T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VİDEO İŞARETLERİNDE YAYA TAKİBİ

ABDULLAH ÖZEK

YÜKSEK LİSANS TEZİ
ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ ANABİLİM DALI
HABERLEŞME PROGRAMI

DANIŞMAN
DR. ÖĞR. ÜYESİ BÜLENT BOLAT

İSTANBUL, 2019

T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

VİDEO İŞARETLERİNDE YAYA TAKİBİ

Abdullah ÖZEK tarafından hazırlanan tez çalışması 07.01.2019 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Haberleşme Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Dr. Öğr. Üyesi Bülent BOLAT

Yıldız Teknik Üniversitesi

Jüri Üyeleri

Dr. Öğr. Üyesi Bülent BOLAT

Yıldız Teknik Üniversitesi _____

Dr. Öğr. Üyesi Nihan KAHRAMAN

Yıldız Teknik Üniversitesi _____

Dr. Öğr. Üyesi Gökalp TULUM

İstanbul Arel Üniversitesi _____

ÖNSÖZ

Tez çalışmalarım boyunca yardım ve katkılarıyla beni yönlendiren tez danışmanım Dr. Öğr. Üyesi Bülent Bolat hocama, arkadaşım Çağlar Çavdar'a ve sevgili aileme teşekkür ederim.

Ocak, 2019

Abdullah ÖZEK

İÇİNDEKİLER

Sayfa

SİMGE LİSTESİ vii

KISALTMA LİSTESİ viii

ŞEKİL LİSTESİ ix

ÇİZELGE LİSTESİ xi

ÖZET xii

ABSTRACT xiv

BÖLÜM 1

GİRİŞ 1

1.1 Literatür Özeti 1
1.2 Tezin Amacı 4
1.3 Orijinal Katkı 5

BÖLÜM 2

YAYA TAKİP YÖNTEMLERİ 6

2.1 Hareketli Nesne Tespiti 6
2.1.1 Arka Plan Çıkarımı Yöntemi 6
2.1.2 Sıralı Görüntüler Farkı Yöntemi 8
2.1.3 Optik Akış 9
2.2 Yaya Tespiti 9
2.2.1 Öznitelik Çıkarma 9
2.2.1.1 Bölge Tabanlı Öznitelik Çıkarımı 9
2.2.2 Sınıflandırıcılar 10
2.3 Yaya Takibi 10
2.3.1 Nokta Takibi 11
2.3.2 Çekirdek Takibi 11
2.3.3 Silüet Takibi 12

BÖLÜM 3

HAREKETLİ NESNE TESPİTİ 13

3.1 Ön Plan Çıkarımı 14
3.2 Gürültü Çıkarımı 17
3.2.1 Morfolojik İşlemler 17
3.2.1.1 Aşınma İşlemi 18
3.2.1.2 Genişleme İşlemi 18
3.2.1.3 Bağlı Bileşenler Analizi 19

BÖLÜM 4

YAYA TESPİTİ VE TAKİBİ 22

4.1 HOG Tanımlayıcı 23
4.1.1 Normalizasyon ve Gama Doğrulama 24
4.1.2 Yön Vektörlerini Hesaplama 26
4.1.3 Yön Vektörü Histogramı Oluşturma 27
4.1.4 Tanımlayıcı Bloklar 28
4.1.5 Blok Normalleştirmesi 29
4.2 Sınıflandırma 30
4.2.1 Destek Vektör Makinesi (SVM) Sınıflandırıcı 30
4.2.2 K-En Yakın Komşu (kNN) Sınıflandırıcı 33
4.2.3 Adaboost 35
4.3 Hareketli Nesne Takibi 37
4.3.1 Kalman Takibi 37
4.3.1.1 Kalman Filtresi 38

BÖLÜM 5

VİDEO İŞARETLERİNDE YAYA TAKİBİ UYGULAMASI 40

5.1 Hareketli Nesne Tespiti 41
5.2 Yaya Tespiti 45
5.2.1 Özniteliklerin Çıkarılması 48
5.2.2 Sınıflandırma 50
5.3 Yaya Takibi 51

BÖLÜM 6

SONUÇ VE ÖNERİLER 53

6.1 Performans Ölçümü 54
6.1.1 Metrikleri Tanımı 54
6.1.1.1 Duyarlılık Oranı 55
6.1.1.2 Kesinlik Oranı 55
6.1.1.3 Doğruluk Oranı 55
6.1.1.4 Hatalı Pozitif Oranı 55
6.1.1.5 Karışıklık Matrisi 55
6.2 Performans Testi 56

6.3 Sonuçlar 57
KAYNAKLAR 58
ÖZGEÇMİŞ 62

SİMGE LİSTESİ

α	Öğrenme oranı
τ	Eşik seviyesi
σ	Değişim
μ	Ortalama
η	Gauss olasılık yoğunluk fonksiyonu
θ_j	Eşik
ω	Tahmin hatası vektörü
θ	Yön
A	Geçiş matrisi
i	Örnekleme indeksi
j	Sınıflandırıcı indeksi
m	Genlik
t	İterasyon sayısı
x_j	Öznitelik
w	Örnek ağırlık vektörü

KISALTMA LİSTESİ

Adaboost	Adaptive Boosting
C-HOG	Dairesel HOG
DN	Doğru Negatif
DP	Doğru Pozitif
GK	Gauss Karışımı (GMM - Gaussian Mixture Model)
HN	Hatalı Negatif
HOG	Histogram of Oriented Gradients
HP	Hatalı Pozitif
INRIA	Institut National de Recherche en Informatique et en Automatique
KKT	Karush Kuhn Tucker
KLT	Kanade Lucas Tomasi
kNN	k-Nearest Neigbors algorithm
MATLAB	Matrix Laboratory
MCMCDA	Markov Chain Monte Carlo Data Association
MIT	Massachusetts Institute of Technology
R-HOG	Dikdörtgen HOG
RGB	Red, Green, Blue - Kırmızı, Yeşil, Mavi
ROI	Region of Interest
SIFT	Scale Invariant Feature Transformation
SURF	Speeded Up Robust Feature
SVM	Support Vector Machine
VSAM	Video Surveillance and Monitoring

ŞEKİL LİSTESİ

Sayfa

Şekil 2. 1 Arka plan çıkarımı örneği 7
Şekil 2. 2 Geçici değişiklikleri bulma örneği 8
Şekil 2. 3 Nokta takibi 11
Şekil 2. 4 Çekirdek takibi 12
Şekil 2. 5 Silüet takibi 12
Şekil 3. 1 Genel mimarinin ön işlem adımı 14
Şekil 3. 2 Orjinal görüntü 16
Şekil 3. 3 Gauss karışımı sonrası görüntü 17
Şekil 3. 4 Aşındırılmış görüntü 18
Şekil 3. 5 Genişletilmiş görüntü 19
Şekil 3. 6 3, 4, 6 ve 8 bağlılık örnekleri 20
Şekil 3. 7 Bağlı bileşenler sonrası elde edilen ROI'ler 21
Şekil 4. 1 HOG öznitelik çıkarma aşamaları 24
Şekil 4. 2 Orjinal ve normalize olmuş görüntü örnekleri 25
Şekil 4. 3 Gama doğrulaması uygulanmış görüntü 26
Şekil 4. 4 Yön vektörü histogram dağılımı 27
Şekil 4. 5 R-HOG ve C-HOG şekli 28
Şekil 4. 6 Karar bölgesi örnekleri 31
Şekil 4. 7 SVM için maksimum marjlı hiper düzlemler 31
Şekil 4. 8 K-en yakın komşu mekanizması 34
Şekil 4. 9 Kalman filtre döngüsü 39
Şekil 5. 1 Video işaretlerinde yaya takibi için genel mimari yapı 41
Şekil 5. 2 Hareketli nesne tespiti akış şeması 42
Şekil 5. 3 Gauss karışımı sonucu 42
Şekil 5. 4 Aşınma işlemi sonucu 43
Şekil 5. 5 Genişleme işlemi sonucu 44
Şekil 5. 6 Hareketli nesne takibi bloğu sonucu elde edilen ROI'ler 44
Şekil 5. 7 Yaya tespit mimarisi 45
Şekil 5. 8 96x160 piksel boyutundan 64x128 piksel boyutuna dönüştürülme 46
Şekil 5. 9 INRIA veri kümesindeki negatif görüntü örnekleri 47
Şekil 5. 10 64x128 piksel boyutuna dönüştürülmüş negatif görüntü örnekleri 47
Şekil 5. 11 64x128 piksel boyutuna dönüştürülmüş negatif görüntü örnekleri 48
Şekil 5. 12 Kullanılan hücre ve blok yapısı 49
Şekil 5. 13 Yön vektörleri çıkarılan görüntü 50

Şekil 5. 14 Sınıflandırma sonuçları 50
Şekil 5. 15 Önerilen sınıflandırıcı ile elde edilen sonuç..... 51
Şekil 5. 16 Video görüntüsünde yaya takibi pencereleri 52

ÇİZELGE LİSTESİ

Sayfa

Çizelge 4. 1 Yön vektörleri türev maskesi 26
Çizelge 4. 2 Adaboost algoritması akışı 36
Çizelge 6. 1 Değerlendirme metrikleri için terminoloji çizimi 54
Çizelge 6. 2 Karışıklık matrisi değerleri 56
Çizelge 6. 3 Performans sonuçları 56

ÖZET

VİDEO İŞARETLERİNDE YAYA TAKİBİ

Abdullah ÖZEK

Elektronik ve Haberleşme Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Tez Danışmanı: Dr. Öğr. Üyesi Bülent BOLAT

Gelişen teknoloji ile birlikte artan görüntü işleme çalışmaları, bu alanda çalışan sistemlerin uygulama alanlarını hızla artırmaktadır. Görüntü işleme çalışmalarında plaka tanıma, yüz tanıma, yaya tanıma gibi nesne tespiti uygulamaları çokça görülmektedir. Havaalanlarında, tren istasyonlarında, caddelerde ve alış veriş merkezlerinde çok sayıda insan sürekli olarak çeşitli yönlerde hareket etmektedir. Günümüzde mağazaların yerleşimleri satış stratejileri açısından oldukça önemlidir. İnsanlar alışveriş merkezlerine, havaalanlarına girdiklerinde bir yön eğilimlerine sahiptir. Bundan dolayı, tasarlanan sistem büyük alış veriş merkezlerinin, havaalanlarının girişlerine yerleştirilen kamera ile elde edilen videoları kullanarak, insanların potansiyel hareket yönlerinin çıkarılmasını sağlamaktadır. Böylece o bölgelerde yer alan mağazaların satış potansiyelinin daha fazla olabileceği düşünülmektedir. Yaya takip sistemleri, insanların potansiyel yön güzergâhlarının çıkarılması açısından oldukça önemli sistemlerdir.

Bu çalışmada nesne tespiti uygulamalarından olan yaya tespiti üzerine çalışılmış ve sabit bir kameradan kaydedilen videoda tespit edilen yaya takip edilerek izlediği güzergâh çıkarılmıştır. Çalışma temel olarak hareketli nesne tespiti, yaya tespiti ve yaya takibi olmak üzere üç aşamadan oluşmaktadır. Hareketli nesne tespiti aşamasında Gauss Karışımı (GK) algoritması kullanılarak arka plan çıkarımı yapılmış ve İlgi Alanı (ROI) tespit edilmiştir. Yaya tespiti aşamasında görüntünün özniteliklerini çıkarmak için tanımlayıcı olarak Yönlü Gradyanların Histogramı (HOG) algoritması, sınıflandırmak için Destek Vektör Makinesi (SVM), k En Yakın Komşu (kNN) ve Adaboost algoritmalarının

birleşimi ile çoğunluk oylamasına dayanan bir sınıflandırıcı geliştirilerek kullanılmıştır. Yaya takibi aşamasında ise kalman filtresi kullanılmıştır.

Gerçeklenen sistem, INRIA veri setinden alınan örnekler ile eğitilmiş ve test edilmiştir. Önerilen sınıflandırıcı kullanılarak video işaretlerinde yaya takibi gerçekleştirilmiş ve yayanın izlediği güzergâh çıkarılmıştır. Önerilen sınıflandırıcı kullanıldığında yayaların tespitinde diğer sınıflandırıcılara göre daha iyi bir oranda tespit yaptığı görülmüştür.

Anahtar Kelimeler: Hareketli nesne tespiti, yaya tespiti, yaya takibi, çoğunluk oylamasına dayalı sınıflandırma

ABSTRACT

PEDESTRİAN TRACKING ON VIDEO SIGNALS

Abdullah ÖZEK

Department of Electronics and Communications Engineering

MSc. Thesis

Adviser: Asst. Prof. Dr. Bülent BOLAT

With the help of advancing technology, the application area of image processing systems is getting broader. Object detection applications such as license plate recognition, face recognition, pedestrian recognition are widely seen in image processing studies. Many people in airports, train stations, streets and squares are constantly moving in various directions. Nowadays, the placement of stores is very important in terms of sales strategies. People have a direction when they enter the shopping centers and airports. Therefore, the designed system enables the removal of potential directions of movement of large shopping centers by using videos from the cameras placed in the entrances of the airports. Thus, it is thought that the sales potential of the stores located in those regions may be higher. Pedestrian tracking systems are very important systems in terms of people's potential route routes.

In this study, the pedestrian detection from the application of object detection is studied and the route, that the pedestrian is detected by tracking the pedestrian detected on the video recorded from a fixed camera, is obtained. The study is basically composed of three stages; moving object detection, pedestrian detection and pedestrian tracking. Background is extracted using Gauss Mixture Model (GMM) during the moving object detection phase and Region of Interest (ROI) is determined. Histogram of Oriented Gradients (HOG) is used as a descriptor to extract the visual attributes in the pedestrian detection phase, and a classifier based on majority basis with the combination of Support Vector Machine (SVM), k Nearest Neighbors (kNN)

and Adaboost algorithms is developed. In pedestrian tracking phase, Kalman filter is used.

The actual system is trained and tested with samples from the INRIA dataset. Using the proposed classifier, pedestrian tracking is performed on the video signals, and the track followed by the pedestrian track is acquired. It is observed that the more accurate results are obtained in the detection of pedestrian when the proposed classifier method is used.

Keywords: Moving object detection, pedestrian detection, pedestrian tracking, majority rating classification

BÖLÜM 1

GİRİŞ

1.1 Literatür Özeti

Günümüzde teknoloji hızla gelişmekte ve bilgisayarla görme alanında yapılan çalışmalar artmaktadır. Böylece insanların günlük ihtiyaçlarını karşılamak amaçlanmaktadır. Nesne takibi, ardışık görüntüler üzerinde çeşitli analiz metotlarıyla ortamdaki hareketli nesnenin görüntüdeki yerinin bulunması, hareketli nesnenin sınıflandırılması ve hareket doğrultusunun çıkarılarak takibi olarak tanımlanabilir. Nesne takibi için literatürde çeşitli çalışmalar yapılarak uygulamaların amaçlarına yönelik yöntemler geliştirilerek iyileştirmeler yapılmıştır.

Yaya tespiti ve takibi konularında, çok sayıda araştırma yapılmış ve birçok uygulama türü geliştirilmiştir. Bu bölümde, literatürde yer alan nesne tespit ve takip üzerine yapılan çalışmalar kısa özetler şeklinde anlatılacaktır.

1999 yılında Lowe nesne tespiti için boyuttan bağımsız özellik dönüşümü (Scale Invariant Feature Transformation – SIFT) algoritmasını kullanmıştır [1]. Bu algoritmada bir eğitim kümesi bulunmaktadır. Eğitim kümesindeki örnekler kullanılmakta ve görüntüdeki bölgesel özellikler kıyaslanmaktadır. Kıyaslama sonucunda doğru eşleşmenin olduğu alan kullanılarak tanıma işlemi yapılmaktadır. 2000 yılında Elgammal arka plan çıkarma yöntemiyle tespit yapmayı önermiştir [2]. Yaya tespitinin basit ve etkili yöntemlerinden biri arka plan çıkarmadır. Bu yöntem mevcut çerçeveden, arka plan çerçevesinin çıkarılmasıyla gerçekleştirilir. Bununla birlikte, tespit için çeşitli sınırlamalar vardır. Tespit için alınan video statik kamera tarafından alınmalıdır. Bu yöntem kullanıldığında; sabit nesneler tespit edilememekte, yaya

grubundan yayalar izole edilememekte veya diğer hareketli nesnelerden farklılaştırılamamaktadır.

2000 yılında Papageorgiou ve Poggio tarafından destek vektör makinesi (Support Vector Machine – SVM) ve HAAR dalgacık dönüşümü kullanılarak yeni bir nesne tanıma yöntemi geliştirilmiştir [3]. Geliştirilen yöntemde pozitif ve negatif veri seti kullanılmış, Haar dalgacık dönüşümü ile görüntüden öznitelikler çıkarılmış ve SVM ile çıkarılan öznitelikler sınıflandırılarak sistem eğitilmiştir.

2003 yılında Viola ve Jones videodaki iki ardışık çerçeveyi hesaba katarak bir algoritma geliştirmiştir [4]. Ardışık çerçevelerdeki görüntülerde değişim bilgisi ve piksel parlaklığı kullanılarak yaya tespit eden bir yöntem önerilmiştir. HAAR dalgacık dönüşümü ile hareket ve görünüş bilgilerine dayalı olarak çıkarılan öznitelikler Adaboost (Adaptive Boosting) tabanlı algılayıcı ile sınıflandırmıştır. Bu yöntem, çok küçük ölçeklerde yayaların tespit edilmesinde bile çok etkilidir. Bu yöntemde hatalı pozitif tespitlerin sayısı oldukça küçüktür.

INRIA'da çalışan iki araştırmacı olan Dalal ve Triggs 2005 yılında Yönlü Gradyanların Histogramı (Histogram of Oriented Gradients – HOG) tanımlayıcı ve SVM sınıflandırıcı kullanarak durağan görüntülerde yer alan yayaları tespit eden bir yöntem önermişlerdir [5]. Önerilen bu yöntemde öznitelik çıkarmak amacıyla HOG algoritması kullanılmıştır. HOG ile özniteliklerin çıkarılması için 3x3 hücrelerden oluşan blok, 6x6 pikselden oluşan hücre ve 9 öğeden oluşan yön vektörü histogramı kullanılmış ve HOG tanımlayıcısı oluşturulmuştur. Çıkarılmış olan öznitelikleri sınıflandırmak için doğrusal SVM kullanılmıştır. INRIA [6] ve MIT [7] yaya veri tabanları kullanılarak performans testleri yapılmış ve test sonuçlarında %85 e varan başarı elde edilmiştir. 2006 yılında Dalal ve Triggs tarafından yapılan bir diğer çalışmada ise HOG algoritması ve hareket tabanlı tanımlayıcılar beraber kullanılarak, hareketli görüntülerde yaya tespit eden bir yöntem geliştirilmiştir [8].

2007'de Sabzmeydani ve Mori durağan görüntülerde yaya tespit problemi üzerine çalışmışlar ve Shapelet özniteleyicilerini kullanan algoritma önermişlerdir [9]. Shapelet özniteleyicileri oluşturulurken Adaboost zayıf sınıflandırıcısı kullanılmış ve yönlü gradyan cevabının bir türevi hesaplanmıştır. Ayrıca Adaboost sınıflandırma amaçlı

olarak da kullanılmıştır. 2007 yılında Wu ve Nevatia, Edgelet tabanlı bir tanıma algoritması önermiştir [10]. Önerilen algoritmada, insan vücudunu parçalara ayırmak ve insan vücudunun farklı parçaları kullanılarak kolayca tespit yapmak amaçlanmıştır.

2008 yılında Lin ve Davis tarafından bir duruş tanımlayıcısı önerilmiştir [11]. İnsan şekillerini ve duruşlarını eş zamanlı olarak bölümlere ayırmış ve görüntünün insan içerip içermediğini sınıflandırmayı başarmıştır. Sınıflandırmanın performansı HOG özniteliklerine ve çekirdek SVM sınıflandırıcılarına dayanmaktadır.

2016 yılında Cai vd. geliştirilmiş HOG özniteliklerine dayalı görüntü sınıflandırıcı algoritması önermiştir [12]. Önerilen algoritmada nesne özniteliklerini tanımlamak ve bunları güçlendirmek için HOG kullanılmıştır. Sınıflandırmayı daha belirgin hale getirmek için SVM ve k en yakın komşu (k nearest neighborhood - kNN) algoritmaları birleştirilerek bir sınıflandırıcı oluşturulmuştur.

