T.C.
KONYA TEKNİK ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

GÖRÜNTÜ İŞLEME İLE AĞAÇ MEYVE YÜKÜNÜN HESAPLANMASI

Gamze Hikmet YAŞAR

YÜKSEK LİSANS TEZİ

Elektrik- Elektronik Mühendisliği Anabilim Dalı

Nisan-2019
KONYA

TEZ KABUL VE ONAYI

Gamze Hikmet YAŞAR tarafından hazırlanan "Görüntü İşleme ile Ağaç Meyve Yükünün Hesaplanması" adlı tez çalışması 08/04/2019 tarihinde aşağıdaki jüri üyeleri tarafından oy birliği ile Konya Teknik Üniversitesi Lisansüstü Eğitim Enstitüsü Elektrik- Elektronik Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Doç. Dr. A. Afşin KULAKSIZ	.....
Danışman Doç Dr. Bayram AKDEMİR	.....
Üye Dr. Öğr. Üyesi Mümtaz MUTLUER	.....

Yukarıdaki sonucu onaylarım.

Prof. Dr. Yakup KARA
Enstitü Müdürü

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Gamze Hikmet YAŞAR

Tarih: 02.05.2019

ÖZET

YÜKSEK LİSANS TEZİ

GÖRÜNTÜ İŞLEME İLE AĞAÇ MEYVE YÜKÜNÜN HESAPLANMASI

Gamze Hikmet YAŞAR

Konya Teknik Üniversitesi
Lisansüstü Eğitim Enstitüsü
Elektrik- Elektronik Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Bayram AKDEMİR

2019, 98 Sayfa

Jüri
Doç. Dr. A. Afşin KULAKSIZ
Doç. Dr. Bayram AKDEMİR
Dr. Öğr. Üyesi Mümtaz MUTLU

Çoğu uygulamada yaygın hale gelen görüntü işleme teknikleri, son zamanlarda tarım alanında da en çok tercih edilen teknikler arasında yerini almıştır. Kültürümüzde, yetiştirilmiş olan meyvelerin henüz dalında iken kümülatif bir ürün çıktısına karşı ölçüp tartmadan (kabala satış) satıldığı bilinmektedir. Bu yöntemde, alıcı ve satıcı arasında oluşabilecek dengesizliğin giderilmesine mühendislik bakış açısı getirerek her iki tarafı da dengeye getirmeyi amaçlayan bir çalışma gerçekleştirilmiştir. Tarımda görüntü işlemenin avantajları kullanılarak etkin tarıma katkıda bulunmak amacıyla, görüntü işleme yöntemleri ile portakal ağaçları üzerinde bulunan meyvelerin ağırlıkları bilgisayar ortamında dijital olarak belirlenmiştir. Bu uygulama sayesinde üretici ve tüketici arasında tam güven sağlamak, hızlı, pratik bir sonuç elde ederek minimum maliyet ve zarar riski hedeflenmiştir. Bu düşünceden yola çıkarak, meyve ağaçlarının dört açıdan görüntüleri alınarak bilgisayar ortamına aktarılmıştır. Aktarılan görüntülerdeki meyveler dijital ortamda ayrıştırılarak, meyveler çıkartılmıştır. Elde edilen sonuca göre meyve yükü hesaplama algoritması yazılarak, bir meyve ağacının verebileceği yaklaşık hasat miktarı dijital olarak hesaplanmıştır. Yapılan hesaplamanın görüntüye ve sayısal verilere dayalı olması tam doğruluk ve üretici ile tüketici arasında tam güven teşkil etmektedir. Geliştirilen uygulama, teknolojinin gelişmesiyle kullanım alanı çok geniş bir çevreye yayılan Android sistemli cihazlara uyumlu hale getirilerek, sadece bilgisayar kullanımına gerek kalmadan herkesin aktif kullanımını mümkün kılmaktadır. Ayrıca, uygulama geliştirilerek meyve verimini tespit edebilmek için meyve öznitelikleri yapay sinir ağları kullanılarak belirlenmiştir.

Anahtar Kelimeler: Ağaç Meyve Yükü, Android, Görüntü İşleme, Matlab, Meyve Tespiti, Tarımda Teknoloji

ABSTRACT

MS THESIS

PREDICTION OF FRUIT TREES LOAD WITH IMAGE PROCESSING

Gamze Hikmet YAŞAR

Konya Technical University
Institute of Graduate Studies
Department of Electrical Electronics Engineering

Advisor: Assoc. Prof. Dr. Bayram AKDEMİR

2019, 98 Pages

Jury
Assoc. Prof. Dr. A. Afşin KULAKSIZ
Assoc. Prof. Dr. Bayram AKDEMİR
Asst. Prof. Dr. Mümtaz MUTLU

The image processing techniques which have become prevalent in many applications have recently turned out to be one of the techniques that are frequently chosen in the field of agriculture. In our culture, the fruits are sold even when the fruits are on their trees. With this technique in dressing the imbalance between the recipient and seller, a study has been conducted in order to achieve the balance between the two by bringing a point of engineering view. By utilising the advantages of the image processing in agriculture to contribute to the efficient agriculture, the weights of orange fruits have been determined in digital environment. The aims of this study are to develop a trust between the producer and consumer, and to achieve quick and practical outcomes with a minimum cost and the risk of loss. With this in mind, the pictures od fruit trees which were taken from four perspectives have been transferred into digital environment. The fruits in pictures have been removed by decomposing them in the digital environment. According to result obtained, the amount of harvest that a fruit tree could produce has been calculated digitally by writing the fruit weigh calculation algorithm. The fact that the calculation depends on the visual and numerical data will probably form an absolute correctness and a trust between the producer and consumer. With the development of technology, the fact that the aforementioned application is made compatible to the Android system devices that the use of is being common makes the extensive use of the application without any necessity for a computer use possible. In addition, fruit characteristics are determined by the use of artificial neural networks so as to detect the yield of fruits with the improvement of this application.

Keywords: Android, Detect Fruits, Image Processing, Matlab, Prediction of Fruit Trees, Technology in Agriculture

ÖNSÖZ

Yüksek lisans tezimin hazırlanmasında başta, desteğini ve yardımını esirgemeyen Selçuk Üniversitesi Mühendislik Mimarlık Fakültesi Elektrik-Elektronik Mühendisliği Bölümü öğretim üyesi yüksek lisans danışmanım Sayın Doç. Dr. Bayram AKDEMİR'e ve öğrenim hayatım boyunca emeği geçen tüm hocalarıma teşekkürü bir borç bilirim. Ayrıca, tez çalışmam sırasında daima yanımda olan oğlum Musa Yağız'a, destek veren anneme ve eşime teşekkür ederim.

Gamze Hikmet YAŞAR
KONYA-2019

İÇİNDEKİLER

ÖZET ..... **iv**

ABSTRACT ..... **v**

ÖNSÖZ ..... **vi**

İÇİNDEKİLER ..... **vii**

SİMGELER VE KISALTMALAR ..... **ix**

1. GİRİŞ **1**

1.1. Amaç 2
1.2. Tezin Kapsamı 2

2. KAYNAK ARAŞTIRMASI **4**

3. GÖRÜNTÜ İŞLEME KAVRAMLARI **8**

3.1. Renk Kavramı 9
3.1.1. RGB renk uzayı 10
3.1.2. HSV renk uzayı 12
3.2. Görüntü Çeşitleri 14
3.2.1. Analog görüntü 14
3.2.2. Dijital görüntü 15
3.3. Görüntü İşleme Yöntemleri 19
3.3.1. Sayısal (dijital) görüntü üzerinde yapılan işlemler 20
3.4. Görüntü İşlemenin Temel Teknikleri 22

4. YAPAY SİNİR AĞLARI **30**

4.1. Biyolojik Sinir Sistemi 31
4.2. Yapay Sinir Ağlarının Matematiksel Modeli 33
4.3. Yapay Sinir Ağlarının Yapısı 35
4.3.1. Ağ yapılarına göre yapay sinir ağları 36
4.4. Yapay Sinir Ağlarının Türleri 38
4.4.1. Tek katmanlı sinir ağları 38
4.4.2. Çok katmanlı sinir ağları (MLP) 40
4.4.3. Radyal tabanlı sinir ağları 41
4.5. Yapay Sinir Ağlarında Öğrenme 43
4.5.1. Danışmalı öğrenme 43
4.5.2. Danışmasız öğrenme 44
4.5.3. Takviyeli öğrenme 45
4.6. Yapay Sinir Ağlarında Öğrenme Kuralları 46
4.6.1. Hebb kuralı 46
4.6.2. Hopfield kuralı 46
4.6.3. Kohonen kuralı 46
4.6.4. Delta kuralı 47

4.6.5. Geri yayılım algoritması 47

5. GÖRÜNTÜ İŞLEME İLE AĞAÇ MEYVE YÜKÜNÜN HESAPLANMASI 48

5.1. Donanım ve Yazılımlar 48
5.1.1. Matlab 48
5.1.2. Android studio 49
5.2.3. OpenCV 54
5.2. Yöntem 56

6. ARAŞTIRMA BULGULARI VE TARTIŞMA 59

6.1. Matlab ile Görüntü İşleme 59
6.1.1. Görüntünün sayısallaştırılması 59
6.1.2. Filtreleme İşlemi 60
6.1.3. RGB renk uzayından HSV renk uzayına dönüşüm 62
6.1.4. Histogram hesaplama 63
6.1.5. Eşikleme İşlemi 64
6.1.6. Morfolojik işlemler 67
6.1.7. Yapay sinir ağları ile kilogram tahmini 71
6.1.7. Maskelenen görüntüyü RGB görüntüye dönüştürme 73
6.2. Mobil Görüntü İşleme 75

7. SONUÇLAR VE ÖNERİLER 82

7.1 Sonuçlar 82
7.2 Öneriler 82

KAYNAKLAR 84

ÖZGEÇMİŞ 88

SİMGELER VE KISALTMALAR

Simgeler

Λ : Dalga boyu

z : Renk derinliği

λ : Renk

t : Zaman

T : Fourier dönüşüm işlem operatörü

N^2 : Global işlem genel karmaşıklığı

P^2 : Bölgesel işlem genel karmaşıklığı

u : Uzamsal frekans

v : Uzamsal frekans 2

A : Fourier dönüşüm çifti

f : Ayrık fonksiyon

N : Fourier dönüşümünde *f* fonksiyonunun eleman sayısı

F : Fourier dönüşüm frekans bileşenleri

L : Laplacian

I : Laplacian dönüşümünde piksel yoğunluk değeri

∂ : Türev operatörü

σ : Gaussian standart sapması

T : Eşikleme denkleminde eşik değeri

r_k : Histogram işleminde k. gri seviye

n_k : Histogram işleminde toplam piksel sayısı

n_j : Histogram eşitlemedeki j. gri seviyedeki piksel sayısı

n : Histogram işleminde ve histogram eşitleme işleminde toplam piksel sayısı

S : Yapay sinir ağlarında toplam fonksiyonu

u_i : Yapay sinir ağları matematiksel modelinde giriş fonksiyonu

w_i : Yapay sinir ağları matematiksel modelinde ağırlıklandırma faktörü

Q : Yapay sinir ağları matematiksel modelinde çıkış fonksiyonu

Ψ(S) : Yapay sinir ağları matematiksel modelinde aktivasyon fonksiyonu

φ : Yapay sinir ağları matematiksel modelinde aktivasyon fonksiyonu 2

θ : Yapay sinir ağları matematiksel modelinde eşik değeri

Kısaltmalar

ADALINE	: Adaptive Linear Element
APK	: Android Application Package File
ART	: Adaptive Resonance Theory
Bit	: Binary Digit
CCD	: Charge Coupled Device
CIE	: International Commission on Illumination
GIMP	: GNU Image Manipulation Program
GLCM	: Gray Level Cooccurence Matrix
GUI	: Graphical User Interface
HSB	: Hue, Saturation, Brightness
HSI	: Hue, Saturation, Intensity
HSV	: Hue, Saturation, Value
LMS	: Least Mean Squares
LoG	: Laplacian of Gaussian
LVQ	: Learning Vector Quantization
MADALINE	: Multiple ADALINE
MLP	: Multi Layered Perceptron
MP	: Megapiksel
NDK	: Native Development Kit
NTSC	: National Television Standards Committee
OpenCV	: Open Source Computer Vision Library
PCA	: Principal Component Analysis
RAM	: Random Access Memory
RBF	: Radial Basis Function
RGB	: Red, Green, Blue
RTFA	: Radyal Tabanlı Fonksiyon Ağı
SDK	: Software Development Kit
SOM	: Self Organizing Map
SVM	: Support Vector Machine
TKA	: Tek Katmanlı Algılayıcı
YCbCr	: Luminance- Chroma Blue- Chroma Red
YSA	: Yapay Sinir Ağları

1. GİRİŞ

Tarım, istihdam, ihracat ve milli gelire yaptığı katkı ile önemli bir sektördür. Türkiye'de dört mevsimin yaşanması, tarım arazilerinin genişliği ve Türkiye'nin orta kuşakta bulunması tarımsal mahsüllerin çeşitliliği açısından büyük bir etkendir. Türkiye arazisinin % 55,9'u 1000 metreden fazla yüksekliğe sahiptir. %15'ten fazla eğime sahip alanlar ülke topraklarımızın % 62,5'ini oluşturmaktadır. 1. 2. ve 3. Sınıf kaliteli topraklar % 24,5 oranındadır ve bu arazilerin % 90'ı tarım arazisidir. 2017 yılı için, meyvelerden üzüm 4.2 milyon ton, elma 3 milyon ton, zeytin 2.1 milyon ton, portakal 1.95 milyon ton, fındık 675 bin ton, çay 1.3 milyon ton üretilmiştir (TÜİK, 2018).

Verilere göre, Türkiye ürün çeşitliliği ve üretimi bakımından tarım potansiyeli oldukça yüksek bir ülkedir. Üreticinin, ürün üretim safhalarını plansız şekilde ilerletmesi ve teknolojik gelişmeleri tarım uygulamalarına adapte etmekte zorlanması veya uzak kalması sonucunda, artan üretim miktarına karşın pazar ihtiyaçlarına cevap verememekte ve kaliteli, güvenilir ürün çıktılarını elde edememektedir. Ayrıca, üretici hasat sonucunda ürününün kalitesini ve miktarını tam olarak bilememekte, yaklaşık olarak tahminde bulunmaktadır.

Türkiye'de meyvecilik sektörü üretimi yıllar itibariyle düzenli artış göstermektedir. Türkiye'de üretimi yapılan meyveler uluslararası ticarette de önemli bir yere sahiptir. Buna karşın değişen pazar isteklerine uygun nitelikte ürünlerin kalite sıkıntısı devam etmekte ve ürünlerin temini konusunda güçlükler yaşanmaktadır. Meyve yetiştiriciliğinde, sınıflamada, korunmasında, lojistik ve pazarlamada gelişmiş tekniklerin kullanılması sonucu birçok üründe gelişmiş ülkeler seviyesine ulaşılmıştır; fakat elde edilecek üretimin gerçek değeri ve hasat miktarı mevsim şartlarına, üreticinin bilgi ve yeteneklerinin farklılıklarına göre büyük ölçüde değişmektedir.

Gerçekleştirilen çalışmada, ülkemizde bu derece önemi olan tarım ve meyvecilik sektörünün önemi vurgulanarak, henüz dalında iken toplanmamış meyve ağaçlarının toplu bir şekilde satılması geleneğine farklı ve daha önce önerilmemiş bir yaklaşım getirilmiştir. Üretici yıl boyunca elde ettiği ürünleri bazı durumlarda henüz olgunlaşmadan veya olgunlaşmış halde alıcıya satabilmektedir. Bir bahçeden elde edilecek toplam ürün miktarı alıcı ve satıcı arasında bilgi birikimine bağlı olarak belirlenmektedir. Ayrıca, üretici bazı durumlarda ürününü erken pazarlayarak kendi açısından bir avantaj yakalamak istemekte; fakat kümülatif ürün çıktı bilgisine tam hakim olamadığı için bu durumdan faydalanamamaktadır. Ürünleri pazarlama olanaklarının

zorluğundan, üreticinin ve alıcının arasındaki tam güven oluşması kavramlarından yola çıkılarak, Antalya Belek konumunda bir köyde bulunan portakal bahçelerindeki ağaçların meyve yükü hesaplaması, görüntü işleme ve yapay sinir ağları teknikleri kullanılarak matematiksel boyutta gerçekleştirilmiştir.

Alınan görüntüler, bilgisayar ortamına aktarılarak, sayısal veriler ile ortalama bir hasat üretim miktarı belirlenmiştir. Tespit edilen meyvelerin alan, sayı, renk bilgilerine göre kilogram tahmini yapan yapay sinir ağları tekniği kullanılmıştır. Böylece hasat edilen ürünlerin kalitesi ağaç başına ölçülebilmektedir. Sonrasında çalışmanın gerçek hayata entegre hale getirilmesi hedeflenerek Android işletim sistemlerinde çalışan bir uygulama geliştirilmiştir.

1.1. Amaç

Tez kapsamında gerçekleştirilen çalışma ülke ekonomisinde önemli bir yeri olan tarım sektörü için bir yeniliği ifade etmektedir. Akıllı bir cep telefonu ile elde edilen ağaç görüntüleri üzerindeki meyve miktarını, alan ve renk bilgileri elde edilerek toplam çıktı miktar, verim hesaplaması gerçekleştirilmektedir. Elde edilen toplam meyve yükü ile ilgili sonuçlar, üreticinin ve tüketicinin kullanmasına imkan vermek adına Android işletim sistemi ile çalışan cihazlara uyumlu hale getirilmiş olup hem tarım sektörüne hem de üretici ve tüketiciye katkıda bulunulması hedeflenmiştir.

