadır. Kullanılan makro blok boyutları 10x10, 16x16 ve 20x20 piksel olarak ayarlanmıştır. Hareketli makro blok sayısı 10 olarak sınırlandırılmıştır.

Çizelge 5.1'de yapılan deneysel çalışmalar sonucunda ortaya çıkan ön işlemlerin ve blok eşleştirme algoritmalarının ortalama işlem zamanları milisaniye cinsinden verilmektedir. Ortalama işlem zamanları 5 farklı video görüntüsünde yapılan 50 ölçümün ortalama değeri alınarak elde edilmiştir.

Ön işlem algoritmaları piksel bazında yapıldığından makro blok boyutunun değişmesi işlem süresini etkilememektedir. Blok eşleştirme algoritmalarında makro blok boyutu büyüdükçe arama alanı da büyüdüğünden işlem zamanı artmaktadır. Tüm arama algoritması en uzun işlem zamanına sahip olmasına rağmen bütün olasılıkları incelediği için en iyi sonucu vermektedir.

Çizelge 5.1. Ön işlemler ve blok eşleştirme algoritmalarının ortalama işlem zamanları

| İşlem | | Süre (ms) | | |
|---|---|---|---|---|
| | | 10x10 | 16x16 | 20x20 |
| Ön işlemler | Görüntülerin alınması | 240 | 240 | 240 |
| | KYM değerlerinin bulunması | 1,75 | 1,75 | 1,75 |
| | Gri tonlu hale çevirme | 7,36 | 7,36 | 7,36 |
| | Alt örnekleme | 0,12 | 0,12 | 0,12 |
| | Kenar bulma | 55,66 | 55,66 | 55,66 |
| | Eşikleme ve hareketli bölgenin bulunması | 19,3 | 19,3 | 19,3 |
| | Filtreleme ile gürültülerin temizlenmesi | 21,66 | 21,66 | 21,66 |
| Blok eşleştirme algoritmaları | Tüm arama | 16,88 | 101,83 | 238,74 |
| | Üç adımlı arama | 8,33 | 10,29 | 13,92 |
| | Baklava biçimli arama | 3,82 | 5,77 | 12,47 |
| | Hiyerarşik arama | 14,13 | 93,96 | 236.96 |

Çizelge 5.2'de yapılan deneysel çalışmalar sonucunda ortaya çıkan hareket yoğunluğu belirleme algoritmalarının ortalama işlem zamanları ve ortalama mutlak hata miktarları verilmektedir. İşlem zamanları mikrosaniye cinsinden hesaplanmıştır. Ortalama işlem zamanları 5 farklı video görüntüsünde yapılan 50 ölçümün ortalama değeri alınarak elde edilmiştir. Yarışmacı öğrenme ağının iterasyon sayısı 30, ağırlık azaltma oranı 0,5 olarak alınmıştır. Bütün algoritmaların ortalama işlem zamanları ve ortalama mutlak hata miktarları tüm arama algoritması ile oluşturulan hareket vektörleri kullanılarak hesaplanmıştır.

Ortalama mutlak hata hareket yoğunluğu belirleme algoritmalarının bulduğu hareket merkezi ile gerçek hareket merkezi arasındaki öklid uzaklık farkı hesaplanarak bulunmuştur. Her bir algoritma için 20 farklı ölçüm yapılmış ve bunların ortalaması alınmıştır. Yapılan ölçümler piksel değerindedir.

Çizelge 5.2. Hareket yoğunluğu belirleme algoritmalarının ortalama işlem zamanları ve ortalama mutlak hataları

| Hareket yoğunluğu belirleme algoritmaları | Süre (μs) | | | Ortalama mutlak hata (16x16) |
|---|---|---|---|---|
| | 10x10 | 16x16 | 20x20 | |
| K-means algoritması yöntem 1 | 16,0774 | 9,8602 | 7,6102 | 24,788 |
| K-means algoritması yöntem 2 | 27,0524 | 21.536 | 17,5392 | 30,467 |
| Görüntü bölütleme algoritması | 33,1904 | 31,8084 | 29,9584 | 29,924 |
| En yakın komşu algoritması | 58,6284 | 44,4598 | 37,4765 | 43,085 |
| Yarışmacı öğrenme ağı | 319,2366 | 170,1892 | 131,138 | 8,467 |

Yarışmacı öğrenme ağı, işlem zamanı en uzun süren algoritma olmasına rağmen en az ortalama mutlak hata miktarına sahip olduğundan etkili bir yöntem olarak görülmektedir. K-means algoritması yöntem 1 en az işlem zamanına sahip olup yarışmacı öğrenme ağından sonra en az ortalama mutlak hata miktarına sahiptir. Sonrasında görüntü bölütleme algoritması, k-means algoritması yöntem 2 ve en yakın komşu algoritması gelmektedir. Makro blok boyutu küçüldükçe işlem zamanlarının artmasının nedeni makro blokların sayısının artmasıdır.