Literatürde yer alan takip yöntemlerine bakıldığında 1986 yılında Broida ve Chellappa, insan vücudunun hareket parametrelerini tahmin eden filtreleme yaklaşımına dayanan bir yöntem önermiştir [13]. 1994 yılında ise Shi takip sırasında afin görüntü değişiklikleri ve ayırt edici özellikler modeline dayanan takip algoritması önermiştir [14]. 2007'de Avidan takip işlemini bir ikili sınıflandırma problemi olarak ele almıştır [15]. Adaboost'u kullanarak, bir grup çevrimiçi eğitimli zayıf sınıflandırıcıdan, güçlü bir sınıflandırıcı oluşturmuştur. Bu yeni güçlü sınıflandırıcı, bir sonraki çerçevenin bir güven haritasını hesaplamak için kullanılmıştır.

2001 yılında Masoud ve Papanikolopoulos model tabanlı yaya takip sistemini önermiştir [16]. Bu sistemde tek bir kamera kullanılmıştır ve bu sistem üst üste binme, belirsizlik gibi birçok zor durumda çalışmıştır. Yaya görünürken, sistem bu yayaların yerini ve hız bilgilerini üretmiştir. İyi bilinen takip yöntemlerinden biri olan kalman filtresi, 1960 yılında R.E. Kalman tarafından yayınlanmıştır [17]. Bu filtre ayrık veri doğrusal filtreleme problemini çözmek için kullanılmaktadır. Bu filtre ile pek çok araştırma yapılmış ve pek çok uygulama geliştirilmiştir.

Çok yönlü takip-tespit algoritmaları için birçok yöntem bulunmaktadır. Benfold 2011 yılında çok parçalı olan çoklu hedef takip sistemini önermiştir [18]. Eşzamanlı Kanade-Lucas-Tomasi (KLT) takibi ile sistem asenkron HOG tespitlerini toplamıştır. Yüksek

çözünürlükteki videoda gerçek zamanlı takip gerçekleştirmek için Markov-Zincirli Monte-Carlo Veri Birliği'ni (Markov chain Monte Carlo data association - MCMCDA) kullanmıştır.

Kalabalık bir sahnede çoklu hedef takibi için bir diğer yöntem Kuo tarafından 2010 yılında önerilmiştir [19]. Takip edilen nesnelerin eğitim örnekleri, zaman kaydırmalı pencereden çevrimiçi olarak toplanmştır. Tanımlayıcıların ve benzerlik ölçümlerinin birleştirilmesi için Adaboost algoritması kullanılmıştır.

1.2 Tezin Amacı

Günümüzde teknoloji hızla gelişmekte ve bilgisayarlar ile yapılan görü çalışmaları hızla artmaktadır. Yapılan çalışmalarda bilgisayarların gördükleri nesneleri bir insan gibi iyi bir şekilde tanımaları, bilgisayarlarla yapılan çalışmaların temel amaçlarındandır. Bilgisayarla yapılan görüntü işleme çalışmaları genellikle plaka tanıma, yüz tanıma, insan tanıma gibi nesne tespiti uygulamaları üzerine özelleşmiştir.

İnsanın tespit edilmesi yıllar boyunca incelenmiş olmasına rağmen, görünüm varyasyonları, şekil değişimleri ve bakış açısı değişimleri nedeniyle hala zorlu bir problemdir. Değişkenlik problemini ele almak bu tez kapsamındadır. Nesne takip uygulamaları, birden fazla nesneyi izleyebilmeli, nesnelerin sahneye girmelerini ve sahneden çıkmalarını fark edebilmelidir. Bu tez çalışmasında, gerçek zamanlı gereksinim olmaksızın bir videodaki yayaları tespit etme ve tespit edilen yayaları takip etme kabiliyetlerine sahip bir sistemin geliştirilmesi amaçlanmıştır.

Bu tez çalışmasında, Gauss karışımı algoritması (Gauss Mixture Model - GK), HOG algoritması, SVM, kNN, Adaboost sınıflandırma algoritmaları ve kalman filtresi kullanılarak bir yaya takip uygulaması gerçekleştirilmiştir. Bu çalışmada SVM, kNN ve Adaboost algoritmaları yardımıyla yeni bir sınıflandırıcı oluşturulmuştur. Oluşturulan sınıflandırıcı çoğunluk oylamasına dayanmaktadır. Deneysel sonuçlar incelendiğinde önerilen yeni sınıflandırıcı kullanılarak yaya tespiti yapıldığında tek bir sınıflandırıcı ile yapılan tespite göre daha yüksek doğruluk oranında sonuçlar elde edilmiştir.

Bu çalışma MATLAB (Matrix Laboratory) ortamında gerçekleştirilmiş, yaya takibi yapılacak olan video statik kamera tarafından alınmıştır. Bilgisayarların

performanslarının farklılığından dolayı geliştirilen uygulama çalıştırılırken, çalışma süreleri değişmektedir. Kullanıcının videodaki yayaların takip sonuçlarını izlerken bilgisayar donanımına ve performansına bağlı olmaması için, sonuçlar bir videoda kaydedilmiştir. Uygulamaya girdi olarak bir video verilmekte ve sonuç olarak yaya takibi yapılan bir video elde edilmektedir.

1.3 Orijinal Katkı

Bu çalışmada uygulanan yöntemle yaya tespit sisteminin başarısının artırılarak yaya takibi yapmaya çalışılmıştır. Komite oylamasına dayalı olarak geliştirilen sınıflandırıcı kullanılmış ve yaya tespitinde doğruluk oranı artırılmaya çalışılmıştır. Yapılan çalışmalar ilerleyen bölümlerde detaylı bir biçimde anlatılmaktadır.

Bu tez çalışmasında, üç aşamalı bir yaya takip uygulaması gerçekleştirilmiştir. Bu aşamalar hareketli nesne tespiti, yaya tespiti ve yaya takibidir. Hareketli nesne tespiti gerçekleştirilirken arka plan çıkarımı yöntemi kullanılmış ve bu yöntem sayesinde potansiyel hedeflere sahip olabilecek ilgi alanları (Region of Interest - ROI) oluşturulmuştur. Yaya tespit aşamasında eğitim ve tespit olmak üzere iki faz bulunmaktadır. Eğitim aşamasında sistem INRIA veri seti kullanılarak eğitilmiştir. Eğitim gerçekleştirilirken görüntüdeki öznitelikleri çıkarmak amacıyla HOG tanımlayıcısı, sınıflandırmak için SVM, kNN ve Adaboost sınıflandırıcıları kullanılmıştır. Tespit aşamasında hareket bilgisi yardımıyla çıkarılan ROI'ler kullanılmış ve bu ROI'lerin öznitelikleri HOG tanımlayıcı ile elde edilmiştir. Elde edilen öznitelikleri sınıflandırmak için SVM, kNN ve Adaboost algoritmaları yardımıyla oluşturulan yeni bir sınıflandırıcı kullanılmıştır. Yeni oluşturulan bu sınıflandırıcı çoğunluk oylamasına dayanmaktadır. Üç sınıflandırıcıdan herhangi ikisinin yaya olarak tespit ettiği nesne, yaya olarak kabul edilmektedir. Yaya takibi aşamasında ise kalman filtresi kullanılmış ve tespit edilen yayalar takip edilerek gittikleri güzergâh çıkarılmıştır.

BÖLÜM 2

YAYA TAKİP YÖNTEMLERİ

Yaya takibi çeşitli aşamalardan oluşmaktadır. Bu aşamalar, hareketli nesne tespiti, yaya tespiti ve yaya takibi olarak listelenebilir. Her bir aşama için literatürde çeşitli yöntemler bulunmaktadır. Çalışmamızda kullanılan yöntemler detaylı bir biçimde ele alınmıştır. Bununla birlikte kapsamlı bir bütünlük olması açısından bu bölümde kullanılan yöntemler hakkında kısaca bilgi verilmektedir.

2.1 Hareketli Nesne Tespiti

Hareket eden nesneleri tespit etmek için çeşitli tespit yöntemleri bulunmaktadır. Bu yöntemler videolarda yer alan hareketli nesnelerin tespit edilmesi ve tespit edilen nesnelerin hareket güzergâhlarının çıkarılması için kullanılmaktadır. Hareket eden nesnelerin tespiti, nesne takip sistemlerinin ilk kısmını oluşturur ve sistemin performansını artırıcı etkisi vardır.

2.1.1 Arka Plan Çıkarımı Yöntemi

Arka plan çıkarımı yöntemi kullanılarak hareketli nesnenin bulunması için görüntünün sabit kamera ile alınması ve hedef nesnenin hareketli olması gerekmektedir. Bu yöntem kullanılırken ilk olarak arka plan görüntüsünün belirlenmesi gerekir. Bu görüntü sistem çalışmadan veya çalışmaya başlayınca alınan görüntüdür. Video çerçevesi içerisinde hareket eden nesneleri tespit etmek için her bir görüntüden arka plan görüntüsü çıkarılır [23]. Elde edilen sonucun mutlak değeri alınarak önceden belirlenmiş olan (τ) eşik seviyesi (2.1) değeriyle karşılaştırılır.

$$|görüntü_i(x,y) - arka\ plan(x,y)| > \tau \quad (2.1)$$

Görüntüden, arka plan görüntüsünün çıkarılmasıyla elde edilen fark değeri, belirlenen eşik seviyesinden büyükse ilgili koordinatta değişiklik olduğu düşünülmektedir. Ancak sadece bu kritere bakılarak ilgili koordinatta hareket olduğu kesin olarak söylenemez. Işık farklılığından dolayı bu kriter sağlanabilir. İlgili alanda hareketli bir nesnenin olduğunu söylemek için değişim olan piksellerde kapalı bir alan oluşması ve eşik seviyesinden büyük olması gerekir.

Bağlı bileşen analizi yöntemi yapılarak birbirine bağlı olan bölgeler tespit edilir. Bu yöntemin uygulanmasıyla ışık farklılığından dolayı değişiklik olan pikseller elenmiş olur. Böylece hareket eden nesneler tespit edilmiş olur (Şekil 2.1).



Şekil 2.1 Arka plan çıkarımı örneği

Hareketli nesnelerin belirlenmesinden sonra gürültü etkisini azaltmak ve tespit edilen nesneyi belirginleştirmek için aşınma, açma ve kapama gibi morfolojik işlemler uygulanır. Yüksek verimde arka plan tespit yapmak için alınan referans görüntüsü, yeni görüntüler geldikçe zamanla yenilenmelidir. Arka plan çıkarım algoritmasının yüksek verimde çalışması için;

- Işık Değişimleri
- Ufak Kamera Hareketleri
- Ortamda bulunan diğer hareketler (Ağaç yaprakları, dalgalar vb.)

gibi çevresel faktörler düşünülerek tasarım yapılmalıdır.

Arka plan çıkarma yöntemine gauss karışımı [24], kernel yoğunluk hesaplamaları [25] ve ortalama kaydırma öbeklemesi [26] yaygın olarak kullanılan metotlar olarak örnek verilebilir.

2.1.2 Sıralı Görüntüler Farkı Yöntemi

Sıralı görüntüler farkı yöntemi, bir video dizisinde ardışık karelerin (iki veya üç) piksel-piksel farkını kullanarak hareketli bölgeleri tespit etmeye çalışır. Lipton vd. (2.2) eşitliğini sağlayan piksellerin ön plana ayrıldığı iki çerçeveli bir farklılık şeması sunmuştur [20]. (2.2)'ye göre çervelerden elde edilen fark belirlenen eşik seviyesinden (τ) büyükse orada hareketli bir nesne bulunmaktadır.

$$|I_t(x,y) - I_{t-1}(x,y)| > \tau \tag{2.2}$$

Bu yöntem çok basit, hızlı ve dinamik sahne değişikliklerine oldukça uyumludur, ancak eşik değerine çok duyarlıdır. Bundan dolayı bazı hareketli nesnelerin tüm ilgili piksellerini tespit etmede başarısız olur. Hatalı hareket tespiti için örnek bir nesne Şekil 2.2'de gösterilmiştir. Sol taraftaki yayanın tek renkli bölgesi, bu yöntemin, yayanın hareketli bölgesinin tüm piksellerini ayıklamakta başarısız olmasını sağlamıştır. Tespit edilen hareketli bölgeler kırmızı piksellerle işaretlenmiştir.



(a) (b)

Şekil 2.2 Geçici değişiklikleri bulma örneği (a) İki hareketli nesneye sahip örnek bir sahne (b) Geçici değişiklikleri bulma sonucu

Bazı durumlarda iki çerçeve farklılığının eksikliklerinin üstesinden gelmek için üç çerçeve ayrımı kullanılabilir [21]. Örneğin, Collins vd. VSAM (Video Surveillance and

Monitoring) projesi için üç çerçeve farklılığını uyarlamalı bir arka plan çıkarma modeli ile birleştiren bir melez yöntem geliştirmişlerdir [22]. Hibrit algoritma, zamansal ayrıştırma ve arka plan çıkarma hataları olmaksızın videodaki hareketli bölgeleri başarılı bir şekilde bölümlere ayırmıştır.

2.1.3 Optik Akış

Optik Akış, bir görüntü dizisindeki hareketli bölgeleri tespit etmek için zaman içinde hareketli nesnelerin akış vektörlerinin özelliklerini kullanır. Videolarda yer alan hareketleri, hareketli bir kameradan bile algılayabilirler, ancak optik akış yöntemlerinin çoğu sayısal olarak karmaşıktır ve özel donanım olmadan gerçek zamanlı olarak kullanılamaz [27]. Optik akış yönteminin dezavantajlarından biri, parlaklık sabitliği ve mekânsal düzgünlük varsayımı altında tutarlı hareketi çıkarmasıdır.

2.2 Yaya Tespiti

Yaya tespiti, yayaları arka plandan ayırma görevidir. Günümüzde yaya tespit tekniklerinde etkileyici bir ilerleme sağlanmasına rağmen aydınlatma, bakış açısı değişimleri gibi nedenlerden kaynaklanan zorluklara sahiptir. Yaya tespiti için öncelikle özniteliklerin çıkarılması ardından sınıflandırılması gerekmektedir.

2.2.1 Öznitelik Çıkarma

Öznitelik çıkarma işleminde, aynı sınıfa ait örnekler için bir öznitelik vektörü oluşturulur. Yaya tespiti için sınıflandırma öncesinde önemli bir adımdır. Öznitelik vektörlerini elde etmek için, kilit noktalar (öznitelikler) tespit edilir. Bilgisayarlı görü çalışmalarında birçok önemli öznitelik çıkarma dedektörü vardır.

2.2.1.1 Bölge Tabanlı Öznitelik Çıkarımı

Bölge tabanlı öznitelikler genellikle bir nesnenin tüm görüntü bölgesinden çıkarılır. Bir nesnenin görüntü bölgesi, genellikle ön plan çıkarma algoritmaları tarafından çıkarılan ön plan alanıdır. Görüntü bölgesinin uzunluğu, yüksekliği, boyutu, alanı SIFT, SURF (Speeded Up Robust Feature) ve HOG tanımlayıcıları için genellikle bir öznitelik vektörü oluşturmak için kullanılır.

Bir SIFT özelliği, görüntüdeki göze çarpan noktaların görünümünü açıklar. Görüntü döndürülse veya aydınlatma değiştirilse bile SIFT belirgin kalır. SIFT özellikleri, aynı nesnenin iki farklı görüntüsünde noktadan noktaya benzerlikleri bulur [1].

SURF tanımlayıcıları Bay vd. [28] tarafından önerilmiştir. Bu tanımlayıcı, SIFT gibi resimler arasında benzeşmeleri bulmak için kullanılır. SURF tanımlayıcıları tasarım performans kaybına izin vererek hesaplama hızına odaklanır. SIFT 'te kullanılan gauss filtreleri yerine kutu filtreleri kullanılmaktadır, bu da hesaplama karmaşıklığını azaltmaktadır [1].

HOG kavramı Dalal ve Triggs tarafından önerilmiştir. HOG algoritmasıyla öznitelik vektörünü hesaplamak için giriş görüntüsünün gradyanı hücrelere bölünür. Her hücre için piksel cinsinden gradyan yönelimlerinin bir histogramı hesaplanır. Görüntü için bir global öznitelik vektörü oluşturmak amacıyla tüm hücrelerin vektörleri birleştirilir. HOG tanımlayıcıları yayaların tespiti için sıklıkla kullanılmıştır [5].

Bu tez çalışmasında, yayaları sınıflandırmak için tanımlayıcı olarak HOG kullanılmıştır. Bölüm 4'te, HOG daha ayrıntılı olarak tanıtılacaktır.

2.2.2 Sınıflandırıcılar

Sınıflandırıcılar, çıkarılan öznitelik vektörünü kullanarak bilinmeyen bir nesne örneğini bilinen bir sınıfa atar. Bu işlem, eğitim verilerinden öğrenilen bilgilere dayanır. Makine öğrenimi algoritmaları, eğitim verilerini kullanarak sınıflandırıcıları oluşturur. Öğrenme algoritmalarının önemli bir özelliği, eğitim verileri ile kontrolüdür. Yaya tespitinde SVM, kNN ve Adaboost algoritmalarının sınıflandırıcı olarak kullanıldığı görülmüştür [4], [12]. Bu çalışmada bu sınıflandırıcılardan yararlanılmıştır. Bölüm 4'te, ayrıntılı olarak tanıtılacaktır.

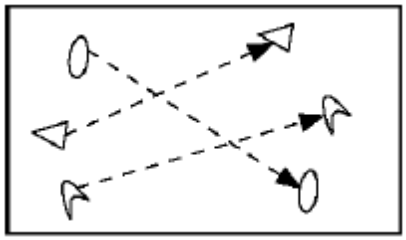
2.3 Yaya Takibi

Takip, tespit edilen nesneyi veya nesneleri video kareleri boyunca ilişkilendirme işlemidir. Nesnelerin takibi sırasında, karmaşık nesne hareketi, nesnelerin parçalı veya tamamen üst üste binmesi, görüntülerdeki aydınlatma değişimleri ve gürültüden dolayı

oluşan çeşitli zorluklar bulunmaktadır [29]. Takip yöntemleri nokta takibi, çekirdek takibi ve silüet takibi olmak üzere üç ana tipte sınıflandırılmıştır [29], [30]:

2.3.1 Nokta Takibi

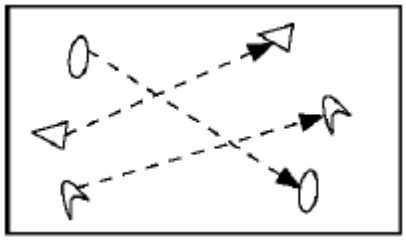
Nokta takibinde nesne noktalarla temsil edilir (Şekil 2.3). Nokta gösterimi, rotasyon, ölçek ve afine dönüşümü değişikliklerine karşı güçlüdür [30]. Bu yöntemde nesne tespiti için harici bir mekanizma gereklidir. Yanlış değerlendirmelerden kaynaklanabilecek nokta problemi vardır ve bu problem deterministik (aynı girişe aynı sonucu veren) yöntemlerle veya istatistiksel yöntemlerle çözülebilir. Deterministik yöntemler, izlenmesi gereken nesnenin sınırlayıcı yakınlığını, maksimum hızını ve katılığını çözer. Öte yandan, istatistiksel yöntemler, konum, hız, boyut ve ivme de dâhil olmak üzere nesnenin durum-uzay temsilini kullanır. Bu yöntemlerde durum tahmini genellikle kalman filtresi veya partikül filtresi ile yapılır.



Şekil 2.3 Nokta takibi

2.3.2 Çekirdek Takibi

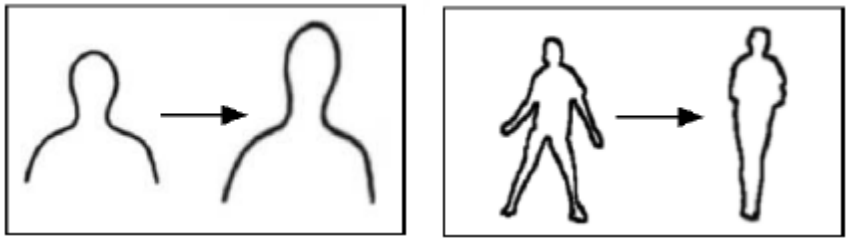
Çekirdek takibi, nesnenin dikdörtgen, elips veya dairesel bölge gibi geometrik bir bölge tarafından temsil edildiği nesne hareketinin hesaplanmasına dayanır (Şekil 2.4). Bu takip yöntemi, şablon modeli ve görünüm modeli olarak adlandıran iki alt kategoriye ayrılmıştır [29], [30]. Şablon modelleri; kareler arası farklılıkları, normalleştirilmiş çapraz korelasyon ve Battacharya katsayısı gibi benzerlik ölçütlerini kullanarak nesneyi eşleştirmeye dayanır. Optik Akış yöntemi, popüler bir çekirdek takip yöntemidir.