1.2. Tezin Kapsamı

Günümüzde görüntü işleme tekniklerinin yaygınlaşmasıyla beraber geliştirilen uygulamalar mühendisliğin ve diğer alanların kullanmasına imkan vermektedir. Bu tez kapsamında tarım alanında matematiksel olarak hesaplanması gerçekleştirilmemiş olan ağaç meyve yükü hesaplama önerisi geliştirilmiş olup çeşitli görüntü işleme ve yapay sinir ağı teknikleri kullanılmıştır. Yapılan tez çalışması Android ve Matlab ortamlarında geliştirilerek uygulama haline getirilmiştir.

Tezin ikinci bölümünde görüntü işleme ile ağaç meyve yükünün hesaplanması uygulamasında kullanılan görüntü işleme konusu hakkında bilgi verilerek literatürde yer alan görüntü işlemenin yöntemleri ve temel teknikleri konularına yer verilmiştir. Üçüncü bölümde yapay sinir ağları literatür özetine yer verilmiş olup, yapıları, türleri ve çalışma prensipleri ele alınmıştır. Tezin dördüncü bölümünde uygulamaya ilişkin daha önce

çalışılmış olan benzer literatür örnekleri kaynak araştırması başlığı altında ele alınmıştır. Beşinci bölümünde görüntü işleme ile ağaç meyve yükü hesaplanması uygulamasında kullanılan Android studio, Matlab ortamı ve OpenCV kütüphanesi gibi yazılımsal materyaller hakkında bilgiler ifade edilmiş olup ele alınan yöntemler detaylı şekilde ve örneklenerek ifade edilip program çıktıları verilmiştir. Tezin altıncı bölümünde gerçekleştirilen uygulamaların sonuçlarına yer verilmiştir. Yedinci bölümde uygulamanın sonuçları özetlenerek, çalışmanın geliştirilmesi için ilerleyen safhalarda uygulamaya entegre edilebilecek öneriler açıklanmıştır.

2. KAYNAK ARAŞTIRMASI

Günümüzde görüntü işleme uygulamaları insan yaşamını kolaylaştırma, çeşitli alanlarda karşılaşılan problemlere hızlı ve pratik çözüm üretebilme, sayısal değerler üzerinde işlem yapabildiği için güvenilirlik teşkil etme, ürün kalitesine ve verimine katkısı, çevreye karşı duyarlılık gibi özelliklerinden dolayı hayatımızın hemen hemen her noktasında sıklıkla kullanılmaktadır. Tarım olarak nitelendirdiğimiz bahçe ve tarla sektöründe de görüntü işleme uygulamaları kullanıcıya sunduğu hız ve pratiklik sayesinde son zamanlarda hızla gelişmekte ve literatürde yapılan araştırmaların sayısı artmaktadır.

Görüntü işleme uygulamaları tarım alanında, kök, yaprak, meyve gelişiminin izlenmesi, meyvelerde ve sebzelerde renk analizi ile sınıflandırılma yapılması ile nitelik çıkarımı, yaprak, meyve, sebze alanının ölçümü, yabancı ve zararlı otların belirlenmesi, meyve ve sebze hasadının robotikleşmesi gibi çeşitli amaçlarla kullanılmaktadır.

Thomas ve Connoly (1986) RGB renkli görüntüleri kullanmış, RGB'nin görüntü işleme işlemleri için etkin olmadığını fakat bu sinyallerin daha etkin kullanılabilecek diğer renk modellerine ve özellikle Hue moduna çevrilebileceğini bildirmişlerdir.

Morrisey (1988), HSI renk özelliklerinin insan gözünün gördüğü renklere çok yakın sinyallere sahip olduğunu ve bu özelliklerin aydınlatmadan kaynaklanan etkilerden bağımsız olduğunu fakat renklerle ilgili değişkenlere bağımlı olduğunu belirtmiştir (Tonguç, 2007).

Konda ve ark. (2000) 'ı ise bir Japon meyvesi olan Iyokan isimli tatlı bir tada sahip kırmızımsı portakalların dış özellikleri yerine iç özellikleri inceleyip sınıflandırma uygulaması gerçekleştirmişlerdir. Bunun sebebi, bazı bölgelerde bu meyveye farklı isimler verilmekte olup, bazılarına göre meyvenin orta büyüklükte kırmızı ve parlak bir yüzeyi olduğuna inanılırken bazı bölgelerde buna inanılmamaktadır. Bu belirsizliği ortadan kaldırmak için meyve içerisindeki şeker ve asit içeriği değerlendirilerek sınıflama işlemi gerçekleştirilmiştir. Çalışmanın sonucunda daha fazla eğitim seti kullanılmasıyla uygulanan yapay sinir ağının performansının geliştirilebileceği belirtilmiştir.

Hannan ve Burks (2004) otomatik naranciye hasat etme üzerine bir araştırma önermişlerdir. Bu araştırmada CCD kamera sistemlerinin, sensörlerin, görüntü işleme yöntemlerinin gelişmesinin ve robot kol teknolojisindeki yeniliklerin önemini belirtmişlerdir. Narenciye hasat etmek için öncelikli olarak meyvelerin özelliklerini belirleyerek görüntüdeki meyveler saptanmıştır. Sonrasında robotik olarak

gerçekleştirilen hasat ile mekanik olarak gerçekleştirilen hasat karşılaştırması yapmışlardır. Robotik hasat sonucunda daha az emek sarfedilerek daha fazla üretkenlik kazancı olduğu ve iş hacmi başına daha yüksek mal elde edilmesi sebebiyle robotik hasat için kullanılan sistemin esnek olduğu ifade edilmiştir. Mekanik hasat için kullanılan sistemin esnek olmaması sonucunda işlem hacmi olarak düşük bir üretim sağlanmış ve yüksek emek karşında az üretkenlik olduğu çalışma sonucunda ortaya çıkarılmıştır (Kahya ve Arın, 2014).

Zhao ve ark. (2009) tarafından meyvelerin kalitesi gerçek zamanlı görüntü işleme teknikleri kullanarak belirlenmiştir. Geliştirdikleri bir makine yardımıyla meyvelerin doku özellikleri, boyut miktarları, renk analizleri tespit edilmiştir. Meyve çeşidi olarak armut üzerinde çalışma gerçekleştiren Zhao ve arkadaşları çalışma ile diğer meyve ve sebzelerin (elma, salatalık, domates, şeftali, mantar vb.) de kalite standartlarının saptanabileceğini öne sürmüşlerdir.

Hannan ve ark. (2010)'ı turuncu meyveleri ağaçtan ayırma çalışması gerçekleştirmiş olup, görüntü işleme uygulamalarında sıklıkla karşılan ortam aydınlık şiddeti problemine çözüm önerisi getirmişlerdir. Meyvelerin segmentasyonu şekil ve renk analiz teknikleri ile çevre tabanlı bir saptamadan oluşmuştur. Geliştirilen çalışmada değişen aydınlatma koşullarının iyileştirilmesi ile meyvelerin saptanma oranları %90' ın üzerine çıkabilmiştir.

Kurtulmuş ve Vardar (2013) genç şeftali meyvelerini dalında iken saptamaya yönelik bir çalışma gerçekleştirmiş olup, bu işlem için bir eğitim seti oluşturulmuş ve iki farklı görüntü tarama metodu geliştirilmiştir. Meyvelerin arka plandan ayrılması için renk tabanlı bir segmentasyon işlemi uygulanmıştır. Araştırmada kullanılan görüntü tarama yöntemi, görüntü üzerinde kayan bir alt pencere ile yatay ve düşey yönde görüntünün taranmasıdır. Bu yöntem sayesinde belirlenen adım mesafesi ile birbirine çok yakın olan meyvelerin tek bir meyve olarak algılanmasının önüne geçilmesi hedeflenmiştir. Önerilen ilk görüntü tarama metoduna göre çeşitli sınıflandırıcıların başarısı kıyaslanmış olup en yüksek başarının %77 ile YSA algoritmasına, önerilen ikinci tarama metoduna göre de %81 oranı ile yine YSA algoritmasına ait olduğu gözlemlenmiştir. Çalışmada benzer olarak aydınlanma koşullarıyla ilgili sıkıntılar ile karşılaşılmış olup, genç meyvelerin tamamı tespit edilememiştir. CCD kamera ile alınan görüntüler üzerinde yapılan bu çalışmanın başarı oranı ile umut vaadettiği görülmektedir.

Kuncan ve ark. (2013)'nın yaptığı çalışmada görüntü işleme yöntemleri kullanılarak gerçek zamanlı çalışan bir sistem üzerinde zeytinlerin renkleri belirlenerek

sınıflandırma işlemi gerçekleştirilmiştir. Zeytinleri renklerine göre ayırt etme metotlarından HSV renk uzayında ve RGB renk uzayında Öklid uzaklığı metotu ve RGB renk uzayında Mahalanobis uzaklığı kullanılmıştır. Zeytinlerin öz niteliklerine göre sınıflandırma işlemi uygulanmış olup %97'lik başarı oranı ile en başarılı metodun Mahalanobis uzaklığı metotu olduğu gözlemlenmiştir. Bunun sebebi olarak bu metodun, nesnelerin standart sapma değerlerini kullanarak işlem yapması gösterilmiştir.

Payne, Walsh ve ark. (2013) görüntü analiziyle mango meyvesinin mahsül verimini segmentasyon yöntemi kullanarak uygulamışlardır. Çalışmalarında RGB ve YCbCr renk uzayları kullanarak segmentasyon gerçekleştirmişlerdir. Veri setini, aydınlatma şiddetine göre 3 farklı durumda oluşturmuşlardır ve ağaç başına ürün sınıflamasını düşük, orta ve yüksek verimli olarak belirlemişlerdir. En son aşamada toplam verim hesabı gerçekleştirmişlerdir. Çalışma güneş ışığının doğrudan geldiği görüntüler üzerinde ve ürün miktarının çok fazla olduğu ağaçlar üzerinde düşük başarı göstermiştir.

Tatlı ve Üncü (2014) mobil cihazların görüntü işleme uygulamalarında güçleri ve depolama kapasitelerinin yetersiz kalması problemine bir çözüm önerisi sunmuşlardır. Bunun için el yazısı tanıma sistemini yapay sinir ağları kullanarak uzun boyutlu el yazısı metinleri dönüştürme denemesi gerçekleştirmişlerdir. Mobil cihaz ile alınan metin belgesi fotoğrafları web servisine aktarılmıştır. Daha sonra web servisinden alınan görüntüler sunucuya işlenmek üzere gönderilmiştir. İşlenen görüntüler tekrar web sitesine aktarılarak buradan mobil cihaza iletilmiştir. Mobil cihazdan gönderilen veriler ile kullanıcının seçtiği veriler tekrar birleştirilip yeniden web servisine aktarım yapılmıştır. Alınan bu veriler sunucuya aktarılarak sunucuda istenilen rapor hazırlanmıştır. Son aşamada ise rapor bir pdf haline getirilerek kullanıcının mobil cihazına aktarılıp kullanıcıya gösterilmiştir. Sonuç olarak gerçekleştirilen öneri mobil cihazın performansına bakılmaksızın tüm akıllı cihazlarda kullanılmasını sağlamıştır. Çalışmanın dezavantajının, el yazısı tanıma aşamasında kullanılan yazılımlar sebebiyle Android 2.1 ve üzerini kullanan cihazlar için uyumlu olduğu gözlemlenmiştir.

Kanimozhi ve Malliga (2017) renk kodlama tekniği kullanarak olgunlaşmamış ve olgunlaşmış meyveleri sınıflandırma çalışması gerçekleştirmişlerdir. Turunç meyvesi görüntü alındıktan sonra ön işlemlerden geçilirek Otsu eşikleme yöntemi ile arka plandan ayrılmıştır. Daha sonra kenar bulma algoritması uyguladıkları görüntüleri morfolojik işlemlerden geçirdikten sonra HIS renk uzayına dönüştürmüşlerdir. Daha sonra dokusal analizleri yaparak sınıflama işlemi gerçekleştirmişlerdir. Çalışmanın sonucunda

olgunlaşmış meyvelerde daha yüksek başarılı sonuç alınmıştır. Aydınlatma faktörü çalışmayı olumsuz etkilemiş olup olgunlaşmamış turunçlarda başarı sağlanamamıştır. Ayrıca kameranın konumunun da çalışmada önemli bir faktör olduğu vurgulanmıştır.

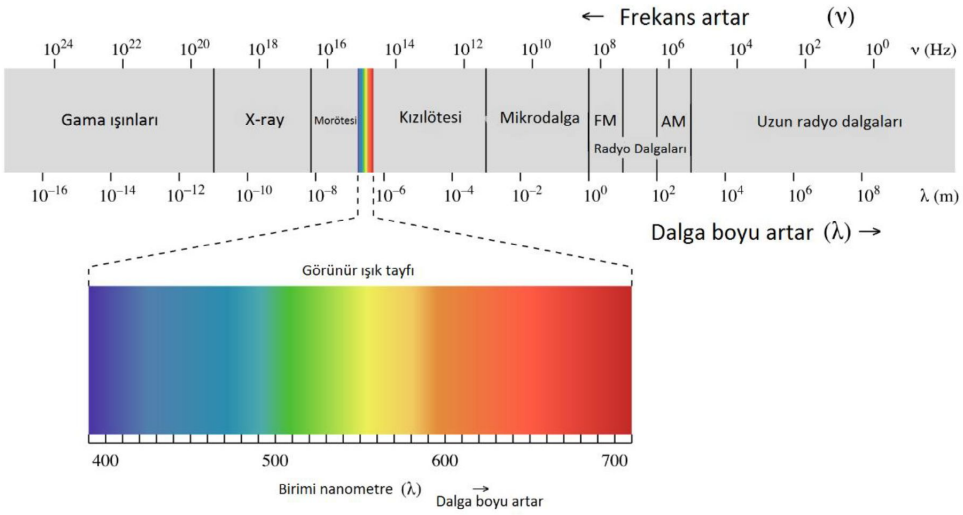
Yaşar ve Akdemir (2017) Türkiye'de üretimi yüksek bir orana sahip portakal ağaçlarının verimi üzerine yapay sinir ağları ve görüntü işleme teknikleri ile çalışma gerçekleştirmişlerdir. Gerçekleştirilen çalışmada renkli görüntüler üzerinde ışık dengesi sağlamak için görüntüleri yaklaşık aydınlık seviyelerine göre sınıflandırmışlar; sonrasında ayrılan görüntüler üzerinde görüntü iyileştirme teknikleri uygulamışlardır. elde edilen görüntüleri yapay sinir ağları algoritmasına göre yarısını eğitim yarısını test sınıfında kullanmış olup çarpraz doğrulama yöntemi ile ağaç meyve yükü hesaplaması gerçekleştirmişlerdir. Uygulama %89.8 başarı oranı elde etmiştir.

3. GÖRÜNTÜ İŞLEME KAVRAMLARI

Bir nesnenin, kişinin veya sahnenin, ayna, mercek ya da kamera gibi optik bir cihaz yardımıyla ürettiği görsel sunum sonucu görüntü meydana gelir. Başka bir deyişle görüntü, *f(x, y)* şeklinde bir fonksiyon olarak tanımlanabilir. Burada *x* ve *y* uzaysal (spitial) koordinatlar, *f* in herhangi bir *(x, y)* koordinatındaki değeri de görüntünün o noktadaki şiddeti olarak adlandırılır (Gonzalez ve Woods, 2002).

Görüntü işleme bakımından insan algılama sistemi; görüntü yakalama, sınıflandırma ve analiz etme konularında en karmaşık olan sistemdir. İnsan görme sistemi gözler ile başlar. Birer algılama sistemi olan gözlerimiz ile ışığın çok kanallı ve pankromatik dalga boyları algılanır. Gözlerimizle görülebilen alandaki elektro manyetik dalgalar algılanır ve beynimiz ile yorumlanarak görüntü haline dönüştürülür. Bir başka deyişle gözümüz bir fotoğraf makinasının işleyişi gibi çalışır ve beynimizin görme mekanizması da karmaşık bir sayısal görüntü işleme sistemi olarak düşünülebilir.

Görülebilen ışık spektrumun tanımı; gözümüzün görebileceği elektro manyetik dalga boyu aralığı olarak verilebilir (Şekil 3.1). Spektrum uzunluk ölçme birimleri ile ölçülebilen periyodik davranış gösteren enerji dalgalarını temsil eder. Görülebilen ışık tayfına ait dalga boyları 0.4 mm-0.7 mm arasındadır (Jähne, 1997).