Hareket yoğunluğunun ve yönünün tespit edilmesi ile ilgili yapılacak sonraki çalışmalarda yazılım geliştirilerek bir PDA (Personal Digital Assistant) yardımıyla IP kameraya bağlanılıp, işlemler PDA aracılığıyla yapılabilir. Kullanılan IP kameranın sayısı çoğaltılarak insan topluluklarının hareketi bir çok noktadan daha iyi izlenebilir. Yapay sinir ağları haricinde farklı bir yapay zeka tekniği kullanılarak çalışma geliştirilebilir.

## KAYNAKLAR


1. Özden, M. ve Polat, E., "Mean shift ve kernel yoğunluk tahmini ile görüntülerde nesne takibi", ***Akıllı Sistemlerde Yenilikler ve Uygulamaları Sempozyumu***, İstanbul, 70-73 (2004).

2. Telatar, Z., "İmge dizilerinde objelerin hareket yönünün kestirimi", ***IEEE SIU 9. Sinyal İşleme ve Uygulamaları Kurultay***ı, Gazimağusa, Kıbrıs, 641-646 (2001).

3. Brox, T., Rosenhahn, B., Cremers, D. and Seidel, H. P., "Nonparametric density estimation with adaptive, anisotropic kernels for human motion tracking", ***Workshop on Human Motion***, Rio de Janeiro, Brazil, 152-165 (2007).

4. Shao, L., Ji, L., Liu, Y. and Zhang, J., "Human action segmentation and recognition via motion and shape analysis", ***Pattern Recognition Letters***, In Press, (2011).

5. McKenna, S. J., Jabri, S., Duric, Z., Wechsler, H. and Rosenfeld, A., "Tracking groups of people", ***Computer Vision and Image Understanding***, 80 (1): 42-56 (2000).

6. Maduro, C., Batista, K., Peixoto, P. and Batista, J., "Estimation of vehicle velocity and traffic intensity using rectifield images", ***IbPRIA 4th Iberian Conference on Pattern Recognition and Image Analysis***, Povoa de Varzim, Portugal, 64-71 (2009).

7. Mohana, H. S., Ashwathakumar, M. and Shivakumar, G., "Vehicle detection and counting by using real time traffic flux through differential technique and performance evaluation", ***ICACC International Conference on Advanced Computer Control***, Singapore, 791-795 (2009).

8. Ozkurt, C. and Camci, F., "Automatic density estimation and vehicle classification for traffic surveillance systems using neural networks", ***Mathematica and Computational Applications An International Journal***, 14 (3): 187-196 (2009).

9. Yao, W., Hinz, S. and Stilla, U., "Extraction and motion estimation of vehicles in single pass airbone LİDAR data towards urban traffic analysis***", ISPRS Journal of Photogrammetry and Remote Sensing***, 66: 260-271 (2011).

10. Pasha, A., "Hava trafik kontrolu benzetiminde etkileşimli çoklu model (interacting multiple model - IMM) kestirim performansı ve kalman filtresi ile karşılaştırılması", ***Havacılık ve Uzay Teknolojileri Dergisi***, 3 (4): 25-36 (2008).

11. Bilge, H. Ş., "Yapay açıklığa dayalı ultrasonik görüntüleme için hareket tahmini", Yüksek Lisans Tezi, ***Kırıkkale Üniversitesi Fen Bilimleri Enstitüsü***, Kırıkkale, 1-25 (1997).

12. Yıldırım, M., "Abdominal mr görüntülerinde hareket tahmini", Yüksek Lisans Tezi, ***Gebze Yüksek Teknoloji Enstitüsü Mühendislik ve Fen Bilimleri Enstitüsü***, Gebze, 15-31 (2007).

13. Eslami, A., Jahed, M. and Preusser, T., "Joint edge detection and motion estimation of cardiac MR image sequence by a phase field method" ***Computers in Biology and Medicine***, 40: 21-28 (2010).

14. Koyuncu, E., Ceylan, O. ve Yeniçeri, R., "Bilgisayarla görü tabanlı hareketli cisim yörüngesi izleyen robot kol tasarımı", ***Otomatik Kontrol Ulusal Toplantısı***, İstanbul, 291-296 (2005).

15. Ogawara, K., Tanabe, Y., Kurazume, R. and Hasegawa, T., "Detecting repeated motion patterns via dynamic programming using motion density", ***IEEE International Conference on Robotics and Automation***, Kobe, Japan, 1743-1749 (2009).

16. Telatar, Z., "Video dizilerinde ayrıt sezmeye dayalı hızlı hareket kestirimi", ***Gazi Üniversitesi Mühendislik Mimarlık Dergisi***, 24 (2): 245-255 (2009).

17. Bazin, J. C., Demonceaux, C., Vasseur, P. and Kweon, I. S., "Motion estimation by decoupling rotation and translation in catadioptric vision", ***Computer Vision and Image Understanding***, 114: 254-273 (2010).