Şekil 2.4 Çekirdek takibi

2.3.3 Silüet Takibi

Nesneler genellikle basit geometrik şekillerden ziyade karmaşık şekillere sahiptir, bu nedenle silüet takipçileri, takip edilecek nesnelerin silüetlerini çıkarır ve önceki çerçevelerden oluşturulan nesne modelini kullanarak her bir çerçevede nesne bölgesini bulur. Silüet izleme yöntemleri şekil eşleme ve çevre çizgisi takibi olmak üzere iki alt kategoriye ayrılır. Şekil eşleştirme yöntemlerinde görüntüde nesnenin silüeti aranır. Çevre çizgisi izleme yönteminde ise, enerji fonksiyonları veya durum uzay modellerinin minimizasyonunu kullanılır. Böylece görüntünün o anki pozisyonunda çevre çizgisi eşleştirilir (Şekil 2.5) [29].



(a) (b)

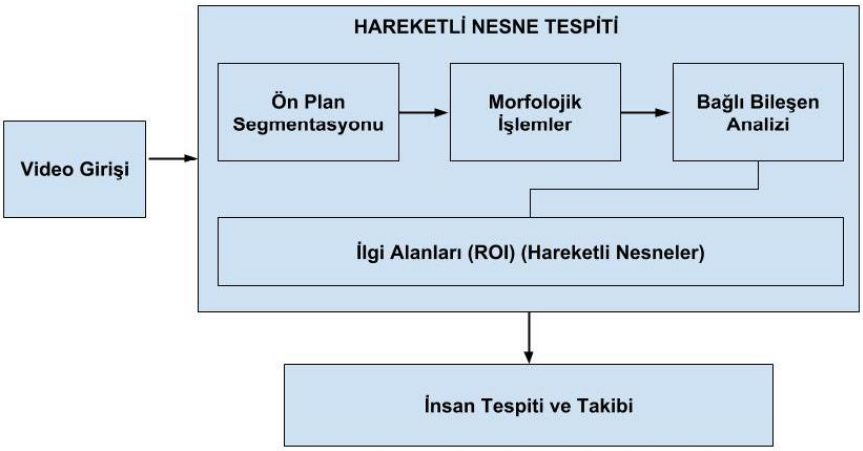
Şekil 2.5 Silüet takibi

BÖLÜM 3

HAREKETLİ NESNE TESPİTİ

Görsel gözetim sistemlerinde, genellikle hareket segmentasyonu veya ön plan çıkarımı olarak adlandırılır ve hareketli nesne tespitinin ilk adımıdır. Nesne takibi ve nesne sınıflandırması gibi işlemler, giriş olarak hareketli nesne tespiti modülünün çıkışını alır. Bu kısım yaya tespitinde sınıflandırma kısmında bir girdi oluşturmak için ilk adımdır. Bu nedenle, hareketli nesne tespiti performansı tüm sistemin genel performansını etkiler. Yaya takibi için yapılan çalışmada, ön plan çıkarımı yöntemi olarak arka plan çıkarımı yöntemlerinden olan GK algoritması tercih edilmiştir. Çünkü bu algoritma aydınlatma değişikliklerine ve tekrarlayan dağınıklıklara karşı, diğer ön plan çıkarımı algoritmalarına göre daha kararlıdır.

Yeni bir video çerçevesi alındığında, ön plan nesneleri GK algoritması tarafından çıkarılır. Bazı morfolojik işlemler uygulanarak gürültü kaldırılır. Sadece ilgili alanlara odaklanmak için bağlı bileşen analizi gerçekleştirilir ve ROI'ler tespit edilir. Tespit edilen ROI'lerde hareketli bir nesne bulunur. Hareketli bu nesnenin yaya olup olmadığını tespit etmek için, tespit edilen ROI'ler yaya tespiti bloğuna verilir. Hareketli nesne tespiti bloğunun akış şeması Şekil 3.1'de görülmektedir.



Şekil 3.1 Genel mimarinin ön işlem adımı

3.1 Ön Plan Çıkarımı

Bu çalışmada, ön plan çıkarımı için gauss karışım model yaklaşımı kullanılmıştır. Stauffer ve Grimson tarafından önerilen gauss karışım model yaklaşımı, görüntüdeki her pikseli bir gauss dağılımı karışımıyla modeller [31]. Her bir piksel, onu temsil eden gauss dağılımının arka plan modeline uygunluğuna göre sınıflandırılır.

Hareketli nesne tespiti yöntemlerinden olan gauss karışım model yaklaşımında, her pikselin son geçmişi {X1, ..., Xt}, K Gauss dağılımlarının karışımı olarak modellenir. Modelde verilen mevcut piksel değerinin gözlenme olasılığı (3.1) eşitliği ile hesaplanır.

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} * \eta\left(X_t, \mu_{i,t}, \sum_{i,t}\right) \tag{3.1}$$

(3.1)'de yer alan K değeri dağılımların sayısıdır. $\omega_{i,t}$ tahmini ağırlık (bu gauss dağılımı ile hesaplanan verinin oranı) ve bu ağırlığın t anındaki karışımda bulunan i. gauss dağılımı, $\mu_{i,t}$ ve$\sum_{i,t}$ t zamanındaki i'nince gauss ortalama değer ve kovaryans matrisidir. η gauss olasılık yoğunluk fonksiyonu (3.2) eşitliğinde verilmiştir.

$$\eta(X_t, \mu, \sum) = \frac{1}{(2\pi)^{\frac{n}{2}} |\sum|^{\frac{1}{2}}} e^{-\frac{1}{2}(X_t - \mu_t)^T \sum^{-1} (X_t - \mu_t)} \tag{3.2}$$

K değeri mevcut hafıza ve hesaplama gücü ile belirlenir. Genel olarak, 3 ile 5 arasında değişen değerler kullanılır. Ayrıca, hesaplama sonuçlarından, kovaryans matrisinin (3.3)'te verilen eşitlikteki gibi olduğu varsayılmaktadır.

$$\sum_{k,t} = \sigma_k^2 I \tag{3.3}$$

Kırmızı, yeşil ve mavi piksel değerlerinin birbirinden bağımsız olduğu ve aynı değişimlere sahip olduğu varsayılır. Bundan dolayı, görüntüde her pikselin son gözlenen değerinin dağılımı gaussların bir karışımı olarak tanımlanır. Her yeni piksel değeri, karışım modelinin ana bileşenlerinden biri ile temsil edilir ve karışımın bileşeninin parametrelerini güncellemek için kullanılır.

t anında K dağılımlarının öncelikli ağırlıkları hesaplanırken (3.4) eşitliği kullanılır.

$$\omega_{k,t} = \alpha * M(k,t) + (1-\alpha) * \omega_{k,t-1} \quad (3.4)$$

(3.4)'de α öğrenme oranı olarak ifade edilir, $M(k,t)$ eşlenen modeller için "1" , geri kalan modeller için "0" dır.

Eşlenmeyen dağılımlar için ortalama (μ) ve değişim (σ) parametreleri aynı kalır. Yeni gözlemlerle eşleşen dağılımın parametreleri (3.5) ve (3.6) da yer alan eşitlikler ile güncellenmektedir.

$$\mu_t = (1-\rho)\mu_{t+1} + \rho X_t \quad (3.5)$$

$$\sigma_k^2 = (1-\rho)\sigma_{k-1}^2 + \rho(X_t - \mu_t)^T(X_t - \mu_t) \quad (3.6)$$

Bu denklemlerde $\rho = \alpha\eta(X_t / \mu_k, \sigma_k)$ olarak ifade edilir. Bu denklemler, ağırlık güncelleme denkleminde kullanılan denklemlerle ve ortalama çalışma yöntemiyle mantıksal olarak aynıdır.

Görüntüde meydana gelen değişiklikler modele yansıtılarak ölçümler düzenlenmelidir. Bu nedenle, Gauss dağılımları aşağıdaki yaklaşımla değerlendirilmektedir. İlk olarak, Gauss dağılımları ω/σ değerlerine göre sıralanır. Bu değer dağılım kazancı daha fazla olduğunda ve değişim azaldıkça artar. Karışımın parametrelerini yeniden tahmin ettikten sonra, eşleşen dağılımdan en olası arka plan dağılımına doğru sıralama yapmak yeterlidir, çünkü sadece eşleşen modeller nispi değeri değişmiş olacaktır. Modelin bu sıralaması, büyük olasılıkla arka plan dağılımlarının en üstte kaldığı ve daha az muhtemel geçici arka plan dağılımlarının tabana doğru sıralandığı ve sonuçta yeni dağılımlar ile değiştirildiği, açık uçlu bir listedir. Daha sonra ilk B dağılımları, arka plan modeli olarak seçilir.

$$B = argmin_b(\sum_{i=1}^{K} \omega_k > T) \quad (3.7)$$

T, verilerin arka plan tarafından hesaba katılması gereken minimum kısmının bir ölçüsüdür. T için küçük bir değer seçilirse, arka plan modeli genellikle tek modeldir. Eğer T daha yüksekse, tekrarlayan bir arka plan hareketinin neden olduğu çok modlu bir dağılım, (örneğin, bir ağaç üzerindeki yapraklar, rüzgârda bir bayrak, bir inşaat flaşörü, vs.) arka plan modeline birden fazla rengin katılmasıyla sonuçlanabilir. Bu sonuç, arka planın iki veya daha fazla ayrı rengi kabul etmesini sağlayan bir saydamlık etkisi doğurur.

Gauss karışımı modeli uygulama adımları sıralandığında aşağıdaki gibi bir sırayla karşılaşılır:

İlk olarak gauss dağılımı bileşenleri sayısı, arka plan bileşenleri sayısı, öğrenme oranı, pozitif sapma eşik değeri, ön plan eşik değeri ve ilk sapma eşik değeri belirlenmelidir. Ardından her bir görüntüdeki gauss dağılımları için eldeki piksel değerleri yardımıyla standart sapmalar ve ortalama gauss dağılım değerleri hesaplanır. Hesaplanan bu değerler standart sapma değerleri ile karşılaştırılır. Karşılaştırma sonucunda görülen farklar belirlenmiş olan limitler dâhilinde ise gauss bileşenleri güncellenmektedir. Bu durum sağlanmadıysa yeni gauss bileşenleri oluşturulur. Diğer bir aşamada ise kırmızı-yeşil-mavi standart sapma değerlerinin ortalaması alınır ve bileşen mertebeleri bulunur. Bulunan mertebeler sıralanır. Bu işlem görüntüdeki her bir gauss dağılımı için yapılır. Son aşama olarak ise ön plan pikselleri belirlenir. Eğer piksel değeri mertebelenmiş gauss değerinin 2,5 standart sapma değeri içinde ise ön plan pikselleri olarak seçilir. Şekil 3.2'de orijinal görüntü ve Şekil 3.3'de Gauss Karışımı uygulaması yapılmış bir görüntü örnek verilmiştir.



Şekil 3.2 Orjinal görüntü



Şekil 3.3 Gauss karışımı sonrası görüntü

3.2 Gürültü Çıkarımı

Ön plan çıkarımı ile hareketli nesnelerin siluetleri üretilir. Ancak, Şekil 3.3'de görüldüğü gibi, gürültü de ön plan segmentasyon çıktısının bir parçasıdır. Bu gürültüler diğer adımların performansını etkiler. Diğer sistem performanslarını iyileştirmek için, gürültü giderme çok önemli bir adımdır. Bu istenmeyen gürültüleri gidermek için bazı basit ancak etkili algoritmalar önerilen sistemde kullanılmıştır. Bu algoritmalar şunlardır:

- Morfolojik işlemler: aşınma ve genişleme
- Bağlı bileşen analizi ve alan filtreleme

3.2.1 Morfolojik İşlemler

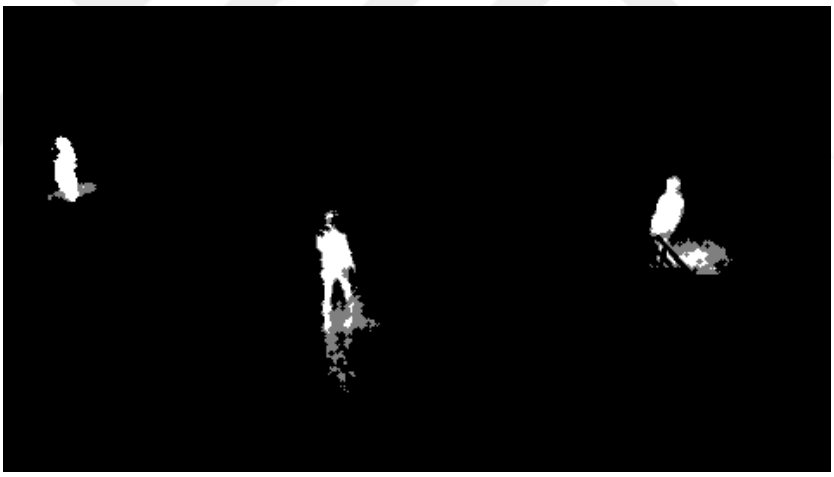
Morfolojik işlemler, nesnelerin parçalanması, gürültünün giderilmesi ve nesnelerin eksik kısımlarının tamamlanması için kullanılır [32]. Bu işlemlerde genellikle bir yapılandırma elemanı kullanılarak ikili görüntülere uygulanır. Yapılandırma elemanı, 0 ve 1'leri içeren ve genellikle 3x3 boyutunda seçilen bir matristir. Yapılandırma elemanının kökeni merkez pikseldedir. Görüntünün üzerine kaydırılır ve görüntünün her bir pikselinde elemanlar görüntüdeki öğelerle karşılaştırılır. Eğer iki set, set operatörü tarafından tanımlanan koşula uyuyorsa (örneğin, iki setin eleman çarpımı ile eleman belirli bir değeri aşarsa), yapılandırma elemanının kaynağının altındaki piksel önceden tanımlanmış bir değere ayarlanır (0 veya 1 için ikili görüntüler). Önerilen algoritmada, iki temel morfolojik işlem aşınma ve genişleme, ön plan segmentasyonu ile elde edilen ikili görüntüye uygulanır.

3.2.1.1 Aşınma İşlemi

Aşınma operatörü, önplan piksellerin bölge sınırlarını aşındırır. Bu amaçla kullanılan bir yapılandırma elemanı (3.8)'de gösterilmiştir. Giriş görüntüsündeki her bir ön plan pikseli, yapılandırma elemanının merkezi ile hizalanır.

$$SE_{aşınma} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{3.8}$$

Yapılandırma elemanında bir "1" değerine sahip olan her piksel için, görüntüdeki karşılık gelen piksel bir ön plan pikselidir, bu işlemden sonra giriş pikseli değişmez. Bununla birlikte, çevreleyen piksellerden herhangi biri arka plana aitse, giriş pikseli de arka plan değerine ayarlanır. Bu işlemin etkisi, Şekil 3.4'te gösterildiği gibi diğer beyaz piksellerle tamamen çevrilmemiş olan ön plan piksellerini kaldırmaktır. Sonuç olarak, gürültüye karşılık gelen küçük bölgeler elemine edilir, ön alan bölgeler küçülür ve bölgedeki delikler büyür.



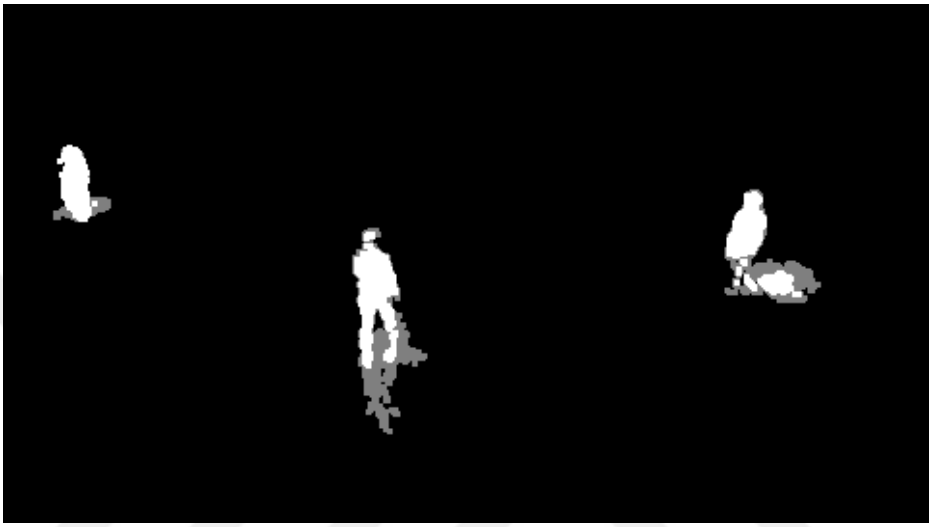
Şekil 3.4 Aşındırılmış görüntü

3.2.1.2 Genişleme İşlemi

Genişleme işlemi aşınma işleminin ikili çalışmasıdır. Genişleme işlemi için kullanılmakta olan bir örnek yapılandırma elemanı (3.9)'da verilmiştir. Yapılandırma elemanı, aşındırmada tanımlandığı şekilde ön plan pikselleri yerine arka plan pikselleri üzerinde çalışır. Genişleme işleminden sonra, ön plan bölgeleri büyürken, bölgelerdeki delikler Şekil 3.5'te gösterildiği gibi küçülmektedir.

$$SE_{genişleme} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad (3.9)$$

Gürültüyü çıkarırken, aşınma işlemi gevşek bağlı bölgeler arasındaki bağlantıların bağlantısını kesebilir. Bir bölgenin bağlılığı kaybolduğunda, artık ön plan olarak görülmemesi mümkündür. Ek olarak, güçlü bağlı bölgeler, aşınmadan etkilenir. Genişleme operasyonu, aşınmanın neden olduğu büzülmeyi telafi eder.

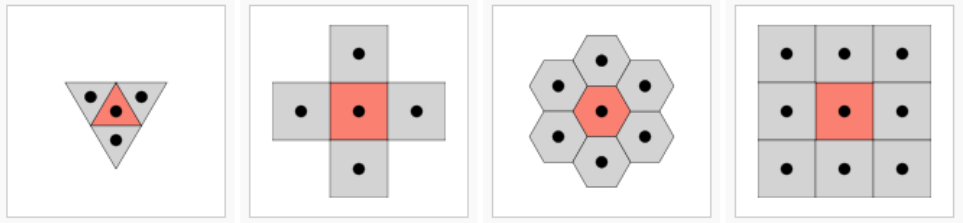


Şekil 3.5 Genişletilmiş görüntü

3.2.1.3 Bağlı Bileşenler Analizi

Bilgisayar görüsündeki en yaygın işlemlerden biri, bir görüntüdeki bağlı bileşenleri bulmaktır. Bağlı bileşenler bir aday bölge oluşturur, bu şekilde bir nesneyi temsil eder. Bağlı bir bileşen etiketleme algoritması, bir görüntüdeki tüm bağlı bileşenleri bulur ve aynı nesnedeki tüm noktalara benzersiz bir etiket atar.

1985'te Gibbons tarafından oluşturulan grafik teorisinde [34], eğer iki köşe birbirine bağlanmışsa ve bu köşelere bağlı bir köşe yoksa bu köşelerin birbirine bağlı olduğu söylenir [35]. Eğer bir nokta diğer üç noktaya bağlantılı ise 3, 4 noktaya bağlantılı ise 4, 6 noktaya bağlantılı ise 6 ve 8 noktaya bağlantılı ise 8 bağlılık olarak adlandırılır. Şekil 3.6'da bağlılık örnekleri görülmektedir.



Şekil 3.6 3, 4, 6 ve 8 bağlılık örnekleri

Hareket tespitinde, bazı yöntemler uygulandıktan ve gürültülü pikseller ortadan kaldırıldıktan sonra, bağlanan bileşenler görüntü üzerindeki nesnelerin kısa ve net bir görünümünü verir. Bağlı bileşenleri belirledikten sonra tüm nesneleri açıkça görebiliriz. Resimler üzerinde bir veya daha fazla bağlı bileşen bulunabilir ve her birini diğerlerinden ayırmak için benzersiz bir ad veya benzersiz bir numara vermemiz gerekebilir. Her bir bağlantılı bileşene benzersiz bir isim veya benzersiz bir numara verilmesi bağlı bileşen etiketleme olarak adlandırılır [36].