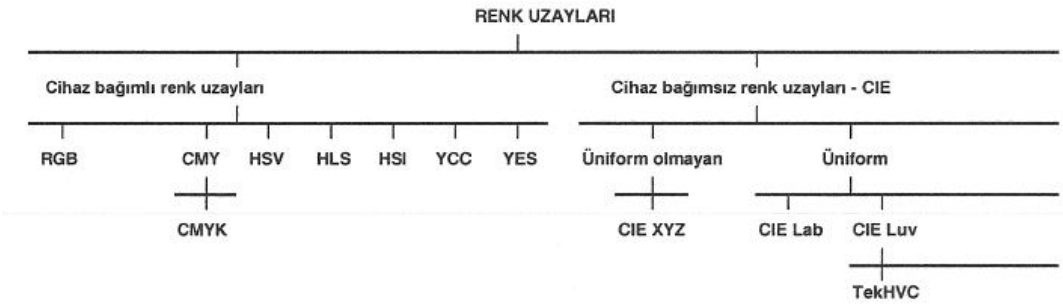
Şekil 3.1. Elektromanyetik dalga boyu aralığı

3.1. Renk Kavramı

Bir rengi kavram olarak ele alacak olursak, ışığın farklı dalga boylarında gözün retinasına ulaşması ile oluşan bir algılama olarak tanımlayabiliriz. İnsanlar tarafından renklerin algılanması, bir ışık kaynağından çıkan ışığın objeler tarafından yansıtılmasına ve objenin göz aracılığıyla beynimize iletilmesi sayesinde gerçekleşmektedir. Renklerin algılanması ışığın objeler üzerine çarpması ve kısmen soğurulup yansıması nedeniyle çeşitlilik göstermektedir. Bu sebeple oluşan farklılıklar renk tonu veya renk olarak adlandırılır. Tüm dalga boyları aynı anda göze ulaşırlar ise beyaz renk, hiçbir dalga boyu ulaşmazsa siyah renk olarak algılanmaktadır (Metlek ve ark., 2009).

Renk uzayları renkleri tanımını oluşturmak için oluşturulan matematiksel modellerdir. Renkmetri bilimine göre bir rengin renk olabilmesi için birbirinden bağımsız üç değişkeni olması gereklidir. Bu sebeple renk uzayları üç boyutlu olarak tasarlanmışlardır. Renklerin renk uzayındaki konumları belirlenen üç değişkene göre hesaplanır. Renk uzaylarının matematiksel modelleri oluşturulurken her bir renk uzayının kendine has biçimde renk oluşturma standartları geliştirilmiştir. Dijital görüntüler üzerinde matematiksel işlemler yapabilmek için çeşitli renk uzaylarına ihtiyaç duyulabildiğinden dolayı renk uzayları birbirinden farklı renk uzaylarına doğrusal veya doğrusal olmayan metodlar ile dönüştürülebilmelidirler. Sayısal görüntüler farklı renk uzaylarına dönüştürülerek matematiksel işlemler gerçekleştirilebilir.

Renk uzayları cihaz bağımlı ve cihaz bağımsız renk uzayları olarak sınıflandırılmıştır. Cihaz bağımlı renk uzaylarında renkler cihazın özelliklerine bağlı olup, cihaz bağımsız renk uzayları CIE (Uluslararası Aydınlatma Komisyonu) tarafından geliştirilen ve tüm renk ölçümünü sağlayan renk uzaylarıdır. Cihaz bağımsız renk uzayları renkmetride kullanılan uzaylardır. Şekil 3.2'de sıklıkla kullanılan renk uzayları verilmiştir (Gül, 2016).



Şekil 3.2. Yaygın şekilde kullanılan renk uzayları

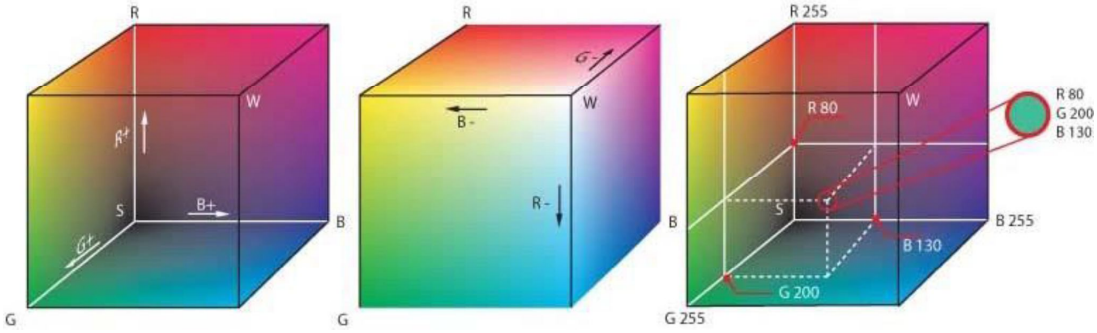
3.1.1. RGB renk uzayı

RGB (red, green, blue) renk uzayı ingilizce 'Red' 'Green' 'Blue' kelimelerinin baş harfleri kullanılarak oluşturulan renk uzayıdır. Türkçe olarak sırasıyla 'Kırmızı' 'Yeşil' 'Mavi' ana renklerini temsil etmektedir ve çalışmalarda en çok kullanılan renk uzayıdır. Bu renkler %100 oranında karıştırıldığında beyaz renk, %0 oranında karıştırıldığında siyah renk elde edilmektedir. 0,4-0,5 mm dalga boyu elektromanyetik spektrumda mavi rengi temsil ederken; 0,5-0,6 mm dalga boyu yeşil rengi; 0,6-0,7 mm dalga boyu ise kırmızı rengi temsil etmektedir.

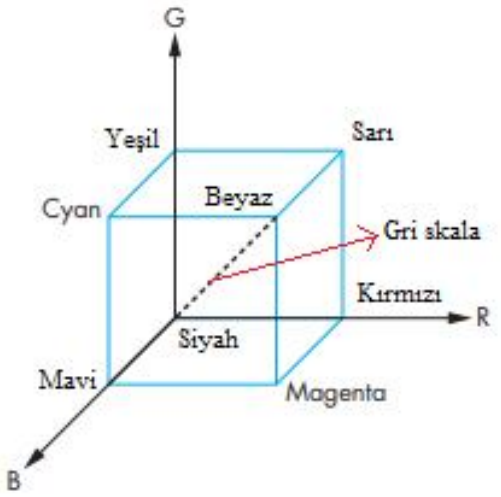
İnternet ortamında da kullanımı en çok tercih edilen renk sistemi RGB renk sistemidir. Bunun sebebi, 1953'te ilk fotoğraf makinesi Polaroid'de ve sonrasında televizyonda Sony'de ilk kez RGB renk uzayı kullanılmıştır. Günümüzde de televizyon ekranlarında, tarayıcılarda ve manuel fotoğraf makinelerinde standart olarak RGB renk uzayı kullanılmaktadır.

Cihaz bağımlı bir renk modeli olan RGB renk uzayında, renk değerleri ve bu değerlerin kırmızı, yeşil ve mavi renk seviyelerine verdikleri tepkiler üreticiden üreticiye değiştiği için RGB renklerin seviyeleri farklılık gösterebilir. Bazı durumlarda aynı cihaz içerisinde bile verilen tepkiler değiştiği için her cihaz RGB renklerin seviyelerini farklı algılarlar veya farklı üretirler. Bu sebepten dolayı RGB renk uzayına ait renk seviyeleri farklı cihazlarda aynı renk tonuna sahip olabilir veya olmayabilir.

RGB renk uzayı, eksenlerinde kırmızı, yeşil ve mavi olan üç boyutlu bir küp ile temsil edilmektedir (Şekil 3.3). Orijinde siyah bulunurken karşı ucunda beyaz bulunur. Grilik skalası, siyah ile beyazı birleştiren doğru olarak temsil edilir. Şekil 3.4'te gri skala ile gösterilen RGB renk uzayı gösterimi verilmiştir (Angel ve Shreiner, 2011).



Şekil 3.3. RGB renk uzayı gösterimi 1 (Anonymous, 2017)



Şekil 3.4. RGB renk uzayı gösterimi 2 (gri skala gösterimi)

Bazen, RGB görüntülerin grilik skalasına dönüştürülmesi gerekmektedir. Bu işlem için Denklem 3.1 kullanılır (Crane, 1997).

$$\text{Grilik Skalası Şiddeti1} = 0.299\ R + 0.587\ G + 0.114\ B \tag{3.1}$$

Bu denklem bir NTSC standartıdır. Bir başka sıkça kullanılan RGB grilik skalası dönüşümü ise basitçe ortalama almaktır (Denklem 3.2) (Demirel, 2008).

$$\text{Grilik Skalası Şiddeti2} = 0.333\ R + 0.333\ G + 0.333\ B \tag{3.2}$$

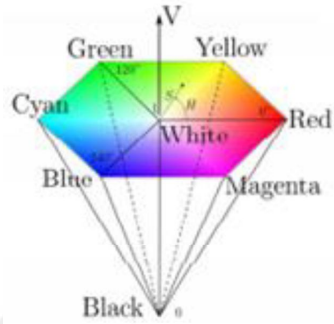
3.1.2. HSV renk uzayı

HSV (Hue, Saturation, Value) renk uzayı içerdiği renklerin RGB değerlerine göre ifade edilmeyen bir renk uzayıdır. Renklerin sırasıyla renk özünü, doygunluğunu ve parlaklığını ifade etmektedir. HSB renk uzayı olarak da adlandırılır.

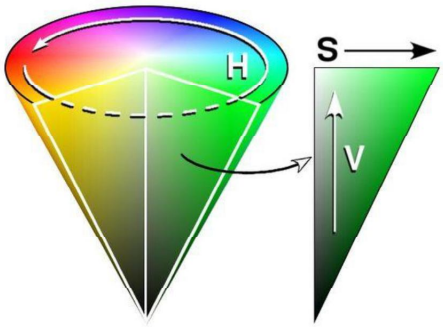
Hue olarak isimlendirilen renk özü, rengin baskın dalga uzunluğu yani renklerin açısal değerleridir. 0° - 360° arasında değer almaktadır. Bazı uygulamalarda ise 0-100 arasına yayılır. Doygunluk olan saturation değeri rengin canlılığını belirlemektedir. Yüksek doygunluktaki kırmızı bir renk koyu kırmızı, bordo gibi renkleri meydana getirirken düşük doygunluktaki kırmızı renk açık pembe, ten rengi gibi renklere neden olur. 0-100 arasında değer almaktadır. Parlaklık olan value değeri ise parlaklığın derecesidir. Rengin içerisindeki beyaz oranını (aydınlığını) belirler. Parlaklık değeri 0-100 arasında değişmektedir (Gonzalez ve Woods, 2002).

1978 yılında Alvy Ray Smith tarafından tanımlanmış olan RGB renk uzayına göre insan görme mekanizmasına daha benzer bir yapı oluşturmak olan bu renk uzayı, RGB renk uzayından doğrusal olmayan bir dönüşüm ile elde edilir. HSV uzayı cihaz bağımlı bir renk uzayıdır. Bunun anlamı RGB renk uzayında olduğu gibi bu uzayda da renk seviyeleri, rengi üreten cihaza göre değişim gösterebilir. Cihaz bağımsız renk gösterimi için “CIE Lab” veya “CIE Luv” ile “CIE XYZ” renk uzayları kullanılmaktadır. HSV renk uzayının gösterimi Şekil 3.5’te verilmiştir (Gonzalez ve Woods, 2002).

HSV renk uzayındaki renk özü değeri sabitlenip, doygunluk ve parlaklık değerleri değiştirilerek aynı rengin farklı doygunluk ve parlaklık değeri elde edilmiş olur. RGB renk uzayında eşik değeri belirlenerek filtreler ile işlem yapmak yerine bu metot sıklıkla tercih edilmektedir. Şekil 3.6’da verilen HSV uzayının konik şeklindeki gösterimi ile bu özelliği görülmektedir. Bu özelliği sebebiyle HSV renk uzayı renkler açısından işlem yapılan uygulamalarda sıklıkla kullanılmaktadır.



Şekil 3.5. HSV renk uzayı gösterimi



Şekil 3.6. HSV renk uzayı konik şeklinde gösterimi

Rengin parlaklığının, tonunun gerekli olduğu durumlarda RGB uzayından HSV uzayına dönüşüm yapmak gereklidir. RGB uzayından HSV uzayına dönüşüm ve HSV uzayından RGB uzayına dönüşüm işlemleri doğrusal olmayan bir bağlantı ile gerçekleştirilir. İlk olarak RGB değerleri 0 ile 255 arasından 0-1 arasına normalize edilip aşağıda verilen denklemler kullanılarak dönüştürme işlemi gerçekleştirilir.

Normalizasyon işleminde maksimum değer 255 alınırken; R, G, B değerleri seçilen koordinattaki değerler alınmaktadır (Denklem 3.3). RGB renk uzayından HSV renk uzayına dönüşüm işlemlerinde kullanılan denklemler Denklem 3.3, 3.4, 3.5, 3.6, 3.7 ile verilmiştir.

$$R' = R/Rmax,\ G' = G/Gmax,\ B' = B/Bmax \tag{3.3}$$

$$H = \begin{cases} \theta, \ if \ B' \leq G' \\ 360 - \theta, \ if \ B' > G' \end{cases} \quad (3.4)$$

$$\theta = \cos^{-1}\left(\frac{\frac{1}{2}[(R\prime - G\prime) + (R\prime - B\prime)]}{[(R\prime - G\prime)^2 + (R\prime - B\prime)(G\prime - B\prime)]^{\frac{1}{2}}}\right) \quad (3.5)$$

$$S = 1 - 3/(\mathrm{R}' + \mathrm{G}' + \mathrm{B}') \ [\min(R', G', B')] \quad (3.6)$$

$$V = 1/3(R' + G' + B') \quad (3.7)$$

HSV renk uzayından RGB renk uzayına geri dönüşüm işlemleri Denklem 3.8,3.9 ve 3.10 ile verilmiştir.

$$B = V(1 - S) \quad (3.8)$$

$$R = V\left[1 + \frac{S * cosH}{\cos(60^{\circ} - H)}\right] \quad (3.9)$$

$$G = 3V - (R + B) \quad (3.10)$$

3.2. Görüntü Çeşitleri

3.2.1. Analog görüntü

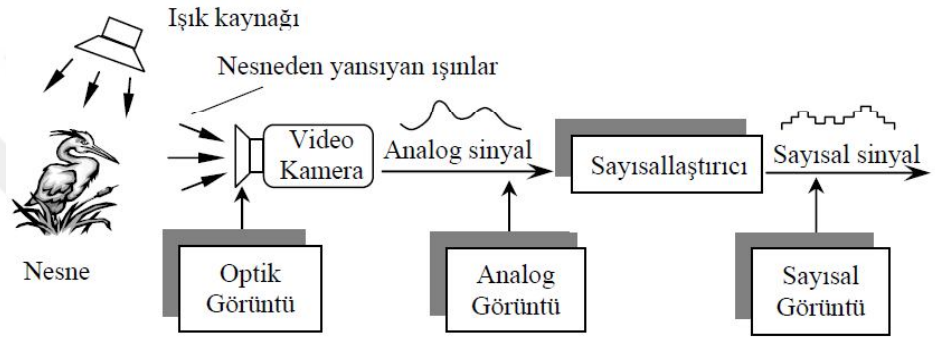
Saydam olmayan opak nesnelere bir ışık kaynağından çıkan ışık hüzmelerinin objeden yansıması sonucu yansıyan ışığın iki boyutlu bir yüzeye düşürülerek uzay düzleminde iki boyutlu ve sürekli (analog) bir iz düşümü meydana gelmesi olayı analog görüntüyü oluşturur. Film şeridi kullanan analog fotoğraf makinelerinde bu iz düşüm kimyasal olarak negatif üzerine işlenir. Parlaklık seviyesi kimyasalın aşınma miktarına göre belirlenir. Mikroskobik olarak bakıldığında ayrık olan bu görüntü makro boyutta insan gözünden daha yüksek bir çözünürlük sunar ve bu sebepten dolayı insan gözü sürekli (analog) bir görüntü olarak algılar (Yıldız, 2010).

Bir başka deyişle, sürekli (analog) görüntüde belirli bir maksimum ve minimum aralığında *f(x, y)* fonksiyonunu oluşturan *x* ve *y* ayrık olmayan, sürekli olarak değişen reel değerler içeriyor ise bu görüntü analog görüntüdür. Sürekli görüntüde bir nesneye ne kadar yakından bakarsak bakalım (örneğin bir mikroskopla) görüntü üzerinde görüntüyü oluşturan tüm renklerin bulunduğunu görebiliriz. Sayısal bilgisayarlar yapıları bakımından sürekli fonksiyonları, parametleri ve değerleri işleyemediklerinden dolayı bu fonksiyonların sayısallaştırılması gerekir (Gonzalez ve Woods, 2002). Sayısallaştırma

işlemi sonucunda da bilgisayarların algılayıp üzerlerinde işlem yürütebildikleri dijital görüntüler meydana getirilir.

3.2.2. Dijital görüntü

Bir obje tarafından yansıyan ışığın (analog sinyal) herhangi sayısal dönüştürücü tarafından belirlenen elektromanyetik aralıkta algılanarak sayısal sinyal olarak dönüştürülmesi işlemi analog görüntünün sayısallaştırılması olan dijital görüntü elde edilmesidir. Şekil 3.7'de görüntünün oluşum aşaması verilmiştir.



Şekil 3.7. Sayısal görüntünün oluşum aşamaları (Yaman, 2000)

Fotoğraf makinesi veya video kaydedici tarafından algılanan görüntü iki boyutlu bir sinyal kaydıdır. Fotoğraf gibi göz ile görülebilen bir şekilde olabileceği gibi, manyetik bantta yazılı bir kayıt ya da bilgisayar belleğindeki sayısal değerler olabilir. Görüntüler sürekli ya da ayrık, analog ya da sayısal veya sürekli sayısal olabilirler. Parlaklık ifadesi görüntü üzerindeki seviye ve konum olarak sürekli değişen bir değişkendir. Bu duruma karşın bilgisayarlar içinde değerler, ayrık sayılar biçiminde ifade edilmektedir. Bu sebeple bir resmin bilgisayar içerisine girebilmesi, onun belirli sayıda konum ve belirli sayıda parlaklık seviyesi bakımından ifade edilebilmesiyle mümkündür (Karakuş, 2006).