18. Kim, J. H., Chung, M. J. and Choi, B. T., "Recursive estimation of motion and a scene model with a two camera system of divergent view", ***Pattern Recognition***, 43: 2265-2280 (2010).

19. Kim, J. S., Hwangbo, M., and Kanade, T., "Spherical approximation for multiple cameras in motion estimation: Its applicability and advantages", ***Computer Vision and Image Understanding***, 114: 1068-1083 (2010).

20. Görgünoğlu, S., Altay, Ş. and Şen, B., "Estimation of Dominant Motion Direction on Video Sequences", ***2nd International Symposium onComputing in Science & Engineering***, Kuşadası, Turkey, (2011).

21. Özden, M., "Ortalama kayma algoritmasının geliştirilerek görüntü dizilerinde hareketli nesne takibi ve görüntü kümeleme amaçlı kullanılması", Yüksek Lisans Tezi, ***Kırıkkale Üniversitesi Fen Bilimleri Ensitüsü***, Kırıkkale, 1-3 (2005).

22. Atalar, M., "Görüntü dizilerindeki artıklıkların temizlenmesi", Yüksek Lisans Tezi, ***Gazi Üniversitesi Fen Bilimleri Enstitüsü***, Ankara, 42-44 (2008).

23. Alımlı, H., "Motion estimation for video sequence", Master of Science Thesis, ***Dokuz Eylül University Graduate School of Natural and Applied Sciences***, İzmir, 13-15 (2008).

24. Şimşek, M., "Görüntü dizilerinde hareket tahmini seminer raporu", ***Gebze Yüksek Teknoloji Enstitüsü Bilgisayar Mühendisliği Bölümü***, Gebze, 12-18 (2006).

25. Yalçın, S., "H.264 motion estimator design", Master of Science Thesis, ***Sabancı Univeristy Graduate School of Engineering and Natural Sciences***, İstanbul, 9-11 (2005).

26. Bovik, A., "Handbook of Image and Video Processing", ***Academic Press***, San Diego, USA, 10-11 (2000).

27. Özkan, Y., "Veri madenciliği yöntemleri", ***Papatya Yayıncılık***, İstanbul, 131-148 (2008).

28. Erdoğan, Ş. Z., "Veri madenciliği ve veri madenciliğinde kullanılan k-means algoritmasının öğrenci veri tabanında uygulanması", Yüksek Lisans Tezi, ***İstanbul Üniversitesi Sosyal Bilimler Enstitüsü***, İstanbul, 47-55 (2004).

29. Dinçer, E., "Veri madenciliğinde k-means algoritması ve tıp alanında kullanılması", Yüksek Lisans Tezi, ***Kocaeli Üniversitesi Fen Bilimleri Enstitüsü***, Kocaeli, 55-56 (2006).

30. Böhm, C., "Powerful database support for high performance data mining", Habilitationsschrift, ***Ludwig-Maximilians Universitat Fach Informatik***, München, Deutschland, 171-172 (2001).

31. Elmas, Ç., "Yapay sinir ağları", ***Seçkin Yayıncılık***, Ankara, 30-35:153-160 (2003).

32. Cin, İ., "Şifre sorgulamada yapay sinirsel ağların kullanımı", Yüksek Lisans Tezi, ***Osmangazi Üniversitesi Fen Bilimleri Enstitüsü***, Eskişehir, 31-34 (1996).

# ÖZGEÇMİŞ

Şafak ALTAY 1986 yılında Erzurum da doğdu. İlk, orta ve lise öğrenimini Zonguldak'ın Devrek ilçesinde tamamladı. 2004 yılında Süleyman Demirel Üniversitesi Teknik Eğitim Fakültesinde Elektronik ve Bilgisayar Eğitimi Bölümü Bilgisayar Sistemleri Öğretmenliği Programında lisans öğrenimine başladı. 2008 yılında mezun oldu. 2009 yılında Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Bilgisayar Eğitimi Anabilim Dalı'nda yüksek lisans öğrenimine başladı. Bir dönem sonunda Karabük Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Bilgisayar Eğitimi Anabilim Dalı'na yatay geçiş yaptı. 2009 yılında Karabük Üniversitesi Teknik Eğitim Fakültesi Elektronik ve Bilgisayar Eğitimi Bölümü Bilgisayar Sistemleri Öğretmenliği programında araştırma görevlisi olarak göreve başladı. Halen Karabük Üniversitesi Teknik Eğitim Fakültesi Elektronik ve Bilgisayar Eğitimi Bölümü'nde araştırma görevlisi olarak görev yapmaktadır.

## ADRES BİLGİLERİ


Adres : Karabük Üniversitesi
Teknik Eğitim Fakültesi
Balıklarkayası Mevkii / KARABÜK

Tel : (0370) 4338200–1017
E-posta : safakaltay@karabuk.edu.tr