Grafik teorisinde, bir imge bölgesi bir görüntüdeki bir nokta veya nesnedir. Bir hareketi tespit etmek için, ilk işlem olarak, hareketini tespit etmek istediğimiz bloğun belirlenmesi gerekir. Tespit edilmek istenen imge bölgesi belirlendikten sonra, diğer imge bölgeleri elemine edilir. İmge bölgeleri tespit etmek için, yaygın olarak kullanılan yöntem bir pikselin komşu piksellerini analiz etmektir. Bu işlem bir görüntüdeki tüm piksellere uygulanır. 4 bağlantılı bağlı bileşenlerin tespit edilmesi gerekiyorsa, kuzey, batı, güney ve doğu pikselleri kontrol edilir, 8 bağlantılı bağlı bileşenlerin analiz edilmesi gerekirse, kuzeydoğu, kuzeybatı, güneybatı, güneybatı pikselleri kontrol edilir.

Analiz edilen piksellerden hiçbiri ikili görüntüde siyah piksel olarak etiketlenmemişse, bu nokta "bölge değeri" olarak etiketlenir. Yalnızca bir piksel siyah piksel olarak işaretlenmişse, bu piksel geçerli "bölgenin elemanı" olarak etiketlenir. Birden fazla piksel siyahsa ve bir bölgede etiketlenmediyse, tüm pikseller "bölge" olarak etiketlenir. Mevcut bölgedeki birden fazla piksel ve bunların bazıları veya hepsi diğer bölgelerde etiketlenmişse, bu çoklu piksellere "diğer bölgeler" denir.

Bağlı bileşenler etiketlendikten sonra, iki eksen için ek marj ekleyerek bileşenlerin sınırları kullanılarak ROI'ler oluşturulur. Bu ROI'ler, Bölüm 4'te eğitilmiş bir detektör ile yaya tespiti için kullanılır. Bağlı bileşenler analizinden sonra elde edilen ROI'ler Şekil 3.7'de gösterilmiştir.



Şekil 3.7 Bağlı bileşenler sonrası elde edilen ROI'ler

BÖLÜM 4

YAYA TESPİTİ VE TAKİBİ

Havalimanları, sokaklar ve tren istasyonları alış veriş merkezleri gibi gerçek dünya sahneleri karmaşıktır çünkü birçok insan yer alır, videolarda insanlar üst üste biner ve dağınık arka plan bulunur. Gerçek dünya sahneleri bu denli karmaşık olmasına rağmen, insan detektörleri yayaları bilgisayar görüsündeki son gelişmelere göre konumlandırabilmektedir. Hareketli videolarda her yaya tespit yöntemi için bir nesne tespiti mekanizması gereklidir. Nesne tespit performansı, nesne takibinin performansını doğrudan etkileyecektir.

Hareketli nesneler tespit edildikten ve nesne blokları elde edildikten sonra, bir sonraki adım, bir sınıfın örnekleri için ideal olarak benzer değerler üreten öznitelik çıkarımıdır. Öznitelik çıkarma işlemi, yaya tespit algoritmalarının önemli bir parçasıdır, çünkü sınıflandırma buna dayanır. Sınıflandırıcının eğitim ve test verisi üzerindeki nesneleri sınıflandırabilmesi için, öznitelik çıkarımı gereklidir. Öznitelikler, yeterli doğrulukta geniş bir veri kümesini tanımlar.

Yaya tespit algoritmasının son kısmı, hareketli nesnelerin yaya olup olmamasına göre sınıflandırılma aşamasıdır. Öznitelik çıkarma bölümünde çıkarılan öznitelikleri kullanarak nesneler arasında ayrım yapabilmek için sağlam bir sınıflandırıcı kullanılmalıdır.

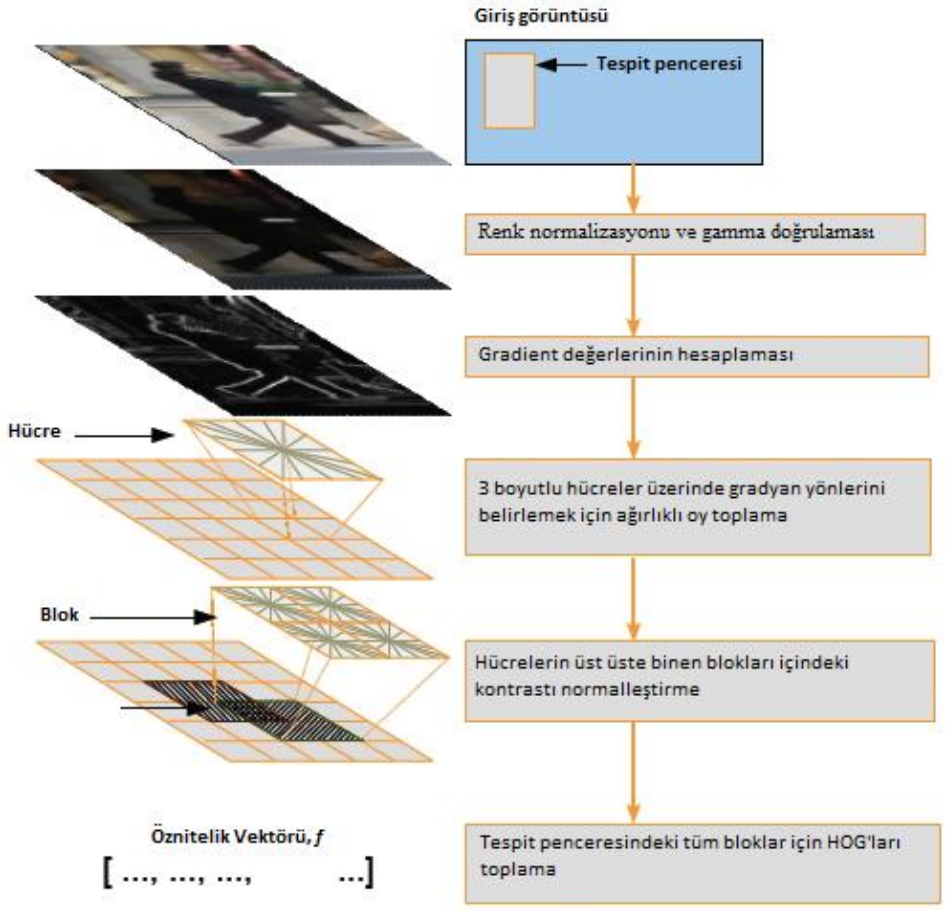
Bu bölümde, 2005 yılında Dalal ve Triggs tarafından önerilen HOG tanımlayıcısı [5], bu çalışmada kullanılan sınıflandırıcılardan olan SVM, kNN ve Adaboost sınıflandırıcıları ve hareketli nesne takibi için kullanılan kalman filtresi açıklanmıştır.

4.1 HOG Tanımlayıcı

Öznitelik çıkarma, insan tespiti yöntemine dayalı olarak yayaların tespit edilmesinde önemli bir adımdır. Bundan dolayı Dalal ve Triggs, 2005 yılında nesne algılama veya örüntü tanıma amacıyla bilgisayarla görüntü işlemede kullanılacak HOG öznitelik tanımlayıcılarını önermişlerdir. Bu tanımlayıcı, belirli bir nesneyle ilgili önemli miktarda bilgiyi veri olarak kodlamak için öznitelik vektörleri üretir. Ayrıca HOG öznitelikleri, eğitilmiş sınıflandırıcıyı kullanarak test görüntüsü üzerinde tahminde bulunma becerisi sağlamak için çıkarılır. Bu teknik, dijital bir görüntünün lokalize kısımlarında gradyan yönelimi olaylarını sayma yeteneğine sahiptir. Dahası, eşit aralıklı hücrelerin yoğun bir ızgarasını hesaplamada diğer yaklaşımlardan farklıdır ve aynı zamanda örtüşen yerel kontrast normalleşmesini kullanarak doğruluğu arttırır.

HOG tanımlayıcısının altında yatan ana fikir, yerel bir görüntü içindeki nesne görünümü ve şeklinin, yoğunluk gradyanlarının veya kenar yönlerinin dağılımı ile açıklanabilmesidir. Bu tanımlayıcı, görüntünün (saptama penceresi) uygun şekilde boyutlandırılmış hücrelere bölünmesiyle ve her bir hücre içindeki piksel yönelimlerinin bir histogramı oluşturularak gerçekleştirilebilir. Bu histogramlar birleşerek daha sonra kullanılacak olunan tespit penceresinin öznitelik vektörü haline gelir. Daha iyi bir doğruluk elde etmek için, yerel histogramlar, bloklar boyunca bir yoğunluk ölçümü hesaplayarak örtüşen uzaysal blokları normalize eder. Normalizasyon sonuçları, aydınlatma ve gölgelenme değişikliklerine karşı değişmezliği artırır.

HOG öznitelik çıkarma aşamaları Şekil 4.1'te gösterilmiştir ve aşağıdaki bölümlerde şekilde bahsedilen akış kısaca açıklanmıştır:



Şekil 4.1 HOG öznitelik çıkarma aşamaları

4.1.1 Normalizasyon ve Gama Doğrulama

HOG öznitelikleri çıkarılırken ilk olarak isteğe bağlı olarak normalizasyon ve gama doğrulaması işlemleri gerçekleştirilir. Normalizasyon, piksel yoğunluğu değerlerinin aralığını değiştiren bir işlemdir. Normalizasyon yapmanın amacı, videodaki ışık, parlama gibi nedenlerle oluşan problemleri aza indirmektir. Bundan dolayı normalizasyon işlemi zayıf kontrasta sahip fotoğrafları içerir. Normalizasyon bazen kontrast germe veya histogram eşitleme olarak adlandırılır [37]. Normalizasyon işlemi renkli veya gri seviye görüntülere uygulanabilir. Görüntüde renk bilgisi olduğunda her zaman daha çok yardımcı olur ve ek bilgiler sağlar. Örneğin insan detektörü için renkli görüntülerde iyi sonuçlar verirken, gri seviyeli görüntülerde performans düşmektedir [5]. Renkli görüntülerde normalizasyon işleminde RGB (Red, Green, Blue) ve LAB renk uzayları eşitlenir.

Renk normalizasyon işlemi, matematiksel hesaplama ve histogram eşitleme işlemleri olmak üzere iki şekilde yapılmaktadır. Matematiksel hesaplama işleminde video karesinde yer alan kırmızı, yeşil, mavi renk değerleri, videodaki toplam kırmızı, yeşil, mavi değerlerinin oranı ile hesaplanır (R = R/(R+G+B), G = G/(R+G+B), B = B/(R+G+B)). Histogram eşitleme işleminde ise, histogramın yoğunluk dağılımını değiştirerek bir görüntünün kontrastını ayarlamak için görüntüler işlenir. Bu tekniğin amacı, görüntü ile ilişkili kümülatif olasılık fonksiyonuna doğrusal bir eğilim kazandırmaktır. Böylece renk tonları değerleri resme yayılmaktadır. Şekil 4.2'de herhangi bir işleme sokulmamış görüntü ve normanlizasyon uygulanmış görüntü yer almaktadır.



Şekil 4.2 Orjinal ve normalize olmuş görüntü örnekleri

Her bir renk kanalı için karekök işlevi veya log fonksiyonu kullanılarak gama doğrulaması uygulanır. Bu doğrulama, yerel gölgeleme ve aydınlatma varyasyonlarının etkilerinin azaltılmasını sağlar. Gama doğrulaması formülü (4.1)'de verilmiştir:

$$V_{out} = A * V_{\text{in}}^{\gamma} \quad (4.1)$$

(4.1) eşitliğinde A sabit bir skalar değerdir, Vin ve Vout girdi ve çıktı matrislerinin negatif olmayan gerçek değerleridir [38].



(a) (b)

Şekil 4.3 Gama doğrulaması uygulanmış görüntü (a) Orijinal görüntü (b) ϒ<1 iken yapılan gama doğrulaması sonucu

4.1.2 Yön Vektörlerini Hesaplama

HOG öznitelikleri çıkarılırken isteğe bağlı olarak yapılacak olan normalizasyon ve gama doğrulaması işlemlerinin ardından ilk olarak yön vektörlerinin hesaplanması gerekmektedir. Yön vektörleri hesaplanırken her bir video karesinde yatay ve dikey yönlerde ayrık türev maskeleri kullanılarak konvolüsyon alınır.

Yön vektörleri hesaplaması için, Dalal ve Triggs, ortalanmamış [-1, 1], ortalanmış [-1, 0, 1], kübik düzeltmeli [1, -8, 0, 8, -1], 3 × 3 Sobel veya 2 × 2 diyagonal maske gibi birkaç türev maskesini test etmiştir. Basit 1-D merkezli maskeler kullanıldığında en iyi sonucun elde edildiği görülmüştür. En iyi sonuç alınması için, görüntünün renklerini yatay (x) ve dikey (y) yönlerde hesaplamak amacıyla Çizelge 4.1 de gösterilen türev maskeleri kullanılmaktadır. Böylece görüntünün renginin filtrelenmesi gerçekleşir.

Çizelge 4.1 Yön vektörleri türev maskesi

Türev Maskesi	
X Düzlemi Türev Maskesi	**Y Düzlemi Türev Maskesi**
-1 0 1	-1 0 1

Yön vektörleri hesaplanırken belirtilen koordinat (x,y) için, genlik (m) ve yön (θ) hesaplanır. Genlik ve yön hesaplamak için ilk olarak yatay ve dikey yön vektörleri hesaplanmalıdır. (4.2) eşitliğinde yatay yön vektörü ve (4.3) eşitliğinde ise dikey yön

vektörü hesaplanmıştır. Elde edilen sonuçlar ile (4.4)'te genlik, (4.5)'te yön hesaplaması yapılmıştır.

$$I_x = I(x+1, y) - I(x-1, y) \tag{4.2}$$

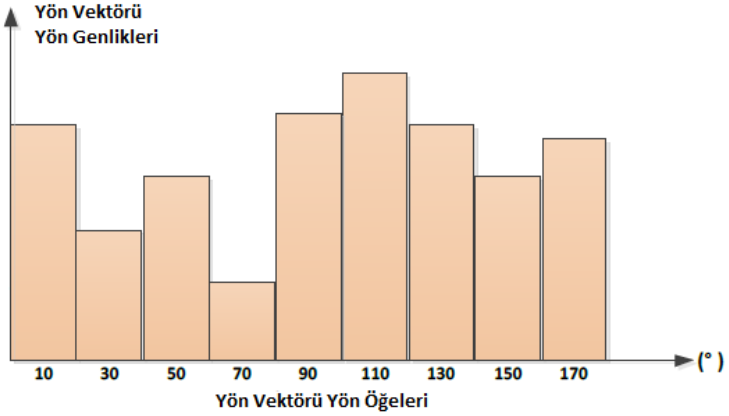
$$I_y = I(x, y+1) - I(x, y-1) \tag{4.3}$$

$$Genlik: m = \sqrt{I_x^2 + I_y^2} \tag{4.4}$$

$$Yön: \theta = \tan^{-1}\frac{I_y}{I_x} \tag{4.5}$$

4.1.3 Yön Vektörü Histogramı Oluşturma

Üçüncü aşamada, yön vektörleri ölçümünde elde edilen genlik ve yön bilgilerine dayanarak yön vektörü histogramı oluşturulur. Çalışılan görüntü birkaç piksel karelik alanlara ayrılarak incelenir. Bu alanlar hücre olarak adlandırılır. Hücrelerin şekli dikdörtgen veya dairesel olabilir. Hücreler içerisindeki her piksel, bir kenar yönlendirme histogram kanalı için ağırlıklı oy kullanır. Eğer yön vektörleri "işaretsiz" ise, yönlendirme kutuları 0 - 180 derece arasında, "işaretli" ise, yönlendirme kutuları 0 - 360 derece arasında eşit aralıklarla yerleştirilir. Dalal ve Triggs'in yaptığı çalışmalarda işaretsiz yön vektörleri kullanılarak, 20 derecelik aralıklardan oluşan 9 histogram kanalı ile en iyi performansa sahip sonuç alınmıştır [5]. Şekil 4.4'te yön vektörlerinin histogramlarının dağılımını gösteren örnek bir grafik oluşturulmuştur.

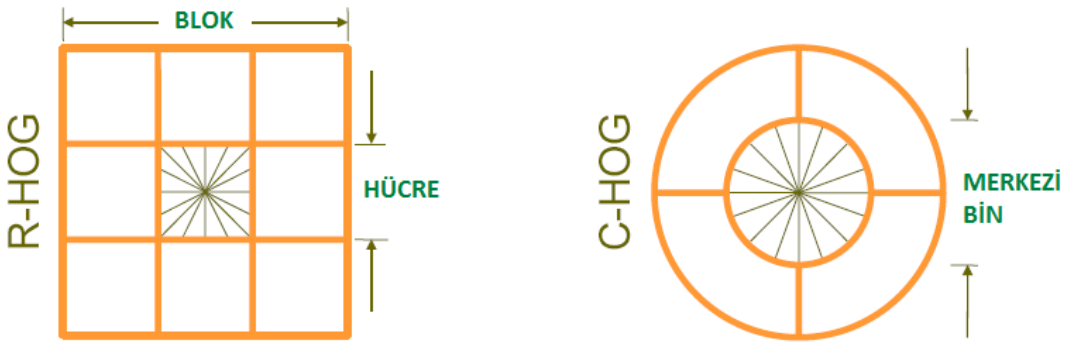


Şekil 4.4 Yön vektörü histogram dağılımı

4.1.4 Tanımlayıcı Bloklar

Ön plan ve arka plan kontrastındaki geniş çaplı değişimlerden dolayı, yön vektörlerinin lokal olarak normalizasyonu iyi bir performans için önemlidir. Farklı normalizasyon şemalarının çoğu, hücrelerin büyük bağlantılı bloklara gruplandırılmasına dayanır ve her blok ayrı ayrı normalize edilir. HOG tanımlayıcısının son hali, tüm blok bölgelerinden gelen normalleştirilmiş hücrelerin birleşiminden oluşan bir vektördür. Blokların üst üste gelmesi, nihai HOG tanımlayıcısına önemli bir katkı sağlar. Dalal ve Triggs, üst üste binen blokların, HOG tanımlayıcı performansını yaklaşık % 4 oranında verimli bir şekilde artırdıklarını açıklamaktadır.

R-HOG (Dikdörtgen HOG) ve C-HOG (Dairesel HOG) olmak üzere iki tür bloklama geometrisi bulunmaktadır. Şekil 4.5'te R-HOG ve C-HOG bloklarının şekilsel hali gösterilmiştir.



Şekil 4.5 R-HOG ve C-HOG şekli

R-HOG blokları, SIFT tanımlayıcılarına çok benzerdir. Ancak bu bloklar tek bir boyutta hesaplanmaktadır ve hesaplama sırasında yön düzeltmesi olmadığından SIFT'ten ayrılmaktadır [39]. R-HOG hücrelerin karesel sıralanmasıdır. Bu bloklama geometrisi 3 parametreye sahiptir. Bu parametreler blok başına hücre sayısı, hücre başına piksel sayısı, hücre başına kanal sayısıdır.

C-HOG blokları şekil bağlamlarına benzemektedir. Şekil bağlamları tek bir yönelimden bağımsız kenar varlığı sayımını kullanmasıyla C-HOG bloklarından ayrılmaktadır. C-HOG bloklarındaki her hücre yön vektörü ağırlıklı yönelimlerinin yığınını içerir [39]. C-HOG hücrelerin dairesel sıralanmasıdır. Bu bloklama geometrisinin ise dört parametresi

bulunur. Bu parametreler açısal ve radyal öğelerin sayısı, merkezi öğenin yarıçapı ve ek radyal öğenin yarıçapı için genişleme faktörü'dür.

4.1.5 Blok Normalleştirmesi

Blok normalleştirilmesi her hücredeki piksellerin ortalaması alınarak belirli sayıda hücrelerden oluşan bir bloğa uygulanır. Bu işlem ile ışık yoğunluğu, arka plan, gölgeleme gibi faktörlerden kaynaklanan problemlerin en aza indirilmesi amaçlanır. Bu işlem sonucunda elde edilen tanımlayıcı, aydınlatma ve gölgelemeye karşı tolerans kazanır. Dalal ve Triggs çalışmalarında blok normalizasyon için dört farklı normalizasyon denklemi üzerinde denemeler yapmışlardır. Bunlar L1-norm ve L1-karekök L2-norm, L2-hys denklemleri'dir. Bu denklemler (4.6), (4.7), (4.8), (4.9) eşitlikleri ile gösterilmektedir.

$$L1 - norm : f = \frac{v}{\sqrt{||v||_1 + \epsilon^2}} \quad (4.6)$$

$$L1 - karekök: f = \sqrt{\frac{v}{||v||_1 + \epsilon^2}} \quad (4.7)$$

$$L2 - norm: f = \frac{v}{\sqrt{||v||_2^2 + \epsilon^2}} \quad (4.8)$$

$$L2 - hys: v > 0.2\ v = 0.2;\ f = \frac{v}{\sqrt{||v||_2^2 + \epsilon^2}} \quad (4.9)$$

Bu denklemlerde kullanılan υ değeri, normalleştirilmemiş bir tanımlayıcı vektördür ve $\|\upsilon\|_r$ terimi ise v teriminin r sayısı normalizasyonu (r-norm) demektir. Burada ε ise küçük bir sabit değerdir.