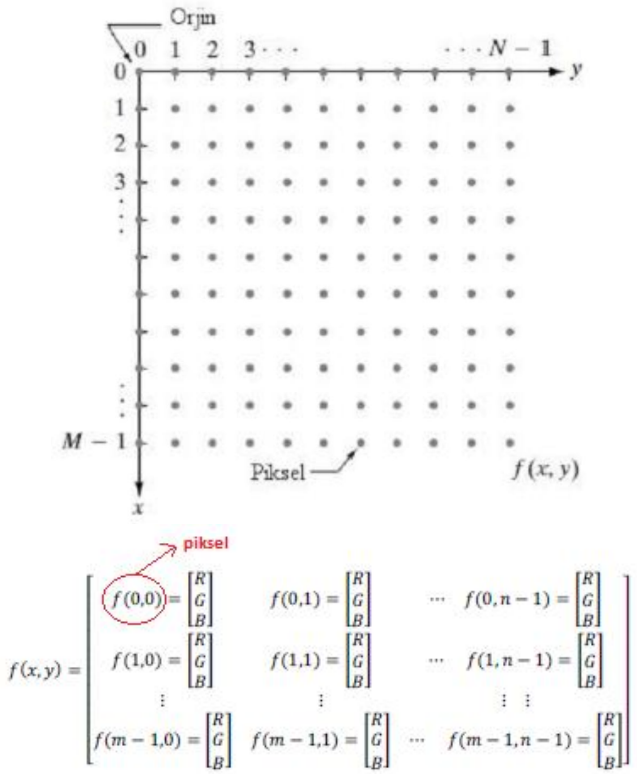
Başka bir ifade ile analog bir görüntü sürekli değerlere sahip olduğu için analog olan görüntünün dijitale dönüştürülmesinde öncelikli olarak iki boyutlu uzayın örneklenerek sürekli görüntünün ayrık hale getirilmesi gerekmektedir.

Sayısallaştırma işlemi için sürekli olan analog görüntü küçük kare şeklindeki parçalara ayrılır ve her bir parça pikselleri meydana getirir. Pikselin sahip olduğu değer, analog görüntünün piksel alanının içinde kalan ortalama parlaklık seviyesidir. Dijitalleştirmedeki sonraki adım, kuantalama olarak bilinen, ortalama parlaklık

seviyesinin ölçeklenmesi ve en yakın tam sayıya yuvarlanması işlemidir. Çoğu uygulamada 256 gri seviyesi (8-bit (binary digit)) işlemlerin yürütülebilmesi için yeterli olmaktadır. 0 değeri siyah, 255 değeri beyaz, ara değerler ise farklı gri seviyelerini temsil eder. Ortalama parlaklık seviyeleri bu mantığa göre 0–255 aralığına ölçeklenir ve sonuç olarak elde edilen reel sayıların kendilerine en yakın tam sayıya yuvarlanması ile sayısal görüntü elde edilir.

Wayne Niblack (1986) tarafından yapılan tanımda ise, görüntü piksellerden oluşan kare şeklindeki ızagaraya ayrıklaştırılır. Bunların her biri, biri sütun biri satır olmak üzere çift koordinat ile belirlenir. Sütun sayıları en solda 0'dan n'ye kadar olup, "n" görüntünün sütun sayısıdır. Benzer şekilde satırlarda en üstte olmak üzere 0'dan m'e kadar olup, "m" görüntünün satır sayısıdr. Örneğin; (100, 200) koordinatlı piksel 100. sütun ile 200. satırın kesiştiği yerde yerleşiktir.

Diğer deyişle, *f(x, y)* şeklinde ifade edilen sürekli görüntüyü (analog görüntü) ayrık örnekler bakımından gösterilmesidir ve ifade şekli *f[x, y]* şeklindedir. Dijital bir görüntünün iki boyutlu dizi şeklindeki her bir elemanı *f[x, y]*' ye "piksel" (ingilizce resim elemanları anlamına gelen "picture elements" kelimelerinin kısaltması) veya kısaca "pel" (resim elemanı) denir. Piksel sayısal görüntünün en küçük elemanı olarak ifade edilir. Dijital görüntünün iki boyutlu modeli ve matris olarak ifadesi Şekil 3.8'de verilmiştir. (Gonzalez ve Woods, 2002; Karakuş, 2006).



Şekil 3.8. Piksel (resim elemanı)

Dijital görüntü içerdiği matris değerlerine göre dört grupta incelenebilir. Bunlar;

1) İkili (binary) görüntü
2) Gri seviyeli görüntü
3) Renkli görüntü
4) İndekslenmiş renkli görüntü

3.2.2.1. İkili (binary) görüntü

Sayısal görüntü türleri içerisinde en basit olan görüntü binary görüntülerdir. Her pikseli sadece 1 bit içerir. Bu yüzden sadece iki değerli gri seviyeye (parlaklığa) sahip görüntüler olarak adlandırılırlar. 0 siyahı temsil ederken 1 beyazı temsil eder.

3.2.2.2. Gri seviyeli görüntü

İkiden daha fazla parlaklık değeri içeren görüntüler için her pikselin 1 bitten daha fazla bir koda ihtiyacı vardır. Yani, 1 bitlik binary bir görüntü sadece 0 ve 1 değerlerinden oluşuyorken, gri seviyeli bir görüntü 00, 01, 10, 11 değelerine sahip dört farklı parlaklık

seviyesi ile 2 bitten oluşmalıdır. Gri seviyeli görüntüler genellikle piksel başına 8 bit ile kodlanmaktadır. Bu görüntülerde her piksel $2^8 = 256$ farklı gri seviyeden oluşur. 0, gri seviye siyahı temsil ederken, 255 beyazı temsil etmektedir. 0 ile 255 arası değerler gri tonlara karşılık gelmektedir. Gri seviyeli görüntüler bazı durumlarda 10 bit (1024 gri seviye) veya daha fazlası 12 bit (4096 gri seviye) değerine kadar kodlanır. Bu durum görüntünün daha kaliteli görünmesini sağlamazken, belirlenebilecek hassas parlaklık aralığını arttırır. Gri seviye renk paleti Şekil 3.9'da verilmiştir.

Şekil 3.9. Gri seviye renk paleti (Gonzalez ve Woods, 2002)

3.2.2.3. Renkli görüntü

Gerçek renkli bir RGB görüntü üç boyutlu bir MxNx3 matris olarak ifade edilir. Renkli görüntü düzeylerinin her biri gri seviyeli bir görüntü gibidir; ancak R, G veya B seviyelerinde görüntülenir.

Renkli görüntülerin bilgisayar ortamında 24 bitlik veri olarak görüntülenmesi sebebiyle bu görüntülerin kırmızı, yeşil ve mavi olarak belirlenmiş aynı nesneye ait üç farklı 8 bitlik gri seviyeli görüntünün üst üste ekrana iletilmesi ile oluştuğu söylenebilir. Bir başka deyişle, farklı dalga boylarında elde edilmiş üç adet 8 bitlik görüntü sayısal bir bilgisayar görüntüleme ortamında sırası ile kırmızı-yeşil-mavi olarak üst üste getirilecek olursa renkli görüntü elde edilmiş olur (Çelik, 2014).

3.2.2.4. İndekslenmiş renkli görüntü

Renkli görüntüdeki 24 bitlik veri yaklaşık olarak 16.7 milyon renk kümesinden oluşmaktadır. İndekslenmiş renkli görüntüde amaç 16 milyonluk bir renk kümesi yerine renk paleti oluşturarak renkli görüntü verilerinin bir alt kümesini meydana getirmektir. İstenilen gri seviye değerleri belirlendikten sonra 256 adet vb. renk ile işlem gerçekleştirilebilir. İndekslenmiş renkli görüntüde bu renk alt kümesine renk paleti denilmektedir. Seçilen her piksel, renk paletindeki bir renge karşılık gelen sayıyı temsil eder. İndekslenmiş renkli görüntüler hafızada daha az yer kaplamak ve 24 bit olarak işlem

yapamayan uygulamalar için avantajlıdır. Bir görüntüde renk paleti arayarak daha kısa bir kod oluşturulabilir.

3.3. Görüntü İşleme Yöntemleri

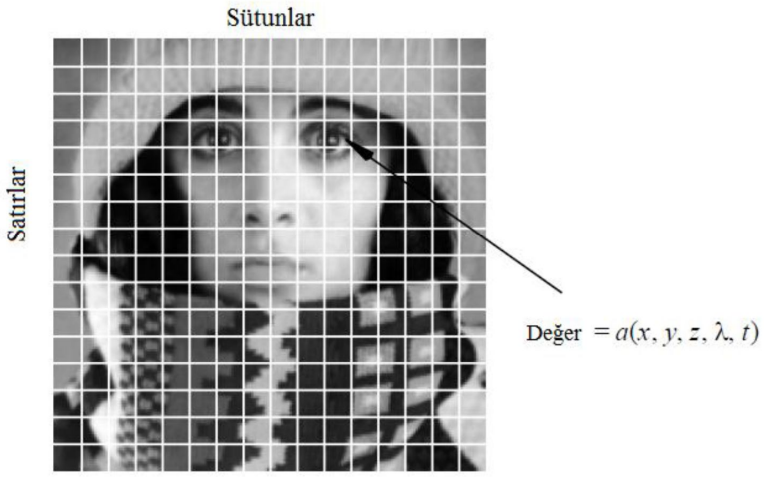
Görüntü işlemede amaç analog bir görüntüyü sayısal bir biçime dönüştürmek ve dönüştürülen sayısal resmi görsel olarak güçlendirmek, onarmak, sıkıştırmak ve istatistiksel, görsel, matematiksel olarak değerlendirmek ile yorumlamaktır. Bu amaç görüntü üzerinde yapılan işlemlerin geliştirilmesi ve uygulanması ile yürütülür. Görüntü işleme üç ana başlıkta incelenebilir. Birincisi optik, ikincisi elektronik olan analog ve üçüncüsü dijital yöntemlerdir.

Optik işleme, bir işlemi yürütmede optiklerin dizilimini kullanır. Gözlükler optik işlemenin bir yoludur (Aytan ve ark., 1993).

Görüntülerin analog olarak işlenmesi, görüntülerin elektriksel olarak işlenmesine bağlıdır. Bunun için en iyi örnek televizyon görüntüsüdür. Televizyon görüntüsünde televizyon sinyali, görüntü süresince aydınlık oluşturmak için genliği değişken olan bir voltaj düzeyidir. Elektriksel olarak bu sinyali değiştirerek en son görüntüyü değiştirebiliriz. Televizyondaki ışık ve kontrast ayarları video sinyalinin genliğini ve referansını ayarlamaya yarar ve sonuçta görüntüde aydınlanma kararma ve aydınlıkta değişiklik oluşur (Aytan ve ark., 1993).

Görüntülerin sayısal olarak işlenmesi iki boyutlu sürekli uzayda *f(x,y)* şeklinde tanımlanan bir analog görüntü ayrıklaştırılarak (örneklendirilerek) yine iki boyutlu sınırlı uzayda *f[m,n]* şeklinde tanımlanan bir sayısal görüntü üretilir. İki boyutlu analog görüntü *f(x,y)*; N satır ve M sütuna bölünür. Bir satırın ve sütunun kesiştiği nokta pikseldir. Tamsayı olarak koordinatlara atanan değerler *f[m,n]* olarak gösterilir. Aslında *f(x,y)* gösterimi iki boyutlu bir sensörden yansıyan iki değişkenli bir sinyal gibi düşünülürken birçok değişkenin fonksiyonudur. Renk derinliği (z), renk (λ) ve zaman (t) gibi birçok değişkeni içerir (Şekil 3.10) (Young ve ark., 1998).

İki boyutlu ayrık zamanlı görüntü fonksiyonu yatayda N, düşeyde M örnekten oluşan toplam NxM sonlu örnek değeri ile yatay ve düşey eksende eşit aralıklarla örnek alınarak oluşturulduğu için analog görüntü düzgün örneklenmiş olur. Oluşan sayısal görüntü N satır, M sütundan oluşan bir matristir.



Şekil 3.10. Sürekli bir görüntünün sayısallaştırılması

3.3.1. Sayısal (dijital) görüntü üzerinde yapılan işlemler

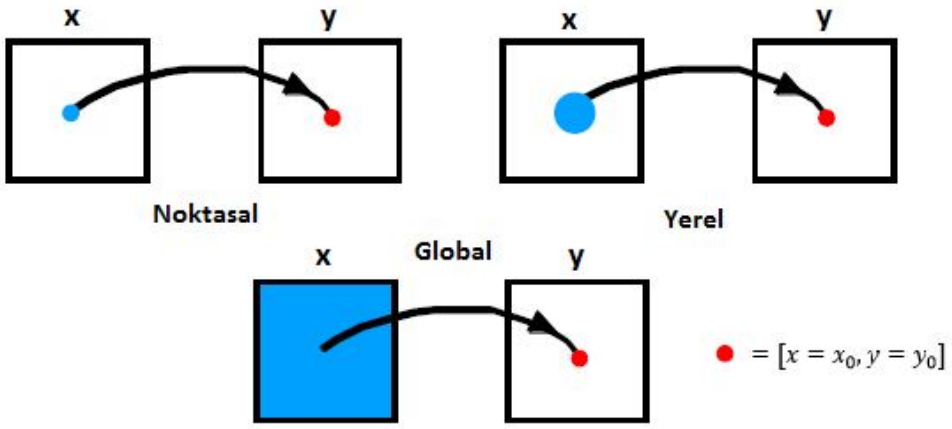
Sayısal görüntü kendine özgü konum ve değerde olan sonlu sayıda elemanlardan oluştuğu için, görüntü içindeki bilgi farklı yollardan gösterilir. En çok kullanılan gösterim biçimi uzamsal gösterim ve dalga sayısı (frekans) gösterimidir. Bu bilgilerin gösterim şekilleri mekansal veriye farklı görüş açılarıyla bakılması sebebiyle birbirinden ayrılır. Uzamsal gösterim ve dalga sayısı (frekans) gösterim biçimleri en sık tercih edilen Fourier dönüşümleri ile birbirlerine çevrilebilirler.

Günlük hayatta kullandığımız sayısal resimlerin oluşturduğu gösterime uzamsal gösterim denir. Bu gösterimde resimlerin pikselleri doğrudan işlenebilir. Görüntünün farklı frekanslı bileşenlerinden oluştuğu kabul edilir. Uzamsal gösterimde görüntü Fourier vb. dönüşüm ile frekans gösterimine çevrilir. Burada, işleme teknikleri uygulanıp ters dönüşüm yapılır. Uzaysal domaindeki işlemler aşağıdaki Denklem 3.11 ile ifade edilir.

$$g(x, y) = T[f(x, y)] \tag{3.11}$$

Burada *f(x, y)* giriş görüntüsünü temsil eder. *g(x, y)* ise çıkış (işlenmiş) görüntüsüdür. *T* olarak ifade edilen operatör, *f*'de belirli bir *(x, y)* komşuluk ilişkisi bölgesinde işlem yapmaktadır (Gonzalez ve Woods, 2002).

T'de yapılan işlemler sırasıyla nokta işlemleri, yerel (lokal) işlemler ve global işlemler olarak üç gruba ayrılabilir (Şekil 3.11).



Şekil 3.11. Sayısal görüntü üzerinde yapılan işlemlerin gösterimi (Young ve ark., 1998)

3.3.1.1. Noktasal işlemler

Görüntünün sadece bir pikselinde yapılan işlemlerdir. Kendisine komşu olan pikseller ile işlem yapılmaz. Pikselin kendi gri seviye değerinin değiştirilmesiyle hesaplanır. Bu işlemlere "piksel değeri haritalama" veya "gri ton değişikliği" de denilmektedir.

3.3.1.2. Yerel (lokal) işlemler

Çıkış görüntüsünü oluşturacak olan piksellerin değeri, giriş görüntüsünde merkezdeki piksellere komşu olan piksellerin değeri ile belirlenmektedir. Filtreler yardımıyla görüntüdeki komşu piksellerin değerleri tüm pikseller üzerinde kaydırılarak görüntünün filtrelenmesi sağlanır. Bu sebeple filtreleme işlemleri için çok sıklıkla tercih edilmektedir. Örnek olarak görüntüdeki bulanıklaşma durumunun azaltılması, gürültünün temizlenerek kenar ve bölge özelliklerinin çıkarılması verilebilir. Bir başka ifade ile yerel işlemlerde çıktı görüntüsünün bir pikselinin değeri, giriş görüntüsündeki komşu piksellerinin değerlerine bağlıdır.

3.3.1.3. Bütünsel (global) işlemler

Bütünsel (global) işlemlerde görüntündeki piksellerin tamamının gri seviye değerleri çıkış görüntüsünü oluşturulacak pikselleri etkilemektedir. Farklı bir ifade ike

global işlemlerde çıkış görüntüsündeki bir pikselin değeri giriş görüntüsündeki tüm piksellerin değerlerine bağlı olmaktadır.

Çizelge 3.1'de sayısal görüntü üzerinde yapılan işlemler özetlenerek tablo haline getirilmiştir (Young ve ark., 1998).