HOG yöntemi kullanılarak yapılan çalışmada denklemlerinin denenmesi sonucunda L2-hys, L2-norm ve L1-karekök denklemleri birbirleriyle benzer sonuçlar verdiği gözlemlenmiştir. Ancak L1-norm denkleminde diğer denklemlere göre daha az performanslı sonuçlar gözlemlenmiştir [5]. L1-norm denklemi daha kötü sonuç verse de hiç normalleştirilmemiş veriden daha iyi sonuç vermektedir.

4.2 Sınıflandırma

Yayaların tespitini gerçekleştirmek için, yayalarla yaya olmayan kalıpları birbirinden ayırmayı öğrenen eğitilebilir bir sınıflandırıcıya ihtiyacımız vardır. Bundan dolayı bu adımda, farklı sınıflandırıcılar önceden çıkarılmış yaya öznitelikleri tarafından eğitilir ve test edilir. Sınıflandırma aşamasında öncelikle örnek bir veri seti kullanılarak bir eğitim seti oluşturulur ve bu set ilgili sınıflandırıcı ile eğitilir. Test aşamasında videoda yer alan her bir pencerenin öznitelikleri çıkarılarak ilgili sınıflandırıcı ile test edilir. Böylece ilgili pencerede yer alan yayalar sınıflandırılmış olur. Çalışmamızda SVM, kNN ve Adaboost sınıflandırıcıları olmak üzere üç farklı sınıflandırıcı kullanılmaktadır.

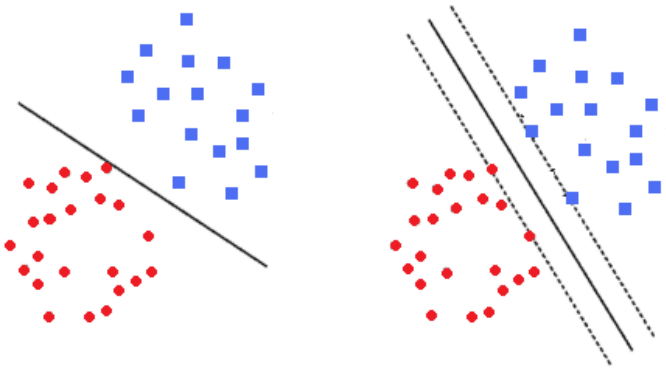
4.2.1 Destek Vektör Makinesi (SVM) Sınıflandırıcı

SVM, 1995 yılında Cortes ve Vapnik tarafından geliştirilmiştir [40]. Bu sınıflandırıcı, yaygın olarak genel desen tanıma uygulamaları için popüler ve güçlü bir denetimli öğrenme aracı olarak kullanılmaktadır. Ayrıca, farklı sınıflandırma problemlerine, veri ayrıştırılmasına ve yoğunluk tahminine uygulayarak makine öğrenimi ve diğer birçok alan üzerinde ümit vaat eden ve mükemmel performans sağlamaktadır [41].

SVM sınıflandırıcısının, doğru ve sağlam sınıflandırma algoritmalardan biri olmasını sağlayan birçok avantajı vardır:

- Eğitim setindeki az sayıda görüntüde bile yüksek performans sağlar.
- Görüntü boyutlarına bağlı değildir.
- Test setinin sayısı yüksek olsa bile veri sınıflandırması için daha iyi genellemeye yol açan deneysel ve yapılsal riski minimize etme yeteneği bulunur.

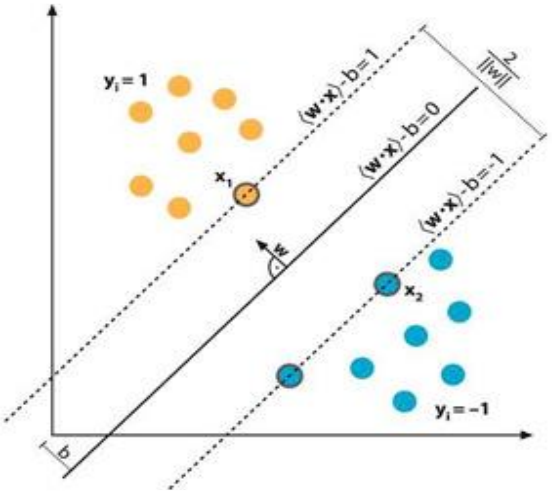
SVM, özellik alanında lineer karar hiper düzlem kullanarak sınıflandırma yapar [42]. Bu sınıflandırıcı destek vektörleri olarak adlandırılan eğitim setinin en bilgilendirici noktalarına dayanarak iki sınıfın örnekleri arasında bir karar hiper düzlemi oluşturur. SVM'nin amacı, en büyük marjı olan iki sınıfın örnekleri arasında en uygun hiper düzlemi bulmaktır (Şekil 4.6).



(a) (b)

Şekil 4.6 Karar bölgesi örnekleri (a) Sıradan sınıflandırıcı (b) SVM

Destek vektör makineleri, giriş vektörlerini, maksimum ayırıcı hiper düzlemin tasarlandığı daha yüksek bir ölçü alanına yönlendirir. Hiper düzlemin her iki tarafında, verileri ayıran iki paralel hiper seviye tasarlanmıştır. Ayırıcı hiper seviye, iki paralel hiper seviye arasındaki mesafeyi maksimize eden hiper seviyedir. Yan kenarlar veya paralel hiper seviyeler arasındaki mesafe büyük ölçüde olduğunda sınıflandırıcının genelleme hatasının daha iyi olduğu kabul edilir. İki kategoriden sadece birine ait her veri noktası bir P boyutlu vektörle gösterilir. Bu doğrusal sınıflandırıcıların tipik bir şeklidir. Bu özelliği yerine getirebilecek birçok doğrusal sınıflandırıcı vardır. İki kategori arasında maksimum marj elde edebilmek için hiper düzlemden bir sonraki veri noktasına olan mesafenin maksimize edilmesi gerekir.



Şekil 4.7 SVM için maksimum marjlı hiper düzlemler

SVM'nin için maksimum marjlı hiper düzlemler gösterimi, Şekil 4.7'de gösterilmektedir. Veri noktaları $\{(x_{1,}, y_1), (x_{2,}, y_2), \ldots, (x_{n,}, y_n)\}$ olarak gösterilir. Burada x_i eğitim kümesindeki örnekler için p-boyutlu öznitelikler vektörü, y_i ise -1 veya 1 değerlerini alan, sınıfın etiketlerini göstermek üzere pozitif ve negatif etiketli örnekleri birbirinden ayırabilen değerdir. w, hiper düzleme dik olan normal bir vektördür. Herhangi bir hiper düzlem, x tatmin edici noktaları kümesi olarak yazılabilir:

$$w \,.\, x - b = 0 \tag{4.10}$$

Maksimum marja bakıldığında, destek vektörleriyle ilgilenilir. Paralel hiper düzlemler, her iki sınıftaki bu destek vektörlerine en yakındır. (4.11) ve (4.12) bu paralel hiper düzlemleri göstermektedir.

$$w \,.\, x - b = 1 \tag{4.11}$$

$$w \,.\, x - b = -1 \tag{4.12}$$

Eğitim verileri doğrusal olarak ayrıldığında, bu hiper düzlemler seçilebilir. Bu düzlemler arasında herhangi bir nokta bulunmaz ve bu düzlemler arasındaki mesafeler en üst düzeye çıkarılmaya çalışılır. Geometri kullanarak, | w | asgariye indirilir ve hiper düzlemler arasındaki mesafe 2 / | w | olarak bulunur. Bu mesafe sınır olarak adlandırılır. Maksimum sınırı bularak optimum hiper düzlemi elde etmek için $\min[\frac{1}{2}||w||^2]$ ifadesi (4.13)'e bağlı olarak hesaplanır.

$$y_i(w \,.\, x_i + b) - 1 \geq 0 \,, y_i \in \{1, -1\} \tag{4.13}$$

Lagrange denklemleri kullanılarak optimizasyon problemleri çözülmektedir. Bu problem Lagrange denklemleri yardımıyla (4.14)'teki gibi yazılmaktadır. Bu denklemde yer alan "α_i" değeri pozitif Lagrange çarpanlarını göstermektedir.

$$L_p = \frac{1}{2}||w||_1^2 - \sum_{i=1}^{k} \alpha_i \, y_i(w \,.\, x_i + b) + \sum_{i=1}^{k} \alpha_i \tag{4.14}$$

(4.14) oldukça karmaşık bir çözüme sahiptir. Bundan dolayı KKT (Karush-Kuhn- Tucker) koşulları kullanılarak sadeleştirilmektedir ve yeni bir eşitlik elde edilmektedir. Elde edilen eşitlik (4.15)'de görülmektedir. Bu problemin çözümü için gereken KKT koşulları (4.16) ve (4.17)'de verilmiştir.

$$\frac{\partial L_p}{\partial w} = 0 \rightarrow \; w = \; \sum_{i=1} \alpha_i \, y_i x_i \tag{4.15}$$

$$\frac{\partial L_p}{\partial b} = 0 \rightarrow \sum_{i=1} \alpha_i y_i = 0 \quad (4.16)$$

$$L_d = \sum_{i=1} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i x_j \quad \alpha_i > 0, \forall_i \quad (4.17)$$

Sistemin eğitilmesi için kullanılan her bir örnek için bir adet Lagrange çarpanı bulunur. Lagrange çarpanlarının çoğunun değeri sıfırdır, geriye kalanlar ise pozitif α_i değerli x_i vektörleri ile destek vektörlerini oluşturur.

4.2.2 K-En Yakın Komşu (kNN) Sınıflandırıcı

Örnek tabanlı kNN yöntemi fikri ilk olarak Aha ve Kibler tarafından 1991 yılında sunulmuştur [43]. K-en yakın komşular sınıflandırıcısı, tüm makine öğrenimi algoritmalarının en basitidir ve uygulanması en kolay olanıdır. kNN, makine öğrenimi uygulamalarında, regresyon ve örüntü tanımada kullanılır [44]. En yakın komşunun sınıflandırma yaklaşımı, yeni örneklerin her birinin, bir mesafe ölçüsünün kullanılmasıyla mevcut örneklerle karşılaştırılarak, en yakın mevcut örnek sınıfına atamak için kullanılır. Tipik olarak, en yakın bir komşudan yararlanılır ve yeni örneğe en yakın K komşularının veya sınıfın sayısal değerlere sahip olması halinde uzaklık ağırlıklı ortalaması verilir. K pozitif bir tamsayıdır. Eğer k bir seçilirse, nesne en yakın komşularının sınıfına aittir. kNN sınıflandırıcının performansı k değerine çok duyarlı olmasına ve sonuçların herhangi bir değişiklikten etkilenmesine rağmen, kNN sınıflandırıcı birçok sınıflandırma probleminde yüksek performansla yaygın olarak kullanılmaktadır[45], [46]. Bağlı oylarla ilgili zorlukları ortadan kaldırdığı için k değerini tek bir sayı olarak seçmek faydalı olacaktır. kNN algoritması nesneleri üç adımda sınıflandırmaktadır [45]. Bu adımlar;

- Tüm eğitim vektörleri ve test vektörü arasındaki mesafe hesaplanır.
- En yakın K vektörleri seçilir.
- En yakın vektörlerin mesafelerinin ortalaması hesaplanır.

Bu algoritmada, Manhattan, Euclidean ve Minkowski mesafe fonksiyonları kullanılarak eğitim ve test vektörleri arasındaki mesafeler hesaplanır.

Manhattan mesafe, 90 derece açılarda eksenler boyunca ölçülen noktaların çiftleri arasındaki mesafedir. Bu mesafe (4.18) eşitliği ile hesaplanmaktadır.

$$d(x,y) = \sum_{i=0}^{n-1} |x_i - y_i| \tag{4.18}$$

Öklid mesafesi, Öklid uzayındaki birkaç nokta arasındaki düz çizgi mesafesi olarak tanımlanmaktadır. Öklid mesafesi (4.19) eşitliği ile hesaplanmaktadır.

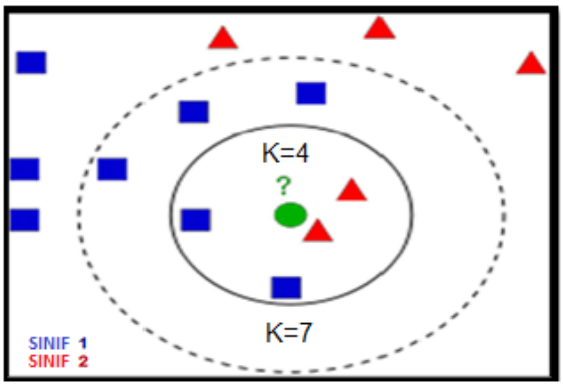
$$d(x,y) = \sqrt{\sum_{i=0}^{n-1} |x_i - y_i|^2} \tag{4.19}$$

Minkowski mesafesi, normalize edilmiş vektör uzayındaki bir kaç nokta arasındaki mesafe olarak tanımlanır. Öklid mesafesi (4.20) eşitliği ile hesaplanmaktadır.

$$d(x,y) = (\sum_{i=0}^{n-1} |x_i - y_i|^p)^{\frac{1}{p}} \tag{4.20}$$

kNN algoritmasında en yaygın olarak kullanılan mesafe hesaplama yöntemi öklid mesafesidir. Bu fonksiyonun kullanılması özelliklerin normalleştirildiği ve eşit derecede önemli olduğu varsayımını verir. Önemli özelliklerin belirlenmesi öğrenme sürecinde en önemli konulardan biridir.

kNN'de; göz önüne alınan çıktı, bir sınıfın üyesidir. Yeni bir nesne, komşularının oy sayısına bağlı olarak sınıflandırılır. En yakın komşularda; eğer K = 1 ise, nesnenin en yakın komşuların sınıfına ait olduğu anlamına gelir. Bununla birlikte, kNN'de K'yı seçmenin özel bir yolu yoktur, sadece en iyisi seçmeye çalışılır. kNN, girdiler ve k değeri verildiğinde bir girişi eğitmek ve daha sonra bir girişi sınıflandırmak için doğrudan eğitim setini kullanır.



Şekil 4.8 K-en yakın komşu mekanizması

Şekil 4.8 de kNN mekanizması gösterilmektedir. Burada iki sınıf bulunur. Mavi kareler birinci sınıfı ve kırmızı üçgenler ikinci sınıfı temsil eder. Bu sınıflar bir özellik uzayında temsil edilir. Öznitelikler, tüm verileri içeren iki boyutlu örnek bir alan olarak düşünülebilir. Dolayısıyla, bu veriler, x ve y iki koordinatı olan iki özelliğe sahiptir ve bu veriler bu alanda temsil edilebilir. Örneğin, yeşil bir daire tarafından temsil edilen veriler yeni bir veridir ve bu veriler kırmızı veya mavi kümeye eklenmek istenmektedir. Bu aşamaya sınıflandırma denir. Sınıflandırma yapmak için en yakın komşuyu kontrol etme yöntemi vardır. Şekilde, kırmızı üçgen yeni veriler (yeşil daire) için en yakın komşudur, bu nedenle; yeni veri, sınıf 2'ye eklenmiştir. Bu örnekte, sınıflandırma süreci en yakın komşuya dayanmaktadır. Bundan dolayı, bu teknik en yakın komşu olarak adlandırılır. Bu örnekte eğer K = 4 ise, bu 4 en yakın nesne demektir. Sınıflandırılacak veriye yakın, iki kırmızı nesne ve iki mavi nesne vardır, bu durumda yeni veriler kırmızı nesnelere daha yakın olduğundan kırmızı nesne sınıfına (2. Sınıf) eklenir. Eğer K = 7 ise, 2 kırmızı nesne ve 5 mavi nesne olduğu anlamına gelir, böylece yeni veriler mavi nesnelere eklenir (1. Sınıf).

4.2.3 Adaboost

Adaboost algoritması, Freund ve Schapire tarafından 1995 yılında önerilen bir sınıflandırma algoritmasıdır [47]. Adaboost, bir dizi zayıf öğreniciyi alan ve güçlü bir sınıflandırıcı üretmek için bir dizi yinelemeyle doğrusal bir kombinasyon oluşturan bir yükseltme algoritmasıdır. Zayıf öğrenen, tüm sorunu tek başına çözecek yetersiz hipotezler veren bir sınıflandırıcıdır. Bu zayıf öğreniciler genellikle, girdi ve bir eşik arasındaki bir karşılaştırmanın sonucunu değerlendirerek çıktıya karar veren eşik sınıflandırıcılar olarak seçilir. Bu eşik sınıflandırıcıları 2001 yılında Viola tarafından önerilmiştir. Böyle bir eşik sınıflandırıcı, $h_j(x)$, (4.21)'de verilmiştir [48].

$$h_j(x_j) = \begin{cases} +1, & p_j x_j < p_j \theta_j \\ -1, & diğerleri \end{cases} \tag{4.21}$$

Bu denklemde x_j öznitelik, θ_j eşik, p_j eşitsizliğin yönünü belirleyen parite ve K 1 ≤ j ≤ K durumunu sağlayan özniteliklerin sayısıdır. Her zayıf öğrenici sadece bir özniteliği inceleyerek kararını verir, böylece her sınıflandırıcı bir özniteliğe karşılık gelir. Zayıf

öğrenicinin eğitimi, j, (4.22)'de verilmiştir. Bu denklem ile t kadar iterasyonda sınıflandırma hatasını en aza indiren $\theta_j$ve p_j değerleri belirlenir.

$$(\theta_j, p_j) = argmin\{\in_{t,j}\} \quad (4.22)$$

Bu işlem basitçe min (x) ≤ θ_j ≤ max (x) ve p_j = {+ 1, −1} aralıklarında arama yaparak elde edilebilir. $\in_{t,j}$'nin tanımı (4.23)'te verilmiştir, burada y_i istenen çıktı etiketidir.

$$\in_{t,j} = \sum_{i:\, h_i(x_i) \neq y_i} D_t(i) \quad (4.23)$$

(4.23) eşitliğinde, $D_t(i)$, t ninci iterasyonda eğitim örnekleri üzerinde dağılım işlevidir. Çizelge 4.2'de verilen Adaboost'un tam algoritmasında görülebileceği gibi, bu dağılım, yanlış sınıflandırılmış örnekleri vurgulamak için kullanılmıştır ve algoritmanın eğitim setindeki zor örneklere odaklanmasını zorlamaktadır. $D_t(i)$ tekdüze olacak şekilde başlatılır ve her yinelemede, gerçek sınıflandırılmış örneklerin değerleri azaltılacak ve yanlış sınıflandırılmış örneklerin değerleri artırılacak şekilde güncellenir.

Çizelge 4.2 Adaboost algoritması akışı[49]

Verilen: $(x_1, y_1), \ldots\ldots, (x_m, y_m);\ x_i \in X, y_i \in \{-1,1\}$

Başlangıç ağırlıkları: $D_1(i) = \frac{1}{m}$

T iterasyon sayısı olmak üzere, her t = 1,.,T için :

1. $h_t = argmin\ \varepsilon_j;\ \varepsilon_j = \sum_{i=1}^{m} D_t(i)\, I[h_i(x_i) \neq y_i],\ h_j \in H$
2. Eğer $\varepsilon_t \geq 1/2$ dur,
3. $a_t = \frac{1}{2}\log\left(\frac{1+\varepsilon_t}{\varepsilon_t}\right)$
4. D güncellenir: $D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t y_t h_t(x_i)}{Z_t}$

Sınıflandırıcı bulunur:

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$$

İlk adım olarak sınıflandırıcıların başlangıç ağırlığı, $D_1(i)$ atanır. Daha sonra öğrenme süreci için basit bir döngü yapılır. Zayıf sınıflandırıcılar Adaboost'a verilir ve bir dizi turda çağrılır (t = 1, ..., T). Her turda, eğitim setindeki ağırlıklı bir hatayı en aza indiren zayıf bir sınıflandırıcı (Adım 1) seçilir. Bu hata 1/2 değerini aşarsa döngü sonlandırılır (Adım 2). T nin değeri hesaplanır (Adım 3) ve ağırlıklar üstel kurallara göre güncellenir (Aşama 4). Aşama 4'te, Z_t, D_{t+1}'nin bir dağılım olduğunu garantileyen bir normalleştirme faktörüdür. Son karar kuralı, katsayıları tarafından seçilen seçilmiş zayıf sınıflandırıcıların doğrusal bir kombinasyonudur. Sınıflandırıcı kararı doğrusal kombinasyonun işareti ile verilir [49].

Adaboost'un hızlı, basit ve programlaması kolay olmak gibi birçok avantajı vardır. Ayrıca, yineleme sayısı T dışında herhangi bir parametrenin ayarlanmasına ihtiyaç olmayan bir sınıflandırıcıdır. Adaboost, birçok zayıf öğreniciyi bir araya getirerek doğru bir tahmin kuralı üretmek için genel ve kanıtlanabilir bir yöntem sunar. Öte yandan, aşırı karmaşık veya çok zayıf hipotez ile yükletme başarısız olabilmektedir.

4.3 Hareketli Nesne Takibi

Hareketli nesne takibi, otomatik bir gözetim sistemi için önemli adımlardan biridir. Artan ihtiyaçlarla birlikte otomatik video analizi ile nesne izleme algoritmalarına büyük ilgi oluşmuştur. En basit haliyle, izleme, uzaktan ölçümlere dayanan hareketli bir nesnenin durumunun tahmini olarak tanımlanmaktadır [50]. Yapılan çalışmada izleme, mevcut ve önceki kareler arasındaki hareketli bölgelerin ilişkilerini bulmak olarak tanımlanabilmektedir.

4.3.1 Kalman Takibi

Kalman filtreleme, tahmin teorisinde popüler bir yaklaşımdır. Bu çalışmada, hareket modeliyle standart kalman filtresini kullanarak nesnenin konumu tahmin edilmektedir. Kalman filtresi, hedefleri izlemek için bağımsız bir araç olarak kullanılmış ve hareketli nesnenin yeri takip edilmiştir.

4.3.1.1 Kalman Filtresi

Kalman filtresi, karesel hatanın ortalamasını en aza indirecek şekilde, bir işlemin durumunu tahmin etmek için verimli bir hesaplama araçları sağlayan bir matematiksel denklem dizisidir [51]. Filtre, çeşitli yönlerden çok güçlüdür: geçmiş, şimdiki ve hatta gelecekteki durumların tahminlerini destekler ve modellenen sistemin kesin doğası bilinmediğinde bile bunu yapabilir.

Kalman filtresi, doğrusal stokastik fark denklemi tarafından yönetilen ayrık zaman kontrollü bir sürecin durum $x \in R^n$ değerini tahmin etmeye yönelik genel problemi ele almaktadır. Stokastik fark denklemi (4.24)'te verilmiştir [52].

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \tag{4.24}$$

bu denklem tarafından tanımlanan bir ölçüm ile;

$$z_k = Hx_k + v_k \tag{4.25}$$

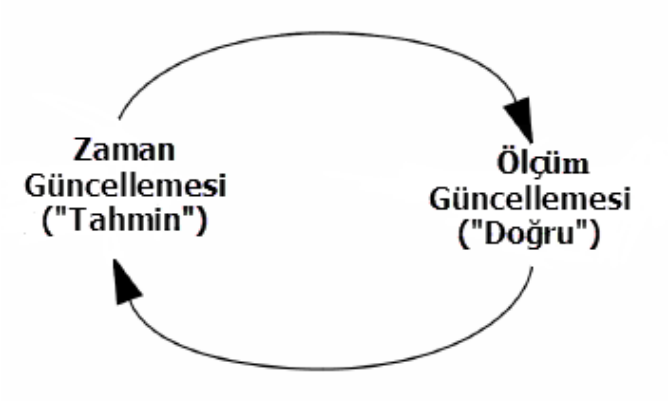
w_k ve v_k rasgele değişkenler sırasıyla süreç ve ölçüm gürültüsünü temsil eder. Birbirlerinden bağımsız, beyaz ve normal olasılık dağılımları ile kabul edilirler. Bu dağılımlar (4.26) ve (4.27)'de verilmektedir.

$$p(w) \sim N(0, Q) \tag{4.26}$$

$$p(v) \sim N(0, R) \tag{4.27}$$

Uygulamada, işlem gürültüsü kovaryans Q ve ölçüm gürültü kovaryans R matrisleri her zaman adımında veya ölçümde değişebilir, ancak bu çalışmada sabit oldukları varsayılmaktadır.

Bir işlemin sonraki durumu Şekil 4.9'da verilen şema ile Kalman filtresi tarafından tahmin edilir. Zaman güncellemesi, zaman içindeki durum tahminini öngörmektedir. Ölçüm güncellemesi ise, o andaki mevcut ölçüm ile tahmin edilen durumu öngörmektedir.



Şekil 4.9 Kalman filtre döngüsü

Kalman filtresinin bu adımları için, yani zaman güncellemesi ve ölçüm güncellemesi için, önceki (x_{k-1}) ve sonraki (x_k) durum tahmin hesaplama eşitlikleri bulunur [52].

Zaman güncellemesi (tahmin):

$$x_k^- = Ax_{k-1} + Bu_{k-1} \quad (4.28)$$

$$P_k^- = AP_{k-1}A^T + Q \quad (4.29)$$

Ölçüm güncellemesi (doğru):

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \quad (4.30)$$

$$x_k = x_k^- + K_k(z_k - Hx_k^-) \quad (4.31)$$

$$P_k = (I - K_k H)P_k^- \quad (4.32)$$

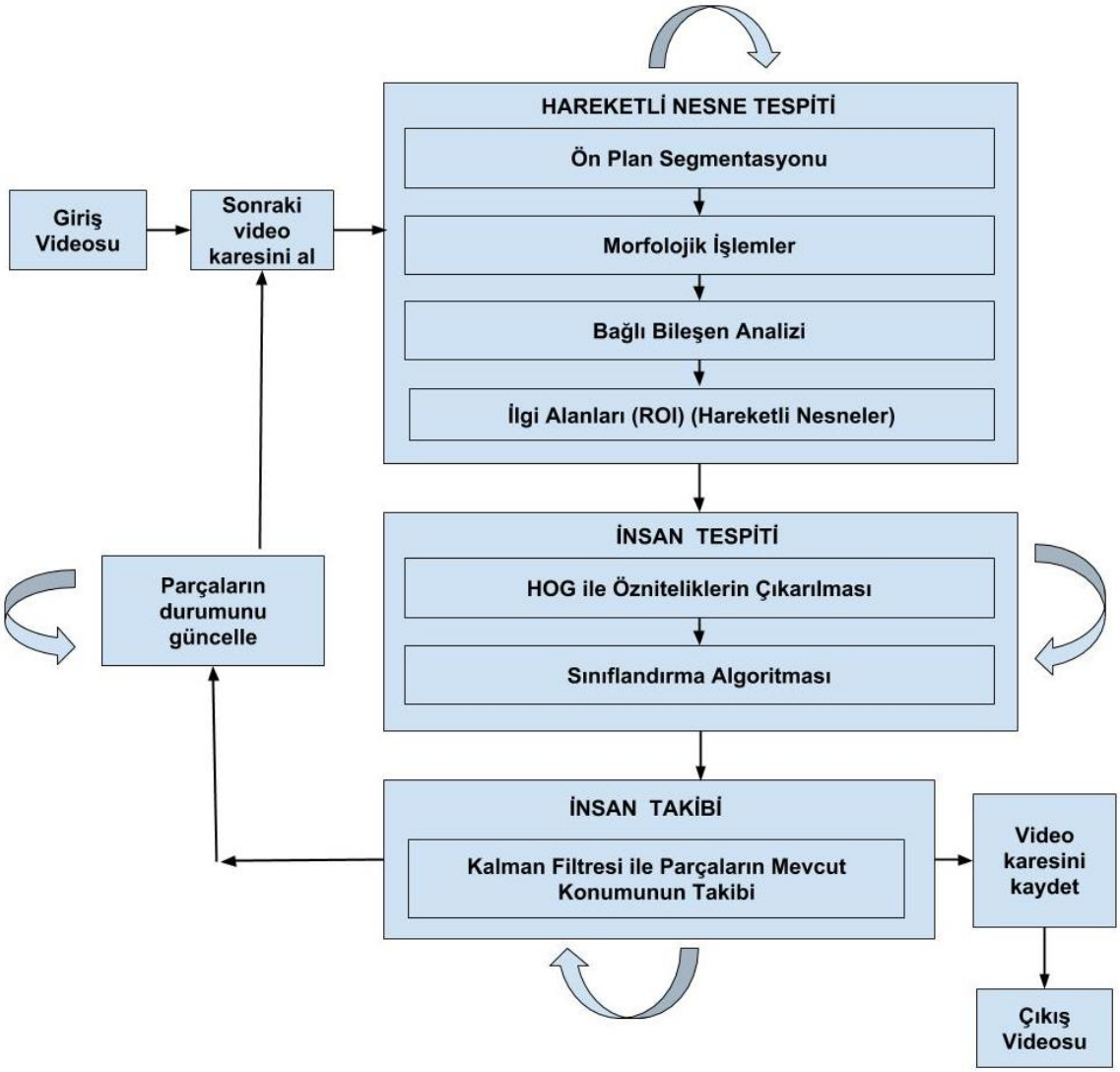
BÖLÜM 5

VİDEO İŞARETLERİNDE YAYA TAKİBİ UYGULAMASI

Bu bölümde MATLAB ile video işaretlerinde yaya takibi uygulaması yapılacaktır. Uygulama temel olarak üç aşamadan oluşmaktadır. Bu aşamalar hareketli nesne tespiti, yaya tespiti ve yaya takibi aşamalarıdır.

Genel mimari yapı, Şekil 5.1'de tasvir edilmiştir. Öncelikle, arka plan, gauss karışımı kullanılarak modellenmiştir ve ROI'ler, gölge kaldırmayı uyguladıktan sonra ön plan çıkarımı ile hesaplanmıştır. Bu işlem adımı ile hareketli nesneler tespit edilmiştir. Hareketli nesnelerin tespit edildiği bölgeler yaya tespiti için kullanılacak olan ROI'leri oluşturmuştur. Yaya tespiti eğitim ve tespit aşamaları olmak üzere iki aşamadan oluşmaktadır. Eğitim aşamasında veri setleri kullanılarak her üç sınıflandırıcı da eğitilmiştir. Veri seti olarak INRIA kullanılmıştır. Tespit aşamasında hareketli nesne tespiti aşamasında elde edilen ROI'ler kullanılmaktadır. Tespit edilen ROI'lerin HOG tanımlayıcısı kullanılarak öznitelikleri çıkarılır. Çıkarılan bu öznitelikler eğitim aşamasında eğitilen sınıflandırıcılara verilir. Bu uygulamada SVM, kNN ve Adaboost sınıflandırıcıları kullanılarak komite oylamasına dayalı bir yaya tespiti algoritması kullanılmıştır. Buna göre herhangi üç sınıflandırıcıdan iki tanesinin insan olarak tespit ettiği sonuçlar yaya olarak kabul edilmiştir. Yayaların tespitinin ardından takip aşamasına geçilmiştir. Bu aşamada tespit edilen tüm yayalar, durum uzayındaki noktalar tarafından temsil edilen hedefleri izlemek için kalman filtresi ile takip edilmektedir. Takip edilen yayaların izlediği rotalar çıkarılarak bir videoda kaydedilmiştir.

Uygulanan sistemin yinelenen bir süreci vardır. Giriş videosundan yeni bir kare getirildiğinde, hareketli nesne tespiti, yaya tespiti ve yaya takibi aşamaları sırasıyla gerçekleştirilmekte ve elde edilen sonuç bir başka videoda kaydedilmektedir. Bu işlem tüm kareler bitene kadar devam etmektedir.

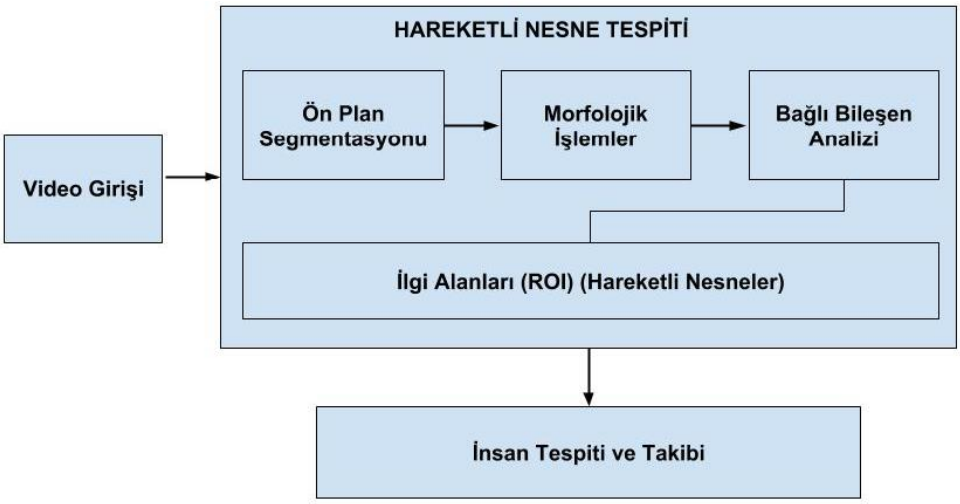


Şekil 5.1 Video işaretlerinde yaya takibi için genel mimari yapı

5.1 Hareketli Nesne Tespiti

Hareketli nesne tespiti aşaması yaya takibi için uygulanan ilk basamaktır. Yapılan çalışmada yeni bir video çerçevesi alındığında, ön plan nesneleri GK algoritması ile çıkarılmıştır. Ardından morfolojik işlemler uygulanarak görüntüde yer alan gürültü kaldırılmıştır. Sadece ilgili alanlara odaklanmak için bağlı bileşen analizi gerçekleştirilmiş ve ROI'ler tespit edilmiştir. Tespit edilen ROI'lerde hareketli bir nesne

bulunmaktadır. Hareket eden bu nesne her tür varlık olabilmektedir. Bu nesnenin yaya olup olmadığının anlamak için elde edilen ROI'ler yaya tespiti bloğuna verilmiştir. Hareketlili nesne tespiti bloğunun akış şeması Şekil 5.2 de görülmektedir.



Şekil 5.2 Hareketli nesne tespiti akış şeması

Bu çalışmada, ön plan çıkarımı için gauss karışım model yaklaşımı kullanılmıştır. Her bir piksel, onu temsil eden gauss dağılımının arka plan modeline uygunluğuna göre sınıflandırılmıştır. Uygulamada gauss dağılımı bileşenleri sayısı olarak 3, arka plan bileşenleri sayısı 40, öğrenme oranı 0,005 ve minimum arka plan oranı 0,7 olarak alınarak işlem gerçekleştirilmiştir. Gauss karışım model yaklaşımının detaylı anlatımı Bölüm 3'te yer almaktadır.

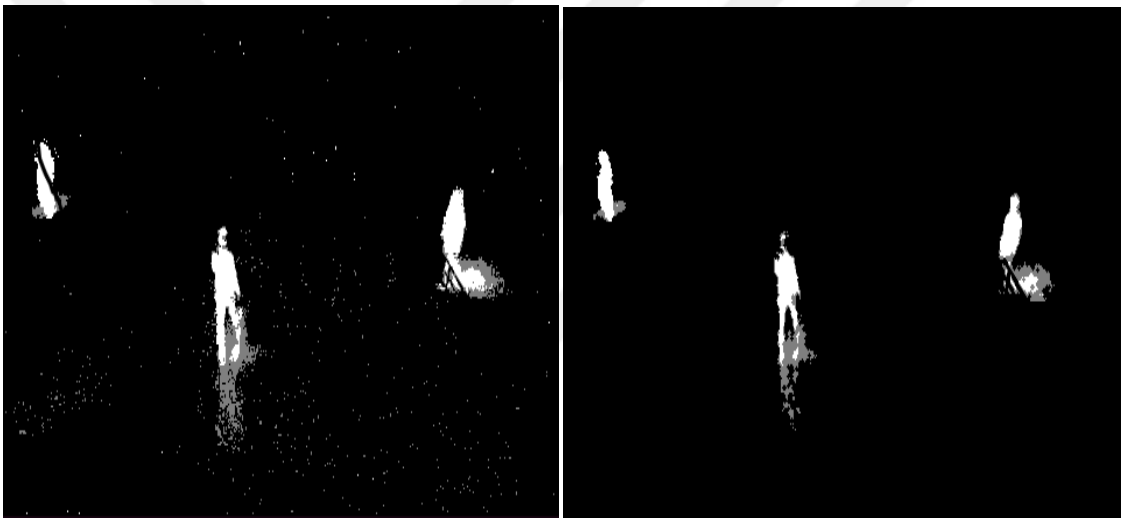


(a) (b)

Şekil 5.3 Gauss karışımı sonucu (a) Orjinal görüntü (b) Gauss karışımı sonrası sonuç

Ön plan segmentasyonu sonucunda görüntüdeki hareketli nesnelerin siluetleri üretilmiştir. Şekil 5.3'te elde edilen sonuç görülmektedir. Ancak görüntüde gürültü de oluşmuştur. Oluşan gürültüyü gidermek ve diğer adımların performansını artırmak için morfolojik işlemler uygulanmıştır. Aşınma ve genişleme işlemleri bu morfolojik işlemlerdir. Aşınma operatörü, ön plan piksellerin bölge sınırlarını aşındırmaktadır. Aşınma işleminde (5.1)'de yer alan yapılandırma elemanı kullanılmıştır. Giriş görüntüsündeki her bir ön plan pikseli, yapılandırma elemanının merkezi ile hizalanmış ve maskelenerek aşınma işlemi gerçekleştirilmiştir.

$$SE_{aşınma} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad (5.1)$$

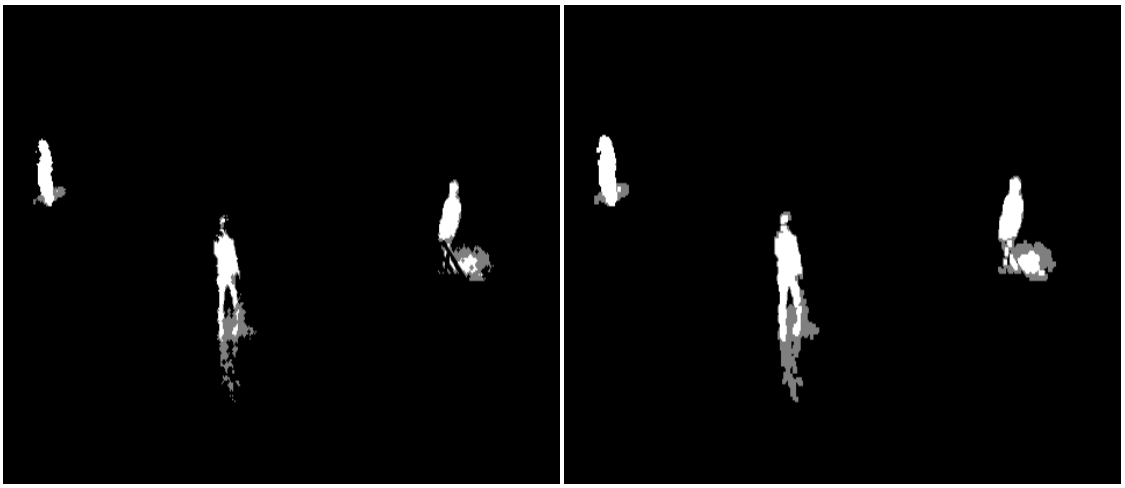


(a) (b)

Şekil 5.4 Aşınma işlemi sonucu (a) Gauss karışımı sonrası sonuç (b) Aşınma sonucu

Aşınma işleminden sonra, aşınma sonucunda elde edilen görüntüye genişleme işlemi uygulanmıştır. Genişleme işlemi aşınma işleminin ikili çalışanıdır. Genişleme işleminde (5.2)'de yer alan yapılandırma elemanı kullanımıştır. Genişleme işleminden sonra, ön plan bölgelerinin büyüdüğü ve bölgedeki deliklerin küçüldüğü elde edilen sonuçlarda görülmüştür.

$$SE_{genişleme} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad (5.2)$$



(a) (b)

Şekil 5.5 Genişleme işlemi sonucu (a) Aşınma sonucu (b) Genişleme sonucu

Genişleme işleminden sonra görüntüdeki bağlı bileşenleri bulunmuştur. Bağlı bileşenler bir aday bölge oluşturmakta ve bu şekilde nesne temsil edilmektedir. Bağlı bileşenler etiketlendikten sonra, iki eksen için ek marj eklenmiş ve bileşenlerin sınırları kullanılarak ROI'ler oluşturulmuştur. Oluşan ROI'ler hareketli nesnelerin olduğu yerleri göstermektedir. Bu ROI'ler yaya tespit bloğuna verilmiş ve tespit edilen ROI'lerde yaya olup olmadığı sonucu elde edilmiştir.