Çizelge 3.1. Sayısal görüntü işlemlerinin türleri

Operasyon	Karakterizasyon	Genel Karmaşıklık/Piksel
*Noktasal	Belirli bir koordinattaki çıkış değeri aynı koordinattaki giriş değerine bağlıdır.	Sabit
*Bölgesel	Belirli bir koordinattaki çıkış değeri aynı koordinattaki giriş değerinin komşularına bağlıdır.	P^2
*Global	Belirli bir koordinattaki çıkış değeri giriş görüntüsündeki tüm değerlere bağlıdır.	N^2

3.4. Görüntü İşlemenin Temel Teknikleri

Sayısal görüntü işlemenin birçok tekniği, uydu görüntüsü kullanan uygulamalar, tele-foto, tıbbi görüntüleme, görüntülü telefon, karakter tanıma ve fotoğraf iyileştirme gibi uygulamalar için 1960'lı yıllarda Jet Propulsion Laboratuvarı, Massachusetts Teknoloji Enstitüsü, Bell Laboratuvarları, Maryland Üniversitesi ve diğer araştırma tesislerinde geliştirilmiştir (Rosenfeld, 1969). Dönemin bilgisayar ekipmanları ile işleme maliyeti oldukça yüksek olduğundan dolayı zamanla değişim yaşanarak, bilgisayarların da çoğalmasıyla, görüntü işlemede kullanılacak olan donanımsal elemanlar daha kolay bir şekilde bulunur hale gelmiştir. Ayrıca, görüntüler gerçek zamanlı bir şekilde işlenebilir hale gelmiştir. Son yıllarda bilgisayarların hız kazanması ve sinyal işleyicilerin gelişimiyle sayısal görüntü işleme kavramı günümüzün en yaygın konusu haline gelmiştir.

Görüntü işleme, bilgilerin alınıp ölçme ve değerlendirme aşamalarından sonra, başka bir cihazda okunabilir bir forma dönüştürülmesi ya da bir elektronik ortamdan başka bir elektronik ortama aktarılmasına yönelik bir çalışma olan "sinyal işleme" den çok farklı bir işlemdir (Gonzalez ve Woods, 2008). Görüntüler önceden belirlenmiş olan hedefe yönelen görüntü kaynağından alınan farklı bilgilerden meydana gelir. Bu şekilde meydana gelen görüntüler ultrason, elektro mikroskop ve bilgisayar içerikli görüntülerdir.

Görüntü işleme için görüntülere uygulanan ön işleme aşaması üzerlerindeki gürültüyü (görüntünün bulanık hale getirilmesi, netlik kazandırma) azaltmaktır. Görüntülerin bilgisayar ortamında işlenmesiyle görüntülerdeki nesnelerin görüntü içeriği belirlenir. Yapılan bu detaylı belirleme aşaması ile görüntü işleme gerçekleştirilmiş olur (Samtaş ve Gülesin, 2012) (Jähne, 2005).

Görüntü işlemenin temel teknikleri olarak görüntünün elde edilmesi, filtreleme işlemleri, renk özelliklerinin çıkarılması, parlaklık eklenmesi, nesne özelliklerinin çıkarılması, kenarlarını bulma, gölge ekleme, bulanıklaştırma, gürültü giderme vb. sayılabilir. Sayısal görüntü işlemenin temelinde matematik formüller ve denklemler vardır. Bu denklemlerden en çok tercih edilenleri Fourier ve Laplacian dönüşümleridir.

3.3.1. Fourier dönüşümü

İki boyutlu bir görüntü dizisinin ayrık Fourier dönüşümü Denklem 3.12'de verilmiştir (Cooley ve Tukey, 1965; Pratt, 2001).

$$F(u,v) = \frac{1}{N}\sum_{j=0}^{N-1}\sum_{k=0}^{N-1} F(j,k)exp\left\{\frac{-2\pi i}{N}\ (uj + vk)\right. \tag{3.12}$$

Burada, i=$\sqrt{-1}$ ise, ayrık ters dönüşümü Denklem 3.13'te verilmiştir.

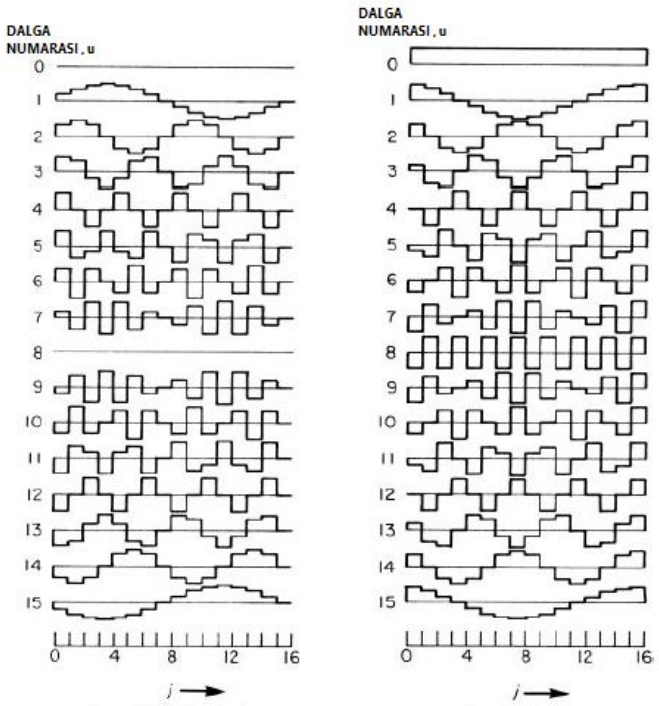
$$F(j,k) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} F(u,v)exp\left\{\frac{2\pi i}{N}\ (uj + vk)\right. \tag{3.13}$$

(u, v) indeksleri sürekli Fourier dönüşümde uzamsal frekanslar olarak isimlendirilir. Tüm yazarlar tarafından Denklem 2.12 evrensel olarak kabul görmemiştir; bazı bilimciler tüm ölçekleme sabitlerini ters dönüşüm Fourier denklemine yerleştirmeyi tercih ederken, bazıları da dönüşüm çekirdeği üzerine ters çevirme uygularlar. Dönüşümün temel fonksiyonları, sinüs ve kosinüs bileşenlerine ayrışabilen karmaşık üstel değerlerdir. Elde edilen Fourier dönüşümü çiftleri daha sonra Euler teoremine göre Denklem 3.14 ve Denklem 3.15'teki durumu alırlar.

$$A(j,k\,;u,v) = exp\left(\frac{-2\pi i}{N}\ (uj + vk)\right) = \cos\left(\frac{2\pi}{N}\ (uj + vk)\right) - i\sin\left(\frac{2\pi}{N}\ (uj + vk)\right) \tag{3.14}$$

$$A(j,k\,;u,v) = exp\left(\frac{2\pi i}{N}\ (uj + vk)\right) = \cos\left(\frac{2\pi}{N}\ (uj + vk)\right) + i\sin\left(\frac{2\pi}{N}\ (uj + vk)\right) \tag{3.15}$$

Şekil 3.12, N = 16 için tek boyutlu Fourier temel fonksiyonlarının sinüs ve kosinüs bileşenlerinin grafiğini göstermektedir (Pratt, 2001).



Şekil 3.12. N=16 için Fourier dönüşümünün temel fonksiyonları

3.3.2. Laplacian dönüşümü

Laplacian, görüntünün ikinci uzamsal türevinin iki boyutlu izotropik bir ölçüsüdür. Bir görüntünün Laplacian'ı, ani yoğunluk değişiminin bölgelerini belirler ve bu nedenle sıklıkla kenar algılaması için kullanılır. Laplacian çekirdekleri görüntüde ikinci türevsel bir yaklaşımı meydana getirdiği için gürültüye karşı çok hassastırlar. Bunu önlemek için, görüntü Laplace filtresini uygulamadan önce genellikle Gauss düzlemine getirilir. Laplacian, Gauss düzleştirme filtresine yakın bir filtre ile düzetilmiş olan görüntüye uygulanır. Bu ön işlem adımı ile yüksek frekanslı gürültü bileşenleri azaltılır. Filtre giriş görüntüsünün tek bir gri tonlama görüntüsünü alır ve çıktı olarak başka bir gri tonlama görüntüsü üretir. Amaç görüntünün gürültüye olan duyarlılığını azaltmak içindir. Piksel yoğunluk değerleri I (x, y) olan bir görüntünün Laplacian L(x, y)'ı Denklem 3.16'da verilmiştir.

$$L(x,y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \tag{3.16}$$

Görüntü üzerinde bu denklem bir konvolüsyon filtresi olarak kullanılabilir. Giriş görüntüsü ayrı pikseller kümesi olarak temsil edildiğinden, Laplacian tanımındaki ikinci türevleri yaklaştıran ayrı bir konvolüsyon çekirdeği kullanılmalıdır. Yaygın olarak kullanılan çekirdekler Şekil 3.13' te gösterilmiştir (Köybaşı, 2013).

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

Şekil 3.13. Yaygın olarak kullanılan Laplace çekirdekleri

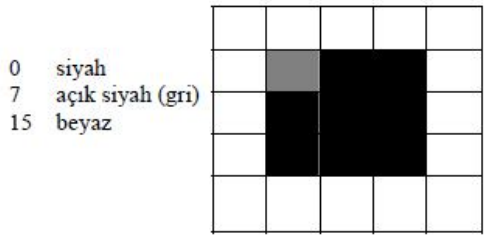
Sıfır merkezli ve Gauss standart sapmasına (σ) sahip iki boyutlu LoG fonksiyonu Denklem 3.17'de verilmiştir (Haralick ve Shapiro, 1992).

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1-\frac{x^2+y^2}{2\sigma^2}\right] e^{\frac{-x^2-y^2}{2\sigma^2}} \tag{3.17}$$

3.3.3. Gri düzey skala

Gri düzey skala, en açık uçta saf beyaz ve en sondaki ucunda saf siyahtan meydana gelen gri tonların koleksiyonu olan renk paletidir. Bilinen durumun aksine sadece beyaz renk tonunu içerir. Beyaz rengin olmaması siyah renk tonu ile ifade edilir. Gri skala sadece parlaklık bilgisi içerir. Böyle ifadelerde görüntü siyah-beyaz renk tonlarında ise, görüntü üzerindeki her bir nokta gri-düzey skala üzerindeki renk değerleriyle ifade edilir.

Her bir aydınlatma düzeyi için gerekli bitlerin yerleşiminin farklı olmasının sebebi, görüntü üzerindeki noktalar farklı olduğundandır. 4 bitlik yani 16 farklı gri-ton aydınlanma seviyesi için her bir pikselin üzerinde bulunacak gri-seviye parlaklık değeri Şekil 3.14'te gösterilmiştir. Bu gibi farklı seviyelerden meydana gelen görüntüler, gri-düzey veya gri-düzey skala ile ifade edilirler. Örneğin; piksel başına düşen bit sayıları 4 bit/piksel yani bir pikselin değerini belirtmek için 4 bit kullanılır. 0 ile 15 arasında 16 adet gri-düzey değerleri mevcuttur (Yaman, 2000).

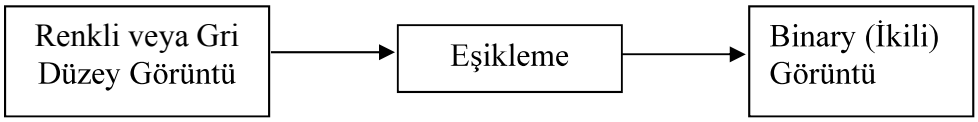


15	15	15	15	15
15	7	0	0	15
15	0	0	0	15
15	0	0	0	15
15	15	15	15	15

Şekil 3.14. On altı bitlik gri düzey skala ifadesi

3.3.4. Eşikleme

Eşikleme, en iyi sınır (eşik) değeri kullanarak gri düzeyli veya renkli bir giriş görüntüsünü binary (ikili) görüntüye dönüştürme işlemidir. Görüntünün eşikleme işleme tabi tutulması genellikle görüntü özelliklerinin (örneğin görüntüdeki nesnelerin) çıkartılmasına sebep olur. Bu nedenle görüntü işlemede kullanılan temel tekniklerdendir. Görüntü eşikleme işlemi yapılırken öncelikli olarak bir eşik değeri belirlenir. Belirlenen eşik değeri görüntüden çıkarmaya çalıştığımız nesnlerin özelliklerinin piksel değerleri olabilir. Sonrasında eşik değerinin üzerinde kalan piksel değerleri beyaza (1), altında kalan piksel değerleri ise siyaha (0) dönüştürülür. Görüntüden çıkarılan özellikler ikili (binary) görüntülerdir. Görüntünün eşiklenerek ikili değerli görüntüye dönüştürmenin amacı, görüntüdeki tek yoğunluklu ve farklı yoğunluklu pikselleri ayırt ederek görüntü içerisindeki hedef nesne veya metinleri ayırt edebilmektir. Eşikleme, noktasal işlemler grubunda yer almaktır. Eşikleme süreci Şekil 3.15'te verilmiştir.



Şekil 3.15. Eşikleme Süreci

Tüm görüntüler aynı özellikte olmadığından ve gerçek zamanlı görüntünlerin özelliklerinin belirlenebilmesi için sabit bir eşik değeri belirlenmesi doğru sonuç vermeyecektir. Bu sebeple eşik değerinin görüntünün renk dizilimine uygun şekilde belirlenmesini sağlayacak bir yöntem kullanılmalıdır. Eşik değerinin belirlenmesi için çok çeşitli yöntemler önerilmiştir.

Genel olarak eşikleme işlemi global ve yerel olmak üzere iki grupta incelenebilir. Global yöntemler, tüm eşiğe bir eşik uygularken, yerel eşikleme yöntemleri görüntünün

farklı bölgelerine farklı eşik değerleri uygular. Global eşikleme kendi içerisinde Traditional (Otsu) (Geleneksel), Iterative (Triclass) (Yinelemeli), Multistage(QIR)(Çok aşamalı) olarak üç gruba ayrılır (Senthilkumaran ve Vaithegi, 2016). Bu tez kapsamında uygulamada kullanılan global eşikleme yöntemi olan Otsu metodu incelenmiştir.

Bir görüntüdeki ön ve arka plan nesneleri arasındaki yoğunluk dağılımı çok farklı olduğunda global eşikleme yöntemleri kullanılır. Yani ön plan ve arka plan nesneleri arasında farklar çok belirgin olduğu zaman her iki nesneyi birbirinden ayırt etmek için tek bir eşik değeri kullanılabilir. Bu tip eşik belirleme yöntemlerinde eşik T değeri, sadece pikselin özelliğine ve görüntünün gri seviye değerine bağlıdır. En yaygın kullanılan bazı global eşikleme çeşitleri, Otsu, entropi tabanlı eşiklemedir. Ayrıca, Kittler ve Illingworth, Kapur, Tsai, Huang, Yen ve diğerleri gibi birçok popüler eşleme teknikleri vardır (Senthilkumaran ve Vaithegi, 2016).

Nobuyuki Otsu tarafından geliştirilen Otsu metodu, eşik değerini görüntü üzerinden hesaplayarak kullanıcının sabit bir eşik değeri belirlemesini ortadan kaldırmaktadır. Otsu metodunda, gri seviyedeki bir görüntünün arka plan ve ön plan olarak iki farklı renk sınıfından oluştuğu varsayılır. Daha sonra tüm eşik değerleri için varyans değerleri hesaplanır. Hesaplanan en küçük eşik değeri, optimum eşik değeridir. Bu yöntem gri seviyeli görüntüler üzerinde çalışmaktadır. 0-255 arasında renklerin kaç kere tekrarlandığını hesaplar. Bunun için de görüntünün histogramının hesaplanması gerekmektedir.

Sonuç olarak 0-255 arasında belirlenen bir T eşik değerine göre siyah beyaz görüntü oluşturularak görüntü eşiklenmiş olur (Denklem 3.18) (Otsu, 1979).

$$f(x,y) = \begin{cases} 0, & g(x,y) < T \\ 1, & g(x,y) \geq T \end{cases} \tag{3.18}$$

3.3.5. Histogram

Histogram, dijital görüntüdeki her gri seviyenin istatistiksel olasılık dağılımı olarak tanımlanır (Cooley ve Tukey, 1965). Histogram, gri düzey skala, segmentasyon, gri seviye dağılımı ve yoğunluğu, görüntünün; iyileştirilmesi, ortalama parlaklık değeri, kontrastı, vb. gibi özelliklerini çıkarabildiği için görüntü işleme tekniklerinde en temel işlem olarak kullanılmaktadır. Histogramlarda piksellerin hangi konumda olduğu bilgisi yoktur. Sadece frekans bilgisi bulunur. Histogram ile elde edilen dağılım ile aydınlık ve

karanlık seviyelere göre görüntü hakkında bilgi edilinilir. Böylece, histogram grafiğinde uygulanmak istenen eşik değerleri tahmin edilebilir. Koyu renkli görüntüler, piksel dağılımları sol tarafa doğru olan histogramlara sahip iken açık renkli görüntüler piksel dağılımları sağ tarafa doğru olan histogramlara sahiptirler. Matematiksel olarak, dijital bir görüntünün normalize edilmiş histogramı Denklem 3.19'da verildiği gibi tanımlanabilir.

$$P(r_k) = {}^{n_k}/_{n} \tag{3.19}$$

r_k: k'ıncı gri seviye
n_k: bu gri seviyeye sahip toplam piksel adedi
n: görüntü üzerindeki toplam piksel adedi

olarak tanımlanmıştır.