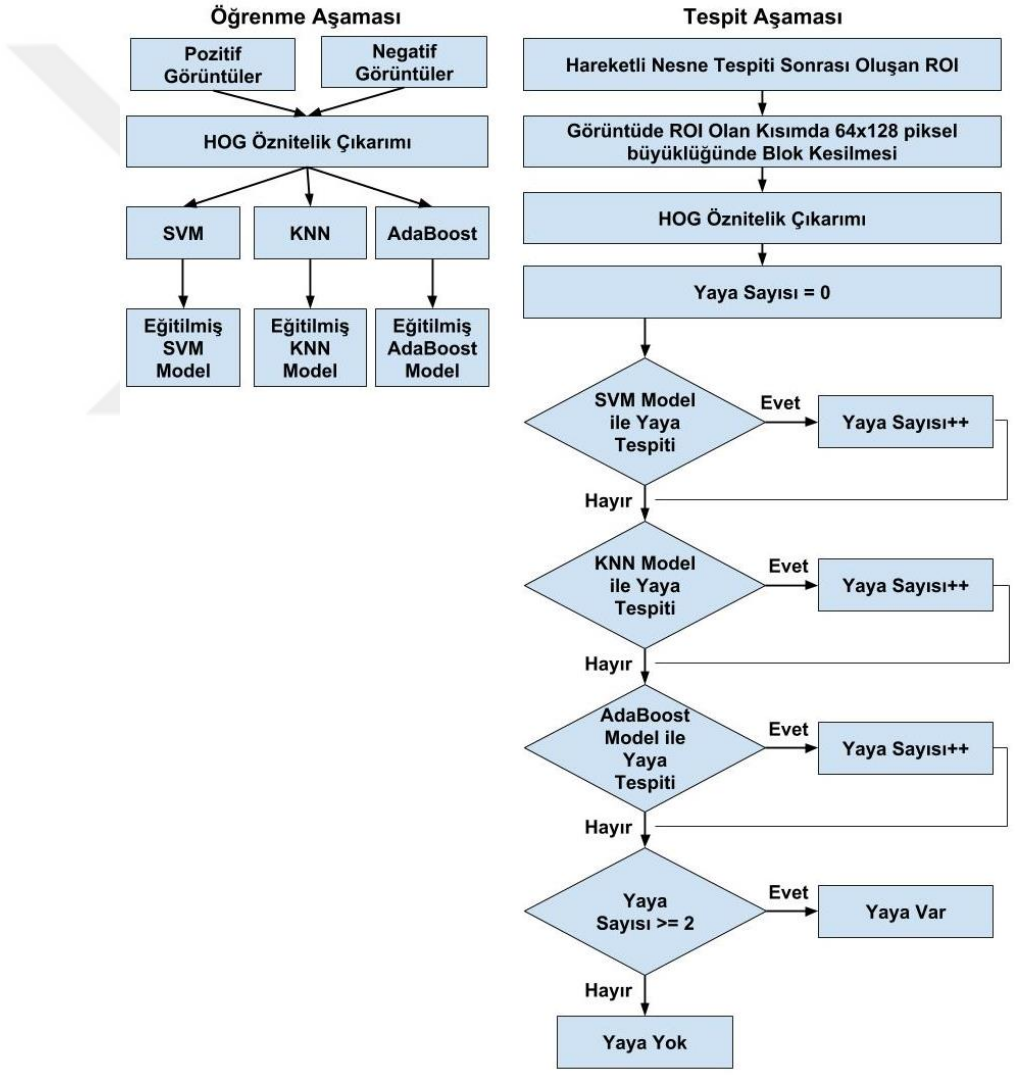


Şekil 5.6 Hareketli nesne takibi bloğu sonucu elde edilen ROI'ler

Şekil 5.6'te ilgilenilen çerçevede 3 adet hareketli nesne tespit edilmiş ve tespit edilen nesneler belirlenerek ROI'ler oluşturulmuştur.

5.2 Yaya Tespiti

Yaya tespiti aşaması özniteliklerin çıkarılması ve sınıflandırma basamaklarından oluşmaktadır. Görüntülerde HOG tabanlı yaya tespit etme yönteminde eğitim ve tespit olmak üzere iki faz bulunmaktadır. Eğitim aşamasında, görüntü pencereleri için insan / insan değil olmak üzere, bir karar vermek için sınıflandırıcı eğitilmiştir. Tespit aşamasında ise eğitilmiş sınıflandırıcı kullanılarak hareketli nesne tespiti aşamasında belirlenen ROI'ler kullanılarak görüntüde yaya olup olmadığı tespit edilmiştir. Çalışmada yaya tespiti için kullanılan yöntem için gerçekleştirilen eğitim ve tespit fazlarının akış şeması Şekil 5.7'te gösterilmiştir.

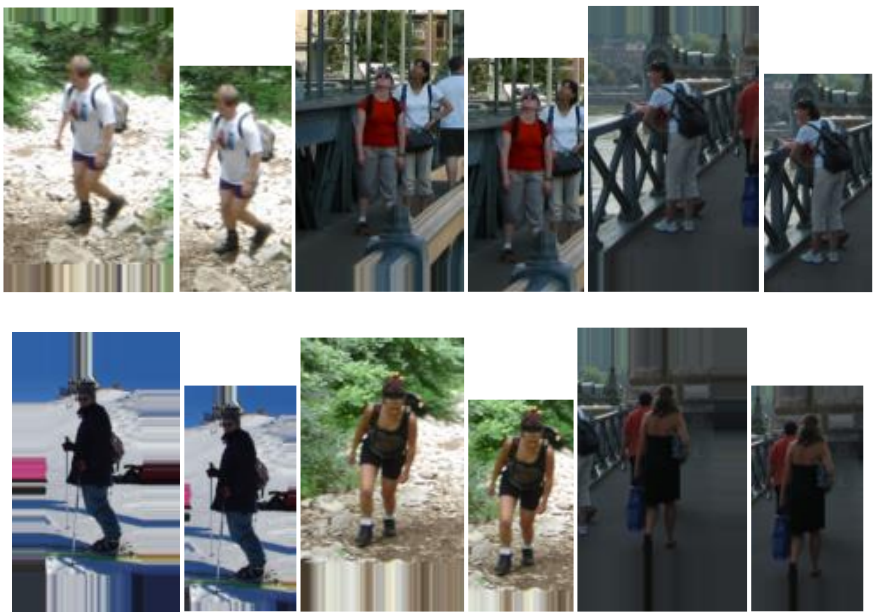


(a) (b)

Şekil 5.7 Yaya tespit mimarisi a) Eğitim aşaması b) Tespit aşaması

Eğitim aşamasında eğitilmiş bir sınıflandırıcı elde etmek için bir veri kümesi kullanılmalıdır. Yaya tespiti çalışmalarında kullanılan geniş veri kümeleri bulunmaktadır. Bir yaya veri tespit kümesi, yaya içeren pozitif ve yaya içermeyen negatif resim gruplarına sahiptir. Veri kümelerinde eğitim aşamasında kullanılacak eğitim görüntü örnekleri ve tespit aşamasında kullanılacak test görüntü örnekleri bulunmaktadır. Yapılan çalışmada yaya tespit sistemlerinde en çok kullanılan veri kümesi olan INRIA veri kümesi kullanılmıştır. Bu veri kümesinde resim içerisinde dik olarak duran yani ayakta olan, yürüyen, koşan yayalar bulunmaktadır. Bu veri kümesi, farklı çözünürlüklere, farklı ışık ve renk çeşitliliğine sahip pozitif ve negatif resimler içermektedir.

Eğitim için kullanılacak veri kümesi, 64x128 piksel boyutunda ve yaklaşık 96 piksel yüksekliğinde yaya içeren 2416 adet pozitif resimlerden ve hiç yaya içermeyen 1218 adet negatif resimlerden oluşmaktadır. Veri kümesindeki pozitif resimler 64x128 piksel olduğundan resimlere sağ ve soldan 16'şar piksel, aşağı ve yukarıdan 31'er piksel ek yapılarak 96x160 piksel boyutlu resimler olarak ayarlanmıştır. Bu ayarlama bu resimlerin pencere dışına taşması gereken öznitelik bulma yöntemleri kullanılması için yapılmıştır. Bu çalışmada HOG ile öznitelik çıkarımı yapıldığından pozitif resimlerde pencere dışına taşma gerekmemektedir. Bundan dolayı eklenen ekler kırpılarak eğitim aşamasında kullanılmıştır.



Şekil 5.8 96x160 piksel boyutundan 64x128 piksel boyutuna dönüştürülme

INRIA veri kümesinde bulunan negatif görüntüler, farklı dış ortamlarda hiçbir yayanın olmadığı farklı piksel boyutlarında olan 1218 adet görüntü içermektedir. Bu negatif resimler eğitim aşaması için istenilen boyutta kesilerek kullanılmaktadır. Yapılan çalışmada bu resimler, her bir resim için 64x128 piksel boyutunda 20 adet rastgele görüntü olmak üzere kırpılmıştır ve bu sayede toplamda 24360 adet görüntü oluşturulmuştur.



Şekil 5.9 INRIA veri kümesindeki negatif görüntü örnekleri



Şekil 5.10 64x128 piksel boyutuna dönüştürülmüş negatif görüntü örnekleri



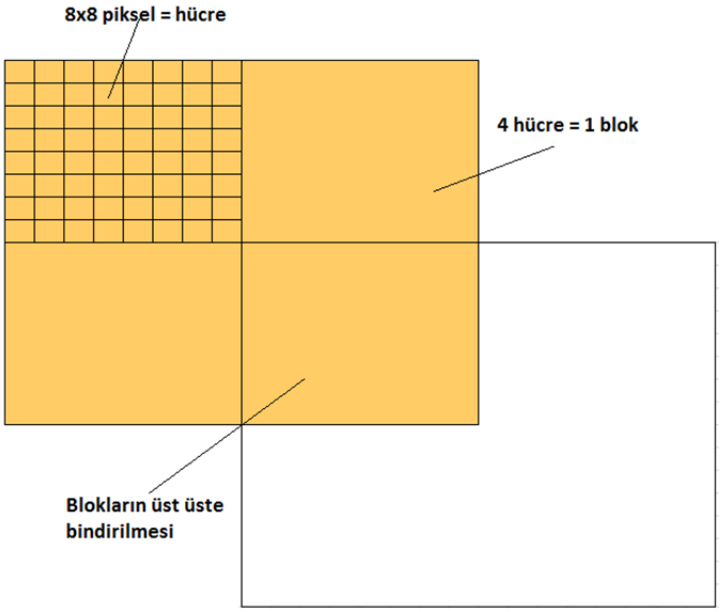
Şekil 5.11 64x128 piksel boyutuna dönüştürülmüş negatif görüntü örnekleri

Eğitim örnekleri 64x128 piksel boyutundaki resimlerden oluşmaktadır. Oluşturulan eğitim örneklerindeki resimler kullanılarak ilk olarak özniteliklerin hesaplanması gerekmektedir.

5.2.1 Özniteliklerin Çıkarılması

Yapılan çalışmada öznitelikler hesaplanırken 64x128 pikselden oluşan pencereler kullanılmıştır. HOG ile öznitelik hesaplanmasında 8x8 pikselden oluşan hücre, 2x2 hücreden oluşan blok ve içerisinde 9 öğe bulunan yön vektörü histogramı kullanılmıştır. Her bir blokta 16x16 piksel bulunmaktadır. Öznitelikler hesaplanırken bir pencere içerisinde 8 yatay ve 16 dikey hücre yer almaktadır. Başka bir ifade ile her bir pencere içerisinde 7 yatay ve 15 dikey blok bulunmaktadır. Öznitelikler hesaplanmadan önce ön işlem olarak normalizasyon yapmak amacıyla histogram eşitleme işlemi yapılmıştır. Ardından yön vektörleri hesaplanmıştır. Yön vektörleri hesaplanırken x ve y düzlemlerinde tek boyutlu bir türev maskesi uygulanmıştır. Türev maskesi hesaplanmasının detaylı anlatımı bölüm 4'te bulunmaktadır. Yön vektörlerinin hesaplanmasının ardından yön vektörlerinin histogramı oluşturulmuştur. Histogram oluşturulurken 9 öğe kullanıldığı için her bir hücredeki pikseller 0 - 180 derece arasında 20 derecelik aralıklara dağılmıştır. Sonuç olarak 20 derecelik aralıklarla yön vektörü histogramı elde edilmiştir. Bu aşamadan sonra görüntüdeki faklı bölgelerdeki yön vektörlerini normalleştirmek için hücreler birleştirilerek, bloklar elde edilmiştir. Bloklar oluşturulurken R-HOG blok geometrisi kullanılmıştır. Blokları elde ederken, bloklar üst

üste bindirilmiş ve blokların arasında kesişen hücreler bırakılmıştır. Yapılan çalışmada kullanılan hücre ve blok yapısı Şekil 5.12 da görülmektedir.



Şekil 5.12 Kullanılan hücre ve blok yapısı [53]

Şekilde de görüldüğü gibi 8x8 piksel alanı bir hücreyi, 2x2 hücreden oluşan alan ise bir blok alanını ifade etmektedir. Şekilde blokların üst üste bindirildiği gözükmektedir. Bloklar üst üste bindirilerek, bölgeler arasındaki geçişlerde oluşacak ışık, yoğunluk gibi etkilerin normaleştirilmesi amaçlanmıştır. Böylece normalleştirme işlemi uygulanarak HOG tanımlayıcının ışık ve gölgelere karşı tolerans elde etmesi sağlanmıştır. Yapılan çalışmada blok normalleştirilmesinde L2-norm yöntemi ile daha iyi sonuç elde edildiğinden, bu yöntem tercih edilmiştir.

Özetle her bir pencerede 7 yatay, 15 dikey blok bulunmaktadır. Her bir blok 4 hücre içermekte ve her bir hücrede 9 ögeden oluşan yön vektörü histogramı bulunmaktadır. Bundan dolayı öznitelik çıkarımında her bir pencere için (7x15x4x9) 3780 boyutunda bir vektör kullanılmıştır. Yapılan çalışma ile bir görüntü üzerinde çıkarılmış olan yön vektörleri Şekil 5.13 da gösterilmektedir.

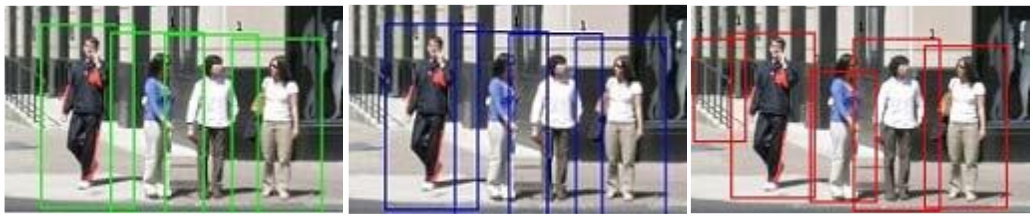


Şekil 5.13 Yön vektörleri çıkarılan görüntü [53]

5.2.2 Sınıflandırma

Eğitim aşamasında INRIA veri setinden oluşturulan pozitif ve negatif görüntülerin öznitelikleri çıkarılmış ardından SVM, kNN ve Adaboost olmak üzere üç farklı sınıflandırıcı ile eğitilerek üç farklı model dosyası elde edilmiştir. kNN sınıflandırıcı ile eğitilirken örnekler arası benzerlikleri hesaplamak için hesaplama yöntemi olarak öklid mesafesi kullanılmış ve K değeri 5 olarak seçilmiştir. Adaboost sınıflandırıcı eğitilirken ise öğrenici olarak karar verme ağacı kullanılmış ve 100 iterasyon ile model dosyası elde edilmiştir [53].

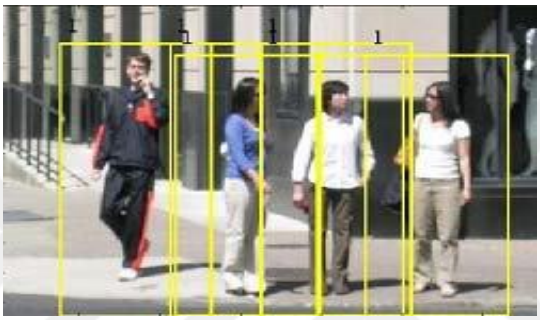
Test aşamasında hareketli nesne takibi sonucunda elde edilen ROI'lerin öznitelikleri çıkarılarak elde edilen farklı üç model dosyası ile sınıflandırılmıştır. Her bir ROI için her bir model dosyası tespit ettiği yaya sayısı için bir sayaç kullanmakta ve sayacı artırmaktadır. Sayaç ile sayılan yaya sayısı ikiden fazla olduğu durumda ROI'de yaya olduğu sonucuna varılmaktadır. Örnek bir görüntüde üç farklı sınıflandırıcı ile elde edilmiş olan sonuç Şekil 5.14 de gösterilmektedir.



(a) (b) (c)

Şekil 5.14 Sınıflandırma sonuçları(a) SVM sonucu (b) kNN sonucu (c) Adaboost sonucu[53]

Örnek görüntü üzerinde elde edilen sonuçlara göre SVM ve Adaboost sınıflandırıcı ile 4 yaya, kNN sınıflandırıcısı ile 5 yaya tespit edilmiştir. SVM ve Adaboost ile doğru tespit yapılmasına rağmen kNN ile görüntüde yer alan merdiven tırabzanları da yaya olarak tespit edilmiştir. Önerilen sınıflandırma yöntemi kullanıldığında tüm sınıflandırıcıların elde ettiği sonuçlara bakılarak, en az iki sınıflandırıcının yaya olarak tespit ettiği bölgeler yayanın bulunduğu bölgeler olarak tespit edilmiştir. Önerilen sınıflandırıcı kullanıldığında kNN sınıflandırıcının yanlış olarak tespit ettiği yaya elemine edilerek tespit yapılmıştır. Şekil 5. 15'de önerilen sınıflandırıcı ile elde edilen sonuç gösterilmektedir. Bu sonuca göre görüntüde 4 yaya yer almaktadır [53].



Şekil 5.15 Önerilen sınıflandırıcı ile elde edilen sonuç [53]

5.3 Yaya Takibi

Yapılan çalışmada yaya takibi için kalman filtresi kullanılmıştır. Hareketli nesnenin tespit edilip, yaya olup olmadığı sınıflandırıldıktan sonra, kalman takipçi nokta izleyici olarak kullanılmıştır. Hareketli yayanın merkez pozisyonu, tahmin edilecek ana nokta olarak kullanılmıştır. Kalman filtresinin durum vektörü, birim zaman aralığı başına x ve y yönlerinde, x ve y pozisyonları ve yer değiştirmeler olarak tanımlanmıştır.

$$X_k = (x_c\ y_c\ v_x\ v_y) \tag{5.3}$$

Kalman filtreleme algoritması, bir ölçüm hatalarına dayanarak durum vektörünü tahmin eder. Durum modeli doğrusal olarak aşağıdaki gibi tanımlanmıştır;

$$X_k = AX_{k-1} + \omega_{k-1} \tag{5.4}$$

$$A = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.5}$$

A, geçiş matrisi ve ω tahmin hatası vektörüdür. Ayrıca, durum vektörü ve ölçümler arasındaki doğrusal bir ilişki olduğu varsayılır.

$$Z_k = HX_k + v_k \tag{5.6}$$

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{5.7}$$

H, gözlem matrisi ve v, ölçüm hatasıdır. Geçiş ve gözlem matrisinin bu değerlerini kullanarak, 4. bölümde verilen zaman-güncelleme ve ölçüm-güncelleme denklemleri, bir sonraki durum nesnesinin tahminini vermiştir. Böylece tespit edilen yayanın takibi sağlanmıştır. Şekil 5.16 te bir video görüntüsündeki yayanın takip edilmesi gösterilmektedir.



Şekil 5.16 Video görüntüsünde yaya takibi pencereleri

BÖLÜM 6

SONUÇ VE ÖNERİLER

Bu çalışmada sabit kamera tarafından kaydedilen videolarda yayaların tespit edilmesi ve tespit edilen yayaların takip edilmesi için bir sistemin geliştirilmesi ve bu sistem ile videolarda yayaların takip edilmesi amaçlanmıştır. Çalışmanın ikinci bölümünde yaya takip yöntemleri hakkında genel bilgiler verilmiştir. Verilen bu bilgiler değerlendirildiğinde yaya takip sistemlerinin geliştirilmesi için üç aşamalı bir çalışma gerekliliği görülmüştür. Bu aşamalar hareketli nesne tespiti, yaya tespiti ve yaya takibi olarak listelenmiştir.

Çalışmanın üçüncü bölümünde videolarda hareketli nesnelerin tespiti üzerinde durulmuştur. Yayaları tespit etmek için bütün videoda aramak yerine yayaların olabileceği muhtemel bölgelerde aranması için bu işlem gerçekleştirilmiştir. Tespit edilen hareketli nesne bölgeleri ilgi alanları olarak alınmış ve bu bölgelerde yayaların aranması için yaya tespit bloğuna verilmiştir. Hareketli nesnelerin tespiti için bu çalışmada ön plan çıkarımı yöntemlerinden olan gauss karışımı yöntemi kullanılmış ve morfolojik işlemler uygulanmıştır.

Dördüncü bölümde hareketli nesne bloğu tarafından çıkarılmış olan ilgi alanları içerisinde bulunan yayaların tespiti ve tespit edilen yayaların takibi üzerinde durulmuştur. Yaya tespiti gerçekleştirilirken özniteliklerin çıkarılması için kullanılan HOG yönteminin basamakları anlatılmış ve sınıflandırma gerçekleştirmek için SVM, kNN ve Adaboost sınıflandırıcıları anlatılmıştır. Tespit edilen yayaları takip etmek ve hareket güzergâhının çıkarılması için kalman filtresi kullanılmıştır.

Beşinci bölümde ise uygulamanın gerçekleştirilmesi anlatılmıştır. Gerçekleştirilen çalışmada SVM, kNN ve Adaboost sınıflandırıcıları kullanılarak komite oylamasına dayalı olarak çalışan yeni bir sınıflandırıcı önerilmiştir. Bu bölümde geliştirilen sınıflandırıcı kullanılarak videolarda yayaların tespit edilmesi ve hareket güzergâhının oluşturularak takip edilmesi uygulaması gerçekleştirilmiştir.

6.1 Performans Ölçümü

Önerilen algoritmanın performans değerlendirmesi yapılarak sonuçların yorumlanması için bazı metriklerin tanımlanması gerekmektedir. Bu metriklerin tanımları aşağıdaki gibi verilmiştir [54].

6.1.1 Metrikleri Tanımı

Performansı değerlendirmek için kullanılan metrikler aşağıdaki gibi tanımlanmıştır:
Doğru Pozitif (DP): Pozitif olan ve tespit sistemi tarafından da pozitif olarak tespit edilen örneklerdir.

Hatalı Pozitif (HP): Negatif olan ancak tespit sistemi tarafından pozitif olarak tespit edilen örneklerdir.

Doğru Negatif (DN): Negatif olan ve tespit sistemi tarafından da negatif olarak tespit edilen örneklerdir.

Hatalı Negatif (HN): Pozitif olan ancak tespit sistemi tarafından negatif olarak tespit edilen örneklerdir.

Çizelge 6.1 Değerlendirme metrikleri için terminoloji çizimi

		Koşul	
		Var	**Yok**
Test	**Pozitif**	Doğru Pozitif	Hatalı Pozitif
	Negatif	Hatalı Negatif	Doğru Negatif

6.1.1.1 Duyarlılık Oranı

Bir sınıflandırıcı performansı için değerlendirme metriklerinden biri duyarlılık oranıdır. Duyarlılık oranı doğru sınıflandırılan pozitif örnek sayısının toplam pozitif örnek sayısına oranlanması ile hesaplanır.

$$Duyarlılık\ Oranı = \frac{DP}{DP+HN} \tag{6.1}$$

6.1.1.2 Kesinlik Oranı

Bir sınıfın kesinliği, bu sınıfın öngörülmüş olması koşuluyla, doğruluğun bir ölçütüdür. Kesinlik oranı sınıfı 1 olarak tahmin edilen doğru pozitif örnek sayılarının, sınıfı 1 olarak tahmin edilen bütün örnek sayılarına oranlanarak hesaplanmaktadır.

$$Kesinlik\ Oranı = \frac{DP}{DP+HP} \tag{6.2}$$

6.1.1.3 Doğruluk Oranı

Doğruluk oranı, sınıflandırıcının genel doğruluğu olup, toplam sınıflandırma sayısına bölünmesiyle doğru sınıflandırmaların toplamı olarak hesaplanır.

$$Doğruluk\ Oranı = \frac{DP+DN}{DP+HP+DN+HN} \tag{6.3}$$

6.1.1.4 Hatalı Pozitif Oranı

Hatalı pozitif oranı, negatif örneklerden kaç tanesinin tespit sistemi tarafından pozitif olarak tespit edildiğini gösterir. Bu oran sıfıra yaklaştıkça sınıflandırıcının başarı oranı artmaktadır [55].

$$Hatalı\ Pozitif\ Oranı = \frac{HP}{HP+DN} \tag{6.4}$$

6.1.1.5 Karışıklık Matrisi

Karışıklık matrisi, bir algoritmanın performansının görselleştirilmesini sağlayan özel bir tablodur. Matrisin her bir sütunu, öngörülen bir sınıftaki örneklerin sayısını temsil ederken, her bir satır, gerçek bir sınıftaki örneklerin sayısını temsil eder.

6.2 Performans Testi

Performans testi gerçekleştirilirken pencere başarılarının karşılaştırılması test edilmiştir. Bu yöntemde tespit sistemine sadece test edilmek istenen pencereler verilmekte ve pencerelerin pozitif veya negatif olarak sınıflandırılması istenmektedir. Sınıflandırıcıların test performansını değerlendirmek için INRIA veri seti kullanılmıştır. Test yapmak için kullanılan veri setinde 70x134 piksel boyutunda ve yaklaşık 96 piksel yüksekliğinde 1126 adet pozitif görüntü ve 453 adet negatif görüntü yer almaktadır. Test aşamasında SVM, kNN ve Adaboost sınıflandırıcıları ve çoğunluk oylamasına dayalı olarak çalışan sınıflandırıcının testi gerçekleştirilmiştir. Çoğunluk oylamasına dayalı olan sınıflandırıcı yanında kıyaslama oluşturması açısından bütün sınıflandırıcıların ortak kararı test edilmiştir. Elde edilen sonuçlar Çizelge 6.2 ve 6.3'de gösterilmiştir.

Çizelge 6.2 Karışıklık matrisi değerleri

Sınıflandırma Yöntemi	Doğru Pozitif	Hatalı Negatif	Hatalı Pozitif	Doğru Negatif
SVM	967	159	7	446
Adaboost	982	144	7	446
kNN	946	180	9	444
Ortak karar	931	195	2	451
Önerilen Yöntem	1003	123	2	451

Çizelge 6.3 Performans sonuçları

Sınıflandırma Yöntemi	Duyarlılık Oranı	Kesinlik Oranı	Doğruluk Oranı	Hatalı Pozitif Oranı
SVM	% 85,8	% 99,3	% 89,5	% 1,54
Adaboost	% 87,2	% 99,3	% 90,5	% 1,54
kNN	% 84,0	% 99,0	% 88,0	% 1,99
Ortak karar	% 82,6	% 99,8	% 87,5	% 0,44
Önerilen Yöntem	% 89,1	% 99,8	% 92,1	% 0,44

6.3 Sonuçlar

Yapılan çalışmada üç farklı sınıflandırıcı kullanılarak komite oylamasına dayalı yeni bir sınıflandırıcı önerilmiştir. Önerilen bu sınıflandırıcı geliştirilirken SVM, kNN ve Adaboost sınflandırıcılarının herhangi ikisinin yaya olarak tespit ettiği bölgelerde yaya olduğu sonucuna varılmıştır. Geliştirilen sınıflandırıcı kullanılarak INRIA veri seti ile test yapıldığında Çizelge 6.1 ve Çizelge 6.2'de gösterilen sonuçlar elde edilmiştir. Elde edilen sonuçlar irdelendiğinde önerilen sınıflandırıcı kullanılarak yapılan yaya tespitinde doğruluk oranının artığı ve hatalı pozitif oranının azaldığı görülmektedir. Bu sonuca bakarak komite oylaması yöntemiyle oluşturulan yeni sınıflandırıcının yaya tespitinde daha başarılı olduğu yorumu yapılabilir. Karşılaştırmak açısından üç sınıflandırıcının ortak kararına bakılarak yaya tespiti yapıldığında doğruluk oranının azaldığı görülmüştür [53].

Daha önce bu alanda yapılan çalışmalara bakıldığında; Dalal ve Triggs tarafından yapılan çalışmada, HOG ve SVM algoritmaları kullanılarak yaya tanıma uygulaması gerçekleştirilen bir çalışma görülmüştür. Yapılan uygulama test edilirken INRIA veri seti kullanılmış ve pozitif görüntülerde %85,9, negatif görüntülerde %99,10 oranında doğruluk oranı ile tespit yapılmıştır [5]. Bir diğer çalışmada HOG, SVM ve kNN (K=10) algoritmaları beraber kullanılarak yaya tanıma çalışması yapılmıştır. Bu çalışmada ALOI veri seti kullanılmış ve %86,72 olarak doğruluk oranı tespit edilmiştir [12]. Yapılan çalışmada ise %92,1 doğruluk oranı ile tespit yapıldığı gözlenmiştir. Yapılan çalışma, literatürde yer alan bu çalışmalarla kıyaslandığında daha başarılı bir sınıflandırıcı oluşturulduğu görülmüştür.

Sonuç olarak, yapılan çalışma ile videodaki yayalar tespit edilerek takip edilmiş ve hareket güzergâhları çıkarılmıştır. Yapılan çalışma MATLAB ortamında gerçekleştirilmiş ve takip edilen yayaları içeren video sonuç videosu olarak kaydedilmiştir. Gelecekte yapılacak çalışmalarda üç farklı sınıflandırıcının ayrı ayrı eğitilip komite oylaması ile sınıflandırmak yerine, bu üç sınıflandırıcının özellikleri kullanılarak tek bir sınıflandırıcı yapılması amaçlanmaktadır.

KAYNAKLAR

[1] Lowe, D.G., (1999). “Object Recognition from Local Scale-Invariant Features”, The Proceedings of the Seventh IEEE International Conference on Computer Vision, 20-27 September 1999, Kerkyra, 1150-1157.

[2] Elgammal, A., Harwood, D. ve Davis, L., (2000). “Non-parametric Model for Background Subtraction”, Computer Vision—ECCV 2000, 751-767.

[3] Papageorgiou, C. ve Papageorgiou, T., (2000). “A Trainable System for Object Detection”, International Journal of Computer Vision, 38: 15-33.

[4] Viola, P., Jones, M.J. ve Snow, D., (2003). “Detecting Pedestrians Using Patterns of Motion and Appearance”, The Proceedings of the Ninth IEEE International Conference on Computer Vision, 13-16 October 2003, Nice.

[5] Dalal, N. ve Triggs, B., (2005). “Histograms of Oriented Gradients for Human Detection”, IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), 20-25 June 2005, 1: 886-893.

[6] Dalal N., (2005). INRIA Person Dataset, http://pascal.inrialpes.fr/data/human/, 11 Ocak 2018.

[7] Center for Biological and Computational Learning, (2013). CBCL Pedestrian Database, http://cbcl.mit.edu/software-datasets/PedestrianData.html, 17 Ocak 2018.

[8] Dalal, N., Triggs, B. ve Schmid, C., (2006). “Human Detection Using Oriented Histograms of Flow and Appearance”, 9th European Conference on Computer Vision, 7-13 May 2006, Graz.

[9] Sabzmeydani, P. ve Mori, G., (2007). “Detecting Pedestrians by Learning Shapelet Features”, IEEE Conference on Computer Vision and Pattern Recognition, 17-22 June 2007, Minneapolis.

[10] Wu, B. ve Nevatia, R., (2007). “Detection and Tracking of Multiple, Partially Occluded Humans by Bayesian Combination of Edgelet Based Part Detectors”, International Journal of Computer Vision, 75: 247-266.

[11] Lin, Z. ve Davis, L., (2008). “A Pose-Invariant Descriptor for Human Detection and Segmentation”, Computer Vision–ECCV 2008, 423-436.

[12] Cai, Z., Yu P., Liang Y. ve Lin B., (2016). “SVM-KNN Algorithm for Image Classification Based on Enhanced HOG Feature”, The 4th IIAE International Conference on Intelligent Systems and Image Processing, 8-12 September 2016, Kyoto.

[13] Broida, T.J. ve Chellappa, R., (1986). “Estimation of Object Motion Parameters from Noisy Images”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 90-99.

[14] Shi, J., (1994). “Good Features to Track”, 1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 21-23 June 1994, Seattle.

[15] Avidan S., (2007). “Ensemble Tracking”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 29 (2): 261-271.

[16] Masoud, O. ve Papanikolopoulos, N.P., (2001). “A Novel Method for Tracking and Counting Pedestrians in Real-time Using a Single Camera”, IEEE Transactions on Vehicular Technology, 50 (5): 1267-1278.

[17] Welch, G. ve Bishop, G., (2002). An Introduction to the Kalman filter, Technical Report, University of North Carolina at Chapel Hill, Department of Computer Science, 2001.

[18] Benfold, B. ve Reid, I., (2011). “Stable Multi-target Tracking in Real-time Surveillance Video”, IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2011), 20-25 June 2011, 3457-3464.

[19] Kuo, C.H., Huang, C. ve Nevatia, R., (2010). “Multi-target Tracking by on-line Learned Discriminative Appearance Models”, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 13-18 June 2010, San Francisco, 685-692.

[20] Lipton, A. J., Fujiyoshi, H., ve Patil, R.S., (1998). “Moving Target Classification and Tracking from Real-time Video”, Proceedings Fourth IEEE Workshop on Applications of Computer Vision, 19-21 Oct. 1998, 129–136.

[21] Wang, L., Hu, W. ve Tan, T., (2003). “Recent Developments in Human Motion Analysis”, Pattern Recognition, 36 (3): 585-601.

[22] Collins, R. T., Lipton, A.J. ve Kanade, T., (2000). “A System for Video Surveillance and Monitoring: VSAM Final Report”, Technical Report, Robotics Institute, Carnegie Mellon University, Pittsburgh.

[23] Heikkila, J. ve Silven, O., (1999). “A Real-time System for Monitoring of Cyclists and Pedestrians”, Proceedings Second IEEE Workshop on Visual Surveillance, 26-26 June 1999, Fort Collins.

[24] Stauffer, C. ve Grimson, W.E.L., (2000). “Learning Patterns of Activity Using Real-time Tracking”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 22 (8): 747-757.

[25] Elgammal, A., Duraiswami, R., Harwood, D., ve Davis, L.S., (2002). "Background and Foreground Modeling Using Nonparametric Kernel Density Estimation for Visual Surveillance", Proceedings of the IEEE, 90(7): 1151-1163.

[26] Han, B., Comaniciu, D. ve Davis, L., (2004) "Sequential Kernel Density Approximation Through Mode Propagation: Applications to Background Modeling", Sixth Asian Conferenceon Computer Vision, 27-30 January 2004, Jeju.

[27] Wang, L., Hu, W. ve Tan, T., (2003). "Recent Developments in Human Motion Analysis", Pattern Recognition, 36 (3): 585-60.

[28] Bay, H., Tuytelaars, T., ve Gool, L. V., (2006). "SURF: Speeded Up Robust Features", Computer Vision–ECCV 2006, 404-417.

[29] Yilmaz, A., Javed, O. ve Shah M., (2006). "Object Tracking: A Survey", ACM Computing Surveys, 38 (4): 1-45.

[30] Kim, I.S., Choi, H.S., Yi, K.M., Choi, J.Y. ve Kong, S.G., (2010). "Intelligent Visual Surveillance - A Survey", International Journal of Control, Automation and Systems, 8 (5): 926-939.

[31] Stauffer, C. ve Grimson, W.E.L., (2000). "Learning Patterns of Activity Using Realtime Tracking", IEEE Transactions on Pattern Analysis and Machine Intelligence, 22 (8) : 747-757.

[32] Jain, R., Kasturi, R., ve Schunk, B.G., (1995). Machine Vision, McGraw-Hill, Inc. New York.

[33] Haralick, R.M. ve Shapiro, L.G., (1992). Computer and Robot Vision, First Edition, Addison-Wesley Longman Publishing Co., Inc. Boston.

[34] Gibbons, A., (1985). Algorithmic Graph Theory, Cambridge University Press, Cambridge.

[35] Akkuş, H. E., (2009). Following a Moving Object in a Bounded Area, Yüksek Lisans Tezi, Çankaya Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

[36] Samet, H. ve Tamminen, M., (1988). "Efficient Component Labeling of Images of Arbitrary Dimension Represented by Linear Bintrees", IEEE Transactions on Pattern Analysis and Machine Intelligence, 10 (4) : 579-586.

[37] González, R.C. ve Woods R.E., (2002). Digital Image Processing, Second Edition, Prentice-Hall, Inc., New Jersey.

[38] Topçu H.H., (2012). Human Body Part Detection and Multi-human Tracking in Surveillance Videos, Yüksek Lisans Tezi, Orta Doğu Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Ankara.

[39] Wu B, ve Nevatia R., (2008). "Optimizing Discrimination-Efficiency Tradeoff in Integrating Heterogeneous Local Features for Object Detection", IEEE Conference on Computer Vision and Pattern Recognition, 23-28 June 2008, Anchorage.

[40] Cortes, C., ve Vapnik, V., (1995). “Support-Vector Networks”, Machine Learning, 20 (3): 273-297.

[41] Begg, R. K., Palaniswami, M., ve Owen, B., (2005). “Support Vector Machines for Automated Gait Classification”, IEEE Transactions on Biomedical Engineering, 52 (5): 828-838.

[42] Vapnik, V.N., (1995). The Nature of Statistical Learning Theory, Springer-Verlag, Berlin.

[43] Aha, D. ve Kibler D., (1991). “Instance-based learning algorithms”, Machine Learning, 6: 37-66.

[44] Şengül, G., (2016). “Classification of Parasite Egg Cells Using Gray Level Cooccurence Matrix and kNN”, Biomedical Research, 27 (3): 829-834.

[45] Sudha, L. R., ve Bhavani, R., (2012). “Gait Based Gender Identification Using Statistical Pattern Classifiers”, International Journal of Computer Applications 40 (8): 30-35.

[46] Barkana, B. D., ve Zhou, J., (2015). “A New Pitch-range Based Feature Set for a Speaker’s Age and Gender Classification”, Applied Acoustics, 98: 52-61.

[47] Freund, Y. ve Schapire R. E., (1997). “A Decision-Theoretic Generalization of On-line Learning and an Application to Boosting”, Journal of Computer and System Sciences, 55 (1) :119–139.

[48] Zöngür, U., (2009). Detection of Airport Runways in Optical Satelite Images, Yüksek Lisans Tezi, Orta Doğu Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Ankara.

[49] Bayhan, M.Ş., (2008). Scale and Pose Invariant Real-time Face Detection and Tracking, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul.

[50] Bar-Shalom, Y. Li X. Kirubarajan T., (2002). Estimation with Applications to Tracking and Navigation, John Wiley & Sons, Inc., New York.

[51] R. E. Kalman, (1960). “A New Approach to Linear Filtering and Prediction Problems”, Transactions of the ASME–Journal of Basic Engineering, 82:35-45.

[52] Ergezer, H., (2007). Visual Detection and Tracking of Moving Objects, Yüksek Lisans Tezi, Orta Doğu Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Ankara.

[53] Çavdar, Ç., Özek, A. ve Bolat, B. (2018). "Pedestrian Detection with Multiple Classifiers on Still Images", IEEE SIU 26th Signal Processing and Communication Applications Symposium, 2-5 May 2018, Çeşme.

[54] Duman, D., (2013). Moving Vehicle Classification, Yüksek Lisans Tezi, Orta Doğu Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Ankara.

[55] Tetik, Y.E., (2014). Sayısal Resimlerdeki Yayaların Tespiti, Doktora Bitirme Tezi, Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Abdullah ÖZEK

Doğum Tarihi ve Yeri : 01.01.1988 Ereğli/KONYA

Yabancı Dili : İngilizce

E-posta : abdullahozek42@gmail.com

ÖĞRENİM DURUMU

Derece	Alan	Okul/Üniversite	Mezuniyet Yılı
Lisans	Elektronik Elektronik Mühendisliği	Dokuz Eylül Üniversitesi	2012
Lise	Fen Bilimleri	Ereğli Anadolu Lisesi	2006

İŞ TECRÜBESİ

Yıl	Firma/Kurum	Görevi
2012 - …	NETAŞ	Yazılım Tasarım Mühendisi

YAYINLARI

Bildiri

1. Çavdar, Ç., Özek, A. ve Bolat, B. (2018). "Pedestrian Detection with Multiple Classifiers on Still Images", IEEE SIU 26th Signal Processing and Communication Applications Symposium, 2-5 May 2018, Çeşme.