3.3.6. Histogram eşitleme

Histogram hesaplaması gri seviye görüntülerde belirli bölgelerde yığılmış pikselleri olan görüntülerde daha iyi sonuç vermektedir. Küçük yayılım gösteren görüntülerin kontrastı azdır ve bu gibi resimler kötü kontrasta sahiptir. İyi kontrastlı bir görüntüde histogram grafiği tüm gri seviye düzeylerine eşit yayılım göstermelidir. Histogram eşitleme, gri seviye renk düzeyleri eşit dağılım göstermeyen görüntülere uygulanan görüntü iyileştirme metodudur. Eşitleme sonrası görüntü daha iyi işlenebilir hale gelebilir. Histogram eşitlemenin amacı, görüntü üzerindeki renklerin frekanslarının histogram üzerinde yığılmadan düzgün olarak dağılmasını sağlamaktır. Bir başka ifade ile görüntünün olasılık dağılım fonksiyonunu doğrusallaştırmaktır. Görüntünün tamamına uygulanırsa global, belirli bir bölgesine uygulanırsa lokal histogram eşitleme ismini alır.

Histogram eşitleme işlemini gerçekleştirmek için öncelikle histogram bulunur. Sonrasında histogramdan faydalanarak toplam histogram elde edilir. Bu değer dönüştürme formülü ile normalize edilerek istenilen en büyük renk değerleri ile çarpılır ve sonuç tam sayıya yuvarlanarak toplam piksel sayısına bölünerek ortalama gri seviyesi

bulunur. Yani, görüntüdeki her orjinal değer bu ortalama gri seviyesine oranlanarak histogram değeri üzerinde dağılım elde edilmelir.

Bu işlem Denklem 3.20 ile gösterilmiştir.

$$T(r_k) = \sum_{j=0}^{k} {}^{n_j}/_{n} * (L-1) \tag{3.20}$$

Burada;

k =0,1,2....,L-1

r_k: *k*'ıncı gri seviye

n_j: j. gri seviyedeki piksel sayısı

n: görüntü üzerindeki toplam piksel adedi

olarak tanımlanmıştır.

4. YAPAY SİNİR AĞLARI

YSA (Yapay Sinir Ağları), insan beyninin bilgi işleme tekniğinden esinlenerek geliştirilmiş bir bilgi işlem teknolojisi olup, karmaşık veya yanlış verilerin anlamını türetme konusunda özelliklerini çıkartmak ve insanlar ya da diğer bilgisayar teknikleri tarafından fark edilemeyecek eğilimleri tespit etmek için kullanılabilir. YSA'nın çalışma şekli basit bir biyolojik sinir sisteminin çalışma şekline benzemektedir. Biyolojik sinir hücreleri nöronlar içerirler ve bu nöronlar farklı şekillerde birbirlerine bağlanarak ağı oluştururlar. Bu ağlar öğrenme, hafızaya alma ve veriler arasındaki ilişkiyi ortaya çıkarma kapasitesine sahiptirler. Diğer bir ifadeyle, YSA'lar, normalde bir insanın düşünme ve gözlemlemeye yönelik doğal yeteneklerini gerektiren problemlere çözüm üretmektedir. Bir insanın, düşünme ve gözlemleme yeteneklerini gerektiren problemlere yönelik çözümler üretebilmesinin temel sebebi ise insan beyninin ve dolayısıyla insanın sahip olduğu yaşayarak veya deneyerek öğrenme yeteneğidir (Elmas, 2003; Öztemel, 2003; Sağıroğlu ve ark., 2003).

McCulloch ve Pitts (1943) nörolojik anlayışa dayalı sinir ağı modelleri geliştirmişlerdir. Bu modeller, nöronların nasıl çalıştığına dair çeşitli varsayımlarda bulunmuştur. Geliştirdikleri sinir ağları sabit eşikli, ikili cihazlar olarak düşünülen basit nöronlaradan oluşmuştur. Modellerinin sonuçları, "a veya b" ve "a ve b" gibi basit mantık fonksiyonlarını göstermiştir. İlk yapay nöronu modellemişlerdir; ancak dönemin kısıtlı olanakları nedeniyle bu alanda çok gelişme sağlayamamışlardır.

Farley ve Clark (1954); bilgisayar sistemleri ile modellerin gelişimini sağlamak konusunda McGill Üniversitesi'ndeki bilim insanları ile görüşerek yapay sinir ağları etkileşimleri konusunda çeşitli önermeler oluşturmuşlardır (Anonymous, 2017).

Ayrıca yalnızca nörofizyologlar değil, psikologlar ve mühendisler de sinir ağı simülasyonlarının ilerlemesine katkı sağlamışlardır. Rosenblatt (1958), perceptronu tasarlarken ve geliştirirken sahadaki kayda değer ilgi ve etkinliği harekete geçirmiştir. Perceptron, orta tabaka ilişkilendirme katmanı olarak bilinen üç katmana sahip bir sistem olup, belirli bir girdiyi rastgele bir çıktı birimine bağlamayı veya ilişkilendirmeyi öğrenebilmiştir.

Başka bir sistem, 1960'da Widrow ve Hoff (Stanford Üniversitesi) tarafından geliştirilen ADALINE'dır. ADALINE, basit bileşenlerden yapılmış analog elektronik bir cihazdır. Öğrenmede kullanılan yöntem perceptronun yönteminden farklı olup, en küçük kareler (LMS) öğrenme kuralı kullanılmıştır.

1969 yılında Minsky ve Papert (1969), perceptronun bazı basit mantıksal işlemlerde yetersiz olduğunu göstermişlerdir. Örneğin bir perceptronun XOR problemini çözemediği görülmüştür. Tek katmanlı perceptronların sınırlılıklarını çok tabakalı sistemlere genelleştirdikleri bir kitap hazırlamışlardır; ancak nasıl eğitilebileceği konusunda çözüm bulunamamıştır.

Klopf (1972) "heterostaz" olarak adlandırılan nöronal öğrenme için biyolojik bir prensibe dayalı suni nöronlarda öğrenim için bir temel geliştirmiştir.

Paul Werbos (1974), geri yayılım öğrenme yöntemini geliştirmiş ve kullanmıştır. Geri yayılım ağları bugün muhtemelen sinir ağlarının en çok bilinen ve yaygın şekilde uygulanan algoritmalarıdır.

Fukushima (1975), el yazısı karakterlerin yorumlanması için eğitimli çok tabakalı sinir ağı geliştirmiştir. Orijinal ağ 1975 yılında basılmıştır ve "Cognitron" olarak anılmıştır.

Kohonen (1988) birbirinden bağımsız olarak ilişkisel teknikler geliştirmiştir. Yapay sinir ağlarının adaptif elemanların birden fazla olarak çok miktarda paralel bağlanmasıyla oluşan ve insan beyin nöronlarını taklit ederek gerçek dünya ile aynen biyolojik sinir sisteminin davranışına benzer şekilde ilişkide bulunabildikleri ifade edilmiştir. Ayrıca, bu yapıların hiyerarşik organizasyonları düzenlenmiş yapılar olduğuna dikkat çekilmiştir (Jang ve ark., 1997).

Steve Grossberg ve Gail Carpenter (1988) ART (Adaptif Rezonans Teorisi) ağlarını geliştirmişlerdir.

4.1. Biyolojik Sinir Sistemi

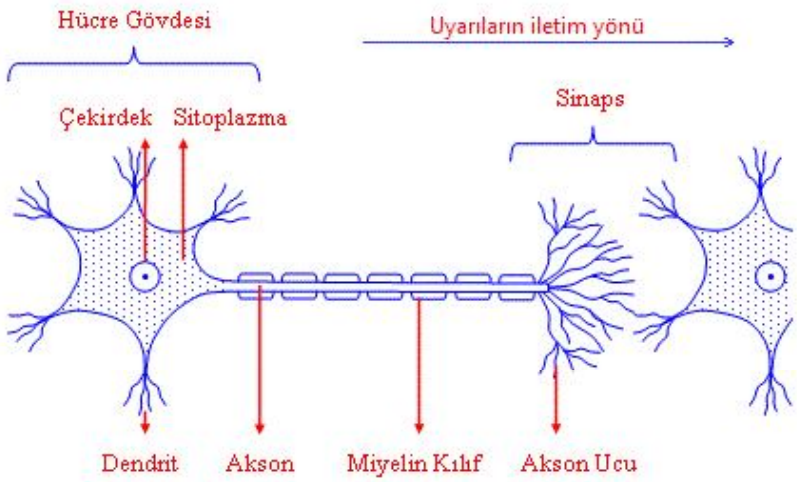
Yapay bir sinir hücresi, biyolojik bir sinir hücresinin davranış şekillerinden esinlenerek ortaya çıkan bir algoritma olarak tanımlanabilir. Biyolojik sinir hücresinin işleyişi göz önüne alınırsa, yapay bir sinir hücresinin diğer sinir hücrelerinden aldığı sinyallerin kendi bünyesinde toplandığı ve toplanan bu sinyal kümülatifinin belli bir sınır değerini aştığı anda, bu yapay sinir hücresinin kendi sinyalini bir başka sinir hücresine aktardığı söylenebilir (Ataseven, 2013).

İnsan beyninde yaklaşık olarak 10 milyar sinir hücresi bulunmaktadır. Bu sinir hüclerini birbirine bağlayan 60 trilyon taşıyıcı hücre vardır. Duyu organlarından sinyalleri alarak merkezi sinir sistemine kadar ulaştırılmasını taşıyıcı hücreler sağlar. Merkezi sinir sistemi kendisine gelen bilgileri alıp yorumlayarak belirli çıkış sinyalleri üretir.

Oluşturulan çıkış sinyalleri ilgili organlara sinir sistemi aracılığıyla ulaştırılır. Şekil 4.1'de biyolojik sinir sistemine ait blok şeması verilmiştir (Fırat ve Güngör, 2004).

Şekil 4.1. Biyolojik sinir sisteminin blok şeması

Merkezi sinir sisteminde veriler, alıcı ve tepki sinirleri arasında ileri ve geri yönlü olarak değerlendirilip uygun çıkış tepkileri üretilmektedir. Bu yönüyle biyolojik sinir sistemi, kapalı çevrim bir denetim sisteminin karakteristiklerini taşımaktadır. Sinir hücresi (nöron) merkezi sinir sisteminin temel işlem elemanıdır. Sinir hücresi; hücre gövdesi, aksonlar ve dendritler olmak üzere üç yapıdan meydana gelmektedir. Diğer hücrelerden alınan bilgiler bir ağaç yapısı şeklinde ince yollarla hücre gövdesine dendritler aracılığıyla iletir. Aksonların görevi elektriksel sinyaller şeklindeki bilgiyi hücreden dışarı taşıyan daha uzun bir yoldan iletmektir. Aksonların son kısmı, ince yollara ayrılabilir ve ayrılan yollar, diğer hücreler için dendritleri oluşturur. Biyolojik sinir sisteminin yapısı Şekil 4.2'de gösterilmektedir (Demir, 1997).



Şekil 4.2. Biyolojik sinir sisteminin yapısı

Öğrenme işlevi biyolojik sistemlerde nöronlar arasındaki sinaptik (synaptic) bağlantıların ayarlanması ile gerçekleşmektedir. Bunun anlamı, insanlar doğdukları andan itibaren yaşayarak öğrenme süreci içerisine dahil olurlar. Bu süre boyunca insan

beyni sürekli olarak gelişim içerisindedir. İnsanların yaşamları boyunca tecrübe ettikleri durumlar sinaptik bağlantılar ile ayarlanır ve yeni sinaptik bağlantılar oluşur. Böylece öğrenme gerçekleşir. Bu durumun benzeri YSA için de geçerlidir. Öğrenme işlemi, eğitme yoluyla örnekler alınarak gerçekleşir. Farklı bir deyişle gerçekleşme olayı giriş ve çıkış verilerinin işlenmesiyle, yani eğitim işleminin algoritmasının verileri kullanarak bağlantı ağırlıklarını (weights of the synapses) bir yakınsama sağlanana kadar, sürekli olarak ayarlamasıyla olur (Elmas, 2003; Öztemel, 2003; Sağıroğlu ve ark., 2003)

4.2. Yapay Sinir Ağlarının Matematiksel Modeli

Nöronlar bir sinir ağının yapısını meydana getirirler. Tek başlarına ele alındıklarında basit bir göreve sahip işlemcilerdir. Bir nöron üç ana bölümden oluşmaktadır. Bir nöronun yapısı sırasıyla sinapslar, toplayıcı ve aktivasyon fonksiyonundan meydana gelir. Şekil 4.3'te bir nöronun matematiksel modeli verilmiştir. Bu şekilden anlaşılacağı gibi, nöron girdileri sinaptik bağıntılar üzerindeki ağırlıklar ile çarpılarak bir toplayıcıya ulaşmakta ve elde edilen toplam, nöronun aktivasyon fonksiyonundan geçirilerek çıkışlar hesaplanmaktadır (Fırat ve Güngör, 2004).

Denklem 4.1'de, ağırlıklı toplamın oluşturulması fonksiyonu, Denklem 4.2'de ise nöron çıkışının fonksiyonu verilmiştir.

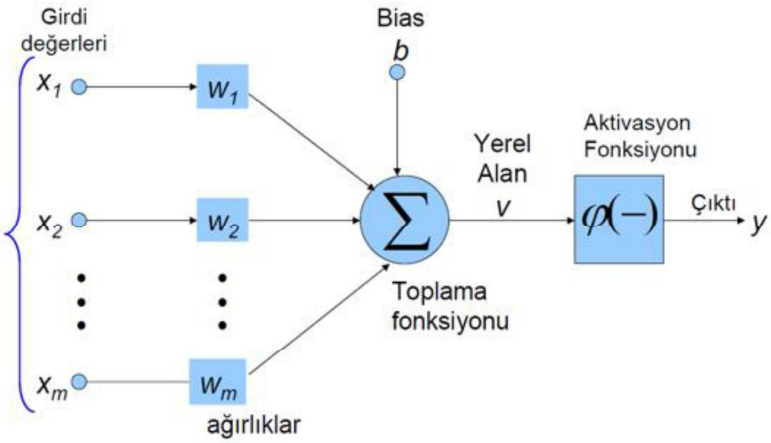
$$S = w_1.u_1 + w_2.u_2 + w_3.u_3 + \cdots + w_n.u_n - \theta = \sum_{i=1}^{n} w_i.u_i - \theta \qquad (4.1)$$

$$Q = \psi(S) \qquad (4.2)$$

S : Toplam fonksiyonu
u_i : Giriş fonksiyonu
w_i: Ağırlıklandırma faktörü
Q : Çıkış fonksiyonu
Ψ(S) : Aktivasyon fonksiyonu
θ : Eşik değeri

Nörona her girişteki değişim, nöronun çıkışında belirli bir değişime sebep olmakta ve bu değişimin genliği, girişinin etki derecesini belirleyen bağlantı kazançlarına, toplayıcının eşik değerine ve nöron aktivasyon fonksiyonunun tipine bağlı olmaktadır.

Eşik değeri pratikte -1 ya da +1 olarak alınır. Sabit bir girişin θ ağırlığı olan bir bağıntı ile toplayıcıya girdiği şeklinde ele alınır (Kaynak ve Efe, 2000).



Şekil 4.3. Bir nöronun matematiksel modeli

Girdiler: Dış dünyadan bir yapay sinir hücresine gelen bilgiler girdileri oluşturur. Dış dünyadan ya da bir önceki katmandan gelen veriler giriş olarak yapay sinir hücrelerine gönderilir (Özveren, 2006).

Ağırlıklar: Ağırlıklar bir yapay sinir hücresine gelen bilginin önem derecesini ve nöron üzerindeki etkisini gösterir. Ağırlıklar (w1, w2, w3…wm) yapay sinir tarafından alınan girişlerin sinir üzerindeki etkisini belirleyen uygun katsayılardır (Öztemel, 2003).

Toplam Fonksiyonu: Toplam fonksiyonu, bir sinir hücresine gelen net girdi değerini hesaplar. Toplam fonksiyonu sonucunda hesaplanan değer, doğrusal ya da doğrusal olmayan türevlenebilir bir aktivasyon fonksiyonundan geçirilerek işlem elemanının çıktı değeri hesaplanır (Taşova, 2011).

Aktivasyon Fonksiyonu: Girdi ve çıktı verileri arasındaki eğrisel eşleşme aktivasyon fonksiyonu tarafından sağlanmaktadır. Aktivasyon fonksiyonunun uygun şekilde seçilmesi, ağın performansını önemli boyutta etkiler (Hwang ve Ding, 1997). Aktivasyon fonksiyonu çoğunlukla doğrusal olmayan bir fonksiyon olarak seçilmektedir. YSA'nın bir özelliği olan "doğrusal olmama" aktivasyon fonksiyonlarının doğrusal olmama özelliğinden kaynaklanır. Aktivasyon fonksiyonu seçimi yapılırken en önemli konu ise bu fonksiyonun türevinin kolay hesaplanabilir olmasıdır. Geri beslemeli ağlarda aktivasyon fonksiyonunun türevi de kullanıldığından dolayı hesaplamanın yavaşlama durumuna karşı türevi kolay hesaplanır bir fonksiyon seçilir. Günümüzde en çok kullanılan "çok katmanlı algılayıcı" modelinde genellikle "Sigmoid fonksiyonu"

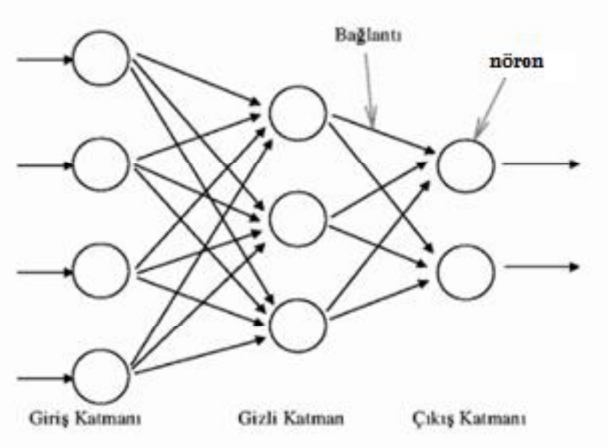
aktivasyon fonksiyonu olarak kullanılır. Seçilen aktivasyon fonksiyonuna göre çıktı değeri genellikle [-1,1] veya [0,1] arasında değişir (Yavuz ve Deveci, 2012).

Çıkış İşlevi: YSA'da kullanılan aktivasyon fonksiyonu uygulanmasıyla elde edilen çıktı değeri çıkış işlevini oluşturmaktadır.

4.3. Yapay Sinir Ağlarının Yapısı

YSA, yapay sinir hücrelerinin birbirlerine bağlanması sonucu meydana gelen yapılardır. YSA'da sinir hücrelerinin birbirine bağlanması rastgele olmaz. Birbirlerine bağlı olan hücrelerin paralel çalışmaktadırlar. Bu hücrelere, işlem elemanı denir. Her bir sinir hücresinin birbirine olan bağlantılarının bir değere sahip olduğu varsayılmaktadır. Bilginin öğrenim ile elde edildiği ve bu bilgilerin bağlantılarda depolandığı kabul edilmektedir. YSA'nın amacı kendisine verilen bir girdi setine karşılık gelebilecek bir çıktı seti belirleyebilmektir. Bunu yapabilmek için ağ, ilgili durumun örnekleri ile eğitilerek (öğrenme) genelleme yapabilecek yeteneğe kavuşturulur. Bu öğrenme işlemi ile benzer durumlara karşılık gelen çıktı setleri belirlenmiş olur (Öztemel, 2003).

Bir YSA sadece bir girdi ve bir çıktı katmanlarından meydana gelmiş ise tek katmanlı bir ağdır. Sadece giriş ve çıkış katmanı olan tek katmanlı ağların karmaşık işlemleri hesaplama yeteneği yoktur. Karmaşık işlemleri hesaplayabilmek için YSA'da en azından bir gizli (ara) katman olmalıdır (Elmas, 2003). Birden fazla gizli ve çıkış katmanı olan ağlara çok katmanlı ağlar denilmektedir. Bu katmanlar bağlantılar, nöron ve ağırlıklar gibi bileşenlerden oluşmaktadır. Çok katmanlı ağların çalışma prensibi temel olarak iki aşamadan oluşur. Bunlar eğitim ve test aşamasıdır. Eğitim aşamasında belirlenen öğrenme algoritmasına göre ağırlıklar hesaplanarak bu değerlere göre bir çıkış hesaplanmaktadır. Test aşamasında ise ağın daha önce tanımadığı örnekler ağa verilerek sistem test edilir. Öğrenme algoritması yöntemleri, girdi setine karşılık uygun çıktı seti belirlenmesinde ve ağları hızlandırma konusunda önemli rol oynamaktadır. Hücrelerin bağlantı şekillerine, aktivasyon fonksiyonlarına ve öğrenme kurallarına göre çeşitli YSA yapıları geliştirilmiştir. YSA, temel olarak üç ana katmandan oluşur. Örnek yapı Şekil 4.4'te verilmiştir (Yavuz ve Deveci, 2012).



Şekil 4.4. YSA yapısı

Giriş Katmanı: Giriş katmanında dış dünyadan veriler alınır ve iç katmanlara yönlendirilir. Giriş katmanındaki nöron adedinin, giriş veri sayısı kadar olduğu düşünülür ve her bir giriş nöronu bir veri alır. Bu ağlarda veri herhangi bir işlemden geçirilmez.

Gizli Katman: Giriş katmanından gelen veriler işlenerek çıkış katmanına gönderilir. Gelen bilgilerin işlenmesi bu katmanda gerçekleştirilir. Giriş tabakasından alınan ağırlıklandırılmış veriler probleme uygun bir fonksiyon ile işlenerek çıkış katmanına gönderilir. Bir ağ içerisinde birden fazla gizli katman olabilir.

Çıkış Katmanı: Bu katmandaki hücreler gizli katmandan gelen bilgiyi işleyerek sunulan giriş seti için üretilmesi gereken çıktıyı üretirler ve üretilen çıktı dış dünyaya gönderilir. Çıkış katmanındaki nöron sayısı ağa sunulan her verinin çıkış sayısı kadar değer alır.

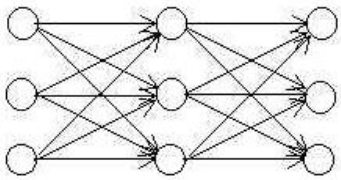
İleri besleme aşamasında giriş katmanındaki nöronlar, veri değerlerini doğrudan gizli katmana iletir. Gizli katmandaki her nöron kendi giriş değerlerinin ağırlıklarını hesaplayarak toplam bir değer hesap ederler ve bu hesaplanan değerleri bir fonksiyondan geçirerek bir sonraki katmana veya doğrudan çıkış katmanına iletirler. Katmanlar arasındaki ağırlıklar rastgele olarak küçük rakamlardan tercih edilir (Fırat ve Güngör, 2004).

4.3.1. Ağ yapılarına göre yapay sinir ağları

Ağ yapılarına göre yapay sinir ağları, nöronlar arasındaki bağlantı yönlerine ve ağ içerisindeki akış yönlerine göre ileri beslemeli ve geri beslemeli yapay sinir ağları olarak iki gruba ayrılmaktadır (Slaughter ve Hobson, 2006).

4.3.1.1.İleri beslemeli yapay sinir ağları

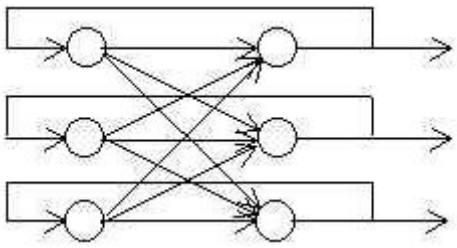
İleri beslemeli yapay sinir ağları türünde gerçekleşen işlemler giriş katmanından çıkışa doğru iletilir. Giriş katmanı, aldığı verileri hiçbir işleme tabii tutmadan gizli katmana iletir. İletilen veri gizli ve çıkış katmanında istenilen fonksiyonlardan geçirilerek ağ çıkışı elde edilir. Şekil 4.5'te ileri beslemeli yapay sinir ağı modeli verilmiştir.



Şekil 4.5. İleri beslemeli yapay sinir ağları

4.3.1.2. Geri beslemeli yapay sinir ağları

Geri beslemeli yapay sinir ağlarında en az bir hücrenin çıkışı kendisine veya diğer hücrelere giriş verisi olarak verilir ve çoğunlukla bir geciktirme elemanı üzerinden geri besleme işlemi yapılır. Geri besleme işlemi bir katmandaki hücreler arasında olduğu gibi katmanlar arasındaki hücreler arasında da gerçekleşebilir. Bu çalışması ile geri beslemeli yapay sinir ağları doğrusal olmayan dinamik bir davranış özelliği gösterirler. Bu sebepten dolayı doğrusal olmayan problemleri çözmede sıklıkla kullanılırlar. Şekil 4.6'da geri beslemeli yapay sinir ağı modeli verilmiştir (Anonim, 2017).



Şekil 4.6. Geri beslemeli yapay sinir ağı

Geri beslemeli yapay sinir ağları, hücreler arası veya katmanlar arası besleme tipine göre değişik isimlerle adlandırılırlar (Minsky ve Papert, 1969).

Tam geri beslemeli ağlar gelişi güzel ileri ve geri bağlantıları olan ağlardır. Bu ağlarda bağlantıların tamamı eğitilebilir.

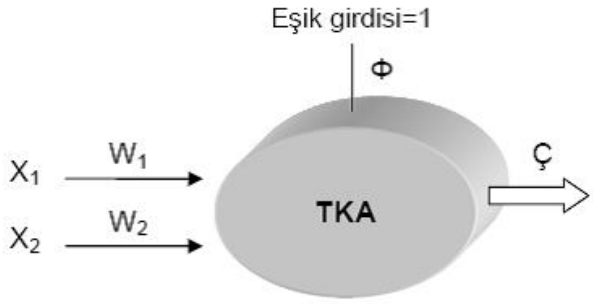
Kısmi geri beslemeli ağlarda ise, ağın hücre elemanlarına ilavaten bir de içerik elemanları bulunur. Geri besleme işlemi yalnızca içerik elemanları üzerinde yapılır ve bu bağlantıları eğitmek mümkün olmaz. İçerik elemanları ara katman elemanlarının geçmiş durumlarını ve geçmiş işlemleri hatırlamak için kullanılırlar (Uğurlu, 2011).

4.4. Yapay Sinir Ağlarının Türleri

Geçmişten günümüze farklı alanlarda ve her biri farklı amaçlar için kullanılan birçok YSA modeli geliştirilmiştir. Örneğin; Perceptron, Adaline, MLP, LVQ, Hopfield, Recurrent, SOM, ART, PCA vb. YSA türleri bakımından üç gruba ayrılabilir.

4.4.1. Tek katmanlı sinir ağları

Girdi ve çıktı katmanlarından oluşan sinir ağlarına tek katmanlı sinir ağları denilmektedir. Çıktı katmanları girdi katmanlarına bağlanırlar. Her bir bağlantının ağırlığı "w" olarak gösterilir. Örnek olarak iki girdi ve tek çıktıdan oluşan tek katmanlı bir yapay sinir ağı Şekil 4.7'de verilmiştir (Öztemel, 2003).

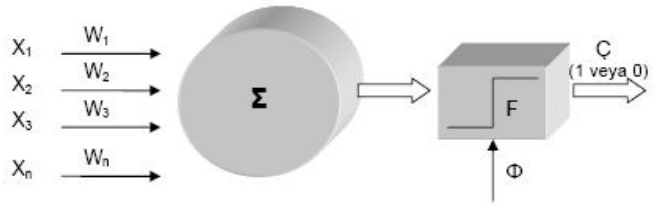


Şekil 4.7. Tek katmanlı algılayıcı modeli

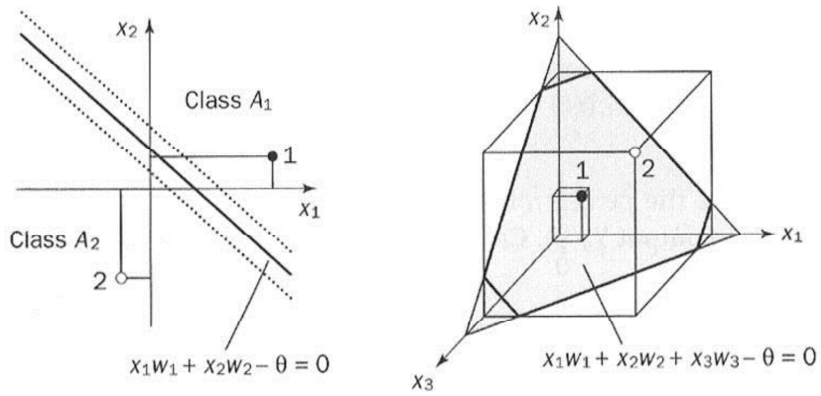
En basit tek katmanlı ağ modeli olarak Perceptron'u örnek verebiliriz. Perceptron ilk kez 1950'li yılların sonuna doğru Rosenblatt tarafından örüntü sınıflandırma amacıyla ortaya atılmıştır. İlk kullanım amacı sınıflandırma işlemi gerçekleştirmek olan tek katmanlı ağ modeli, McCulloch ve Pitts tarafından geliştirilen daha karmaşık bir modeldir (Nygren, 2004). Bu modelin çalışması bir sinir hücresinin birden fazla girdiyi alarak bir çıktı üretmesi prensibine dayanmaktadır. Şekil 4.8'de basit algılayıcı sistem olan

Perceptron gösterilmektedir. Bir başka deyişle, bir araya gelmiş sinir hücrelerinin miktarına bağlı olarak perceptronlar ile çeşitli sayıda sınıflandırma problemi çözülebilir. Sınıflandırma işleminin doğru sonuç vermesi için sınıfların düzlemsel olarak ayrılması gerektiği ifade edilmiştir (Minsky ve Papert, 1969).

Perceptron modeli eğitilebilen tek bir YSA hücresinden oluşmaktadır. Elde edilen ağın çıktısı mantıksal bir değerdir. Ağırlıklandırılmış girdi bilgileri aktivasyon fonksiyonundan geçirilir ve 0 veya 1 şeklinde çıktı elde edilir. Perceptron'un amacı girdileri sınıflandırmaktır. N boyutlu bir uzay Şekil 3.9'daki gibi bir doğru ya da düzlem ile iki bölgeye ayrılır.



Şekil 4.8. Basit algılayıcı model (Perceptron)



Şekil 4.9. Sınıfların düzlemsel veya doğrusal olarak ayrılması

1959 yılında Stanford Üniversitesi'nden Bernard Widrow, basit nöron benzeri elemanlara dayanan ve ADALINE olarak isimlendirilen bir adaptif lineer elemanı fikri ortaya atmıştır. Adaptive Linear Element (ADALINE) olarak bilinen yinelemeli olan algılayıcıda hedef karelerin ortalamasının en az düzeye indirilmesidir. Basit perceptron ile arasındaki fark öğrenme kuralıdır. Öğrenme kuralı olarak danışmalı öğrenme kullanılmaktadır (Aygören ve ark., 2012).

ADALINE, birçok uygulama için iyi çalışmasına rağmen lineer çalışma uzayı ile sınırlı olup, lineer transfer fonksiyon kullanmaktadır. Giriş ve istenilen çıkış desenlerinin tekrar tekrar ağa uygulanmasıyla eğitim gerçekleştirilir. Özellikle ses sinyalleri üzerindeki gürültünün yok edilmesi, karakter tanımlama, hava tahmini ve adaptif kontrol gibi uygulamalarında kullanılmaktadır.

MADALINE (çoklu yinelemeli algılayıcı) olarak bilinen algılayıcının ADALINE'den farkı değişken sayısının daha fazla olabilmesidir (Anonim, 2017).

4.4.2. Çok katmanlı sinir ağları (MLP)

Tek katmandan oluşan algılayıcılar sadece lineer fonksiyonlar üzerinde tahmin yürütebildikleri, dolayısıyla lineer olmayan fonksiyon üzerinde çözüm üretemedikleri için hem yapısal hem de eğitme algoritması bakımından daha çok geliştirilmiş olan MLP önerilmiştir.

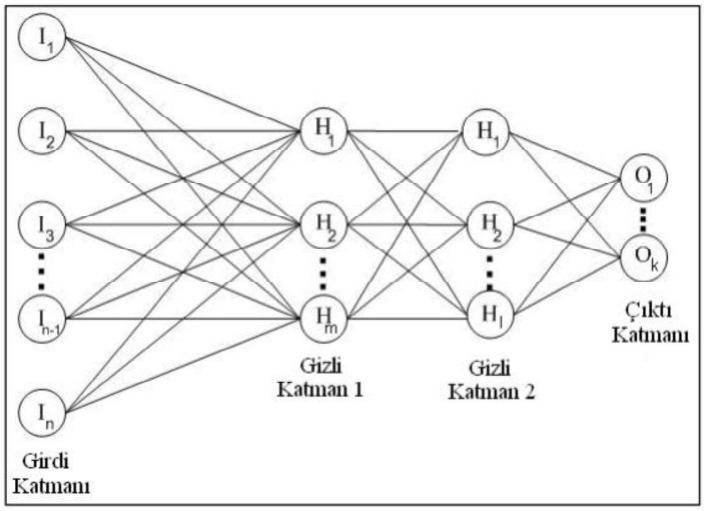
Günlük hayatta karşılaşılan problem çoğu lineer olmayan fonksiyonlar üzerinedir. Lineer olmayan bir ilişki gösteren XOR problemini çözebilmek için yapılan çalışmalar neticesinde çok katmanlı sinir ağları geliştirilmiştir. Hata yayma ya da geriye yayılım modeli olarak anılan bu model Rumelhart ve arkadaşları tarafından geliştirilmiştir (Saraç, 2004). MLP'nin geliştirilmesi ile mühendislik uygulamalarındaki lineer olmayan problemlerin çözüme kavuşturulmuştur. Lineer olmama özelliği genellikle sigmoid fonksiyonu ile sağlanır. Aynı zamanda bu ağda birçok öğrenme algoritmasının eğitmede uygulanabilir olması MLP modelinin yaygın kullanılmasına neden olmuştur.

MLP, giriş ve çıkış katmanı arasında gizli katmanlara sahiptir. Her bir katmanda bir veya daha fazla sayıda işlem elemanı barındırır. İşlem elemanlarının fazla olması çok yüksek derecede bilgi işlenmesi demektir. Bir katmandaki işlem elemanlarının tümü bir sonraki katmanda olan bütün işlem elemanlarına bağlıdır. Ağın bilgi işlemesinde herhangi bir değişiklik olabilmesi için bağlantı sayısı ya da ağırlıklar farklı olmalıdır.

İleri beslemeli sinir ağı olarak da adlandırılan çok katmanlı sinir ağlarında geri besleme yoktur. MLP'de bilgiler giriş katmanında kabul edilir; fakat giriş katmanında bilgi işleme yapılmaz. Gizli katmanda gerçekleştirilen işlemler çıkış katmanına aktarılır. Gizli katman sayısı ve gizli katman işlem elemanları sayısı deneme yanılma yöntemiyle elde edilebilir.

MLP'nin amacı, ağın istenen çıktı değeri ile ürettiği çıktı değeri arasındaki hatayı en aza indirgemektedir. Bu tür ağlara eğitim sırasında hem girişler hem de o girişlere

karşılık üretilmesi gereken yani istenen çıktı değerleri gösterilir. Yapılan bu işleme danışmalı öğrenme denmektedir. Ağın görevi her giriş değerine karşılık gelen çıkış değeri üretmektir. Belirlenen eğitim algoritmasına göre ağın çıkışı ile istenilen çıkış arasındaki hata tekrar geriye doğru yayılarak minimum değere düşünceye kadar ağın ağırlıkları tekrar tekrar hesaplanarak değiştirilir (Saraç, 2004). Ağ, kendisine verilen girdilerden öğrenme işlemini gerçekleştirerek problemi temsil eden bir çözüm uzayı üretir. Sonrasında verilen benzer girdiler için bu çözüm uzayı sonuçlar ve çözümler elde etmektedir (Öztemel, 2003). Şekil 4.10'da çok katmanlı yapay sinir ağı modeli gösterilmiştir.



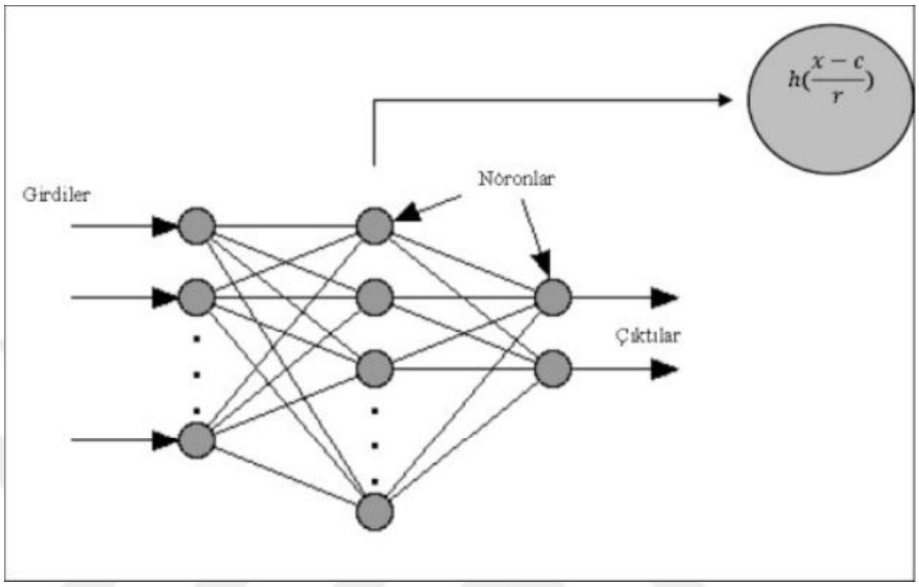
Şekil 4.10. MLP sinir ağı modeli

4.4.3. Radyal tabanlı sinir ağları

RTFA (Radyal tabanlı sinir ağları), çok değişkenli modelleme ve yakınsamalarda kullanılır. RTFA mimarisi çok boyutlu uzayda eğri uydurma yaklaşımının bir benzeri olarak düşünülebilir. RTFA'ı eğitme işlemi çok boyutlu uzayda eğitim verilerine en uygun bir yüzeyi bulma problemine dönüşmüş olur. RTFA, ileri beslemeli yapay sinir ağlarına benzer şekilde giriş, gizli ve çıkış katmanlarından oluşur; ancak giriş katmanından gizli katmana dönüşüm RTFA ile doğrusal olmayan sabit bir dönüşümdür. Orta katmandan çıkış katmanına ise doğrusal bir dönüşüm gerçekleştirilir (Saraç, 2004).

RTFA, girdi veri elemanlarının belirlenmiş olan bölgelere karşılık gelen değerler için en büyük (ya da en küçük) değerini alan ve bu noktadan uzaklaştıkça daha küçük (ya da daha büyük) değerler üreten fonksiyonlardır. Herhangi bir tahmin modeli için kullanıcı

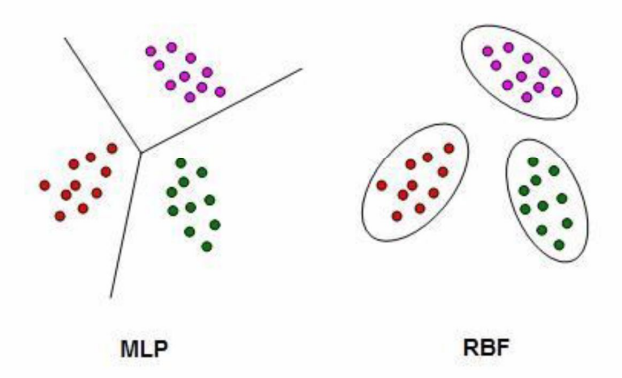
tarafından rastgele denemelerle belirlenen ve deneme yanılma yöntemi ile geliştirilebilen parametrelerinin az oluşu, ilgili modelin kullanılabilirliği bakımından olumlu bir durumdur. Bu özelliği ile RTFA'nın çok katmanlı YSA'ya göre daha kullanışlı olduğu düşünülebilir. Şekil 4.11'de RTFA modeli verilmiştir.



Şekil 4.11. Radyal tabanlı sinir ağı modeli

RTFA'da giriş katmanının görevi sadece ağ modeline dış dünyadan veri alınmasıdır. Yani ağa verilen girdiler, giriş katmanında veriler üzerinde işlem yapılmadan doğrudan gizli katmana iletilir. Giriş katmanını gizli katmana bağlayan tüm ağırlık değerlerinin "1" olduğu ve çözüm boyunca değişmediği varsayılmaktadır. Bu özelliği bakımından öğrenme aşamasında değeri değiştirilecek parametre sayısında önemli bir azalma gerçekleşmektedir ve öğrenme hızlanmaktadır.

Şekil 4.12'de MLP ve RBF (Radial Basis Function) karşılaştırması gösterilmiştir (Anonymous, 2017).



Şekil 4.12. MLP ve RBF Karşılaştırması

4.5. Yapay Sinir Ağlarında Öğrenme

YSA'nın verilen girdilere göre çıktı verebilmesinin yöntemi ağın öğrenebilmesinden geçer. Öğrenme işlemi ağa verilecek olan örnekler arasında yapının iyi bir öğrenme davranışı göstermesini sağlayabilecek olanlardan seçilerek gerçekleştirilir. Bunun için bağlantı ağırlıklarının hesaplanması gerekmektedir. Ağ, öğrenme sırasında edindiği bilgileri sinir hücreleri arasındaki bağlantı ağırlıkları olarak depolar. Bu ağırlıklar sayesinde YSA'nın verileri düzgün ve doğru bir şekilde işleyecek kapasiteye ulaşılmış olur. Ağırlıklarda verilerin gerekli olan bilgileri bulunur (Şen, 2004).

Yapay sinir ağlarında temel olarak üç çeşit öğrenme şekli vardır. Bunlar, danışmalı öğrenme, danışmasız öğrenme ve takviyeli öğrenmedir.

4.5.1. Danışmalı öğrenme

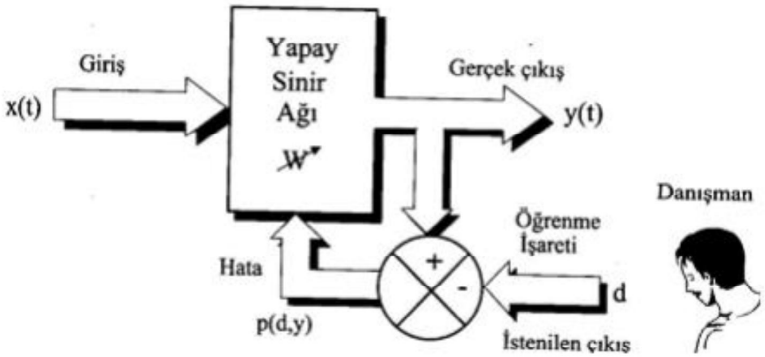
Uygulamalarda en çok kullanılan yöntemdir. Eğitimli veya denetimli öğrenme olarak da isimlendirilebilir. Danışmalı öğrenmenin temel mantığında YSA'nın kullanılmadan önce eğitilmesi yatmaktadır.

Eğitme işlemi sinir ağına giriş ve çıktı bilgilerini verilmesiyle gerçekleştirilir. Girişe verilen bilgiler genellikle eğitme kümesi olarak adlandırılır. Bunun anlamı her girdi kümesi için çıktı kümesi ağa gösterilmektedir (Elmas, 2003).

Danışmalı öğrenme modelinde öğrenme sonucu elde edilen çıktı değerleri ile gerçek çıktı değerlerinin karşılaştırması yapılır. Hata miktarına göre minimum bir değere gelinceye kadar döngü devam eder. Hedef olarak belirlenen çıktı değerine yakınlaşmış bir değer elde edilince öğrenme bitmiş olur. Öğrenme bitince ağırlıklar tekrar değişmez. Test süresince boyunca eğitim esnasında ağa önceden gösterilmeyen veriler girdi olarak

verilir. Test sonucunda elde edilen sonuçlar istenilen sonuçlar ile karşılaştırılarak ağın test süreci tamamlanır. Test aşamasında çıkan sonuçların istenilen düzeyde olmaması durumunda ağdaki iterasyon sayısı, ara katmanlar, ara katmanlardaki nöron sayıları ve ağırlıkları değiştirilir. Böylece öğrenme süreci tekrar edilmiş olur (Anonim, 2017).

Şekil 4.13'te danışmalı öğrenme yapısı verilmektedir.



Şekil 4.13. Danışmalı öğrenme yapısı

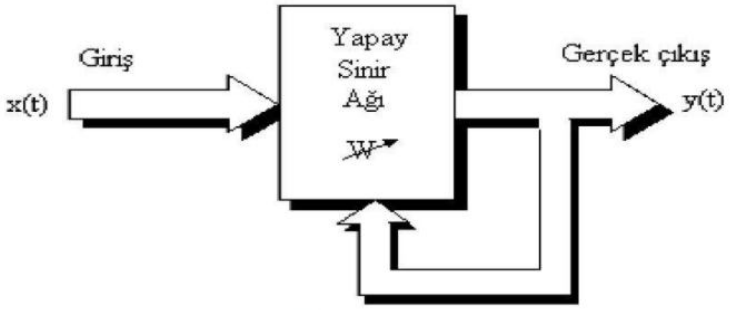
Danışmalı öğrenmede kullanılan etki metotlara örnek olarak kullanılan ağlar; perceptron ve ilişkili hafızalar, takviyeli öğrenme, stokastik öğrenme, vektör nicelik öğrenmesi, delta ve genelleştirilmiş delta kuralı, geri yayılma algoritması şeklinde sıralanabilir (Civalek, 2004).

4.5.2. Danışmasız öğrenme

Danışmasız öğrenmede sistemin öğrenmesine yardımda bulunan herhangi bir danışman yoktur. Sisteme sadece giriş değerleri verilir (Öztemel, 2003). Eğitimli veya denetimsiz öğrenme olarak da adlandırılmaktadır. Bu öğrenme çeşidinde herhangi bir beklenen çıktı bilgisi, yani sonuçlar verilmez. Sadece giriş değerleri verilerek ağın eğitimi sağlanmaktadır. Sistem giriş değerlerine göre ağırlık değerlerini ayarlar. İstenilen sonuç elde edilinceye kadar ağırlık değerleri değiştirilir.

Danışmasız öğrenmede çıktı değerinin yerine benzer tip girdilerin kümesi oluşturularak bazı matematiksel kural veya fonksiyonlar ile sınıflandırılma gerçekleştirilir (Vashisth ve Chandra, 2010). Şekil 4.14'te danışmasız öğrenme yapısı verilmektedir (Sağıroğlu ve ark., 2003). Bu metot çoğu uygulamada etkin olarak kullanılmaktadır. Kohonen'in kendini düzenleyen uzaylar (SOM) (Self Organizing Map)

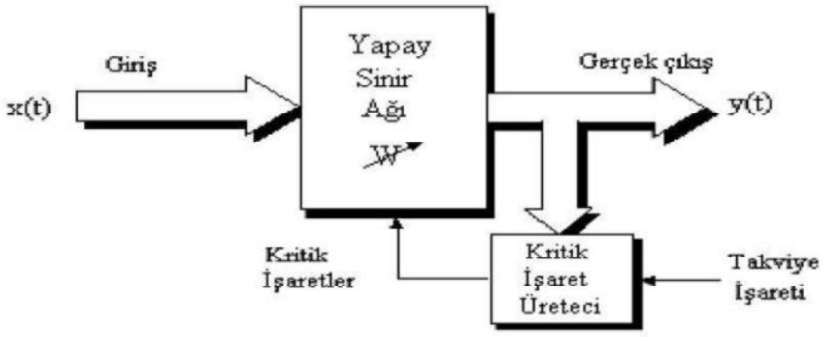
ve Grossberg tarafından geliştirilen adaptif rezonans teorisi (ART) (Adaptive Resonance Theory) öğrenme kuralı danışmasız öğrenme kuralına örnek olarak verilebilir (Civalek, 2004).



Şekil 4.14. Danışmasız öğrenme yapısı

4.5.3. Takviyeli öğrenme

Bir sistemi kontrol etmeyi öğrenmekle ilgili bir öğrenmedir. Takviyeli öğrenmede bir danışman bulunur; fakat danışmalı öğrenmedeki gibi sisteme çok bilgi vermez veya veremez. Sisteme sadece giriş değerleri verilerek, çıkan sonuçlara göre eldeki sonuçların karşılaştırılması yapılır. Yapılan karşılaştırma sonucunda sistem bir iyilik derecesi vererek ağın kendini eğitmesini sağlar. Şekil 4.15'te takviyeli öğrenme yapısı verilmiştir (Sağıroğlu ve ark., 2003).



Şekil 4.15. Takviyeli öğrenme yapısı

4.6. Yapay Sinir Ağlarında Öğrenme Kuralları

Öğrenme kuralları YSA' nın doğru sonuca ulaşmasında etkili olan yöntemlerdir. Ağırlıkların sonuçlara göre değiştirilmesi bu kurallar ile gerçekleştirilir. YSA' da sıklıkla kullanılan öğrenme kuralları Hebb kuralı, Hopfield kuralı, Kohonen kuralı, delta kuralı ve geri yayılım algoritması olarak sayılabilir.

4.6.1. Hebb kuralı

1949 yılında biyolojik sistematiğine bağlı olarak geliştirilen ilk öğrenme kuralıdır ve ismini buluşu gerçekleştiren Kanadalı bir psikolog olan Donald Hebb' ten almaktadır. Hebb kuralının işleyiş mantığı, eğer bir nöron başka bir nörondan giriş alıyorsa ve iki nöron da aktifse aralarındaki ağırlık değeri kuvvetlendirilir. Başka bir ifade ile aktif olan nöron bağlantılı olduğu nöronu da aktif hale getirmeye çalışacaktır (Saraç, 2004). Hebb kuralı, ilk öğrenme kuralı olduğundan bilinen tüm diğer öğrenme kurallarının temelidir. Hebb kuralı danışmasız öğrenme kuralıdır.

4.6.2. Hopfield kuralı

Hopfield kuralı, 1980'lerin başında fizikçi John Hopfield tarafından geliştirilmiştir ve Hebb kuralına benzerlik gösterir. Eğer beklenen giriş ve çıktıların her ikisi de aktif ya da pasif ise öğrenme katsayısı tarafından bağlantı ağırlığı değeri arttırılır ya da tersi durumlarda azaltırlır. Öğrenme katsayısı kullanıcı tarafından belirlenir ve çoğunlukla 0 ile 1 arasında değer almaktadır (Saraç, 2004).

4.6.3. Kohonen kuralı

Bu kural, Teova Kohonen tarafından geliştirilen, biyolojik sistemlerdeki öğrenme baz alınarak ortaya atılan öğrenme kuralıdır. En büyük ağırlığa sahip nöron çevresindeki nöronları sınırlama yetkisine sahiptir. Sadece en büyük ağırlığa sahip nöron ağırlıkları değiştirebilir. Kohonen kuralı, çıkış verisine gerek duymaz. Bu sebeple danışmasız öğrenme kuralıdır.

4.6.4. Delta kuralı

Widrow ve Hoff tarafından geliştirilen Delta kuralı, Hebb kuralının geliştirilmiş bir versiyonudur. En çok kullanılan kurallardan biri olup kuralın temelinde beklenen değerler ile ağın ürettiği değerler arasındaki fark, ağırlıklar değiştirilerek en küçük kareler yönteminde olduğu gibi en aza indirgenmeye çalışılır. Hata değeri aynı anda bir katmandan kendisinden önceki bir katmana geriye doğru yayılarak azaltılır.

Delta kuralı uygulanırken girdi setindeki verilerin rastgele dağılmış olmasına dikkat etmek gerekmektedir. Eğitim setinin düzgün bir biçimde olması, istenilen çıkış değerlerine ulaşılmasına engel olmaktadır ve ağın öğrenmesini zorlaştırmaktadır (Bayır, 2006).

4.6.5. Geri yayılım algoritması

Geri yayılım algoritması uygulamalarda çok fazla kullanılan öğrenme kuralıdır. Genelleştirilmiş delta kuralı olarak da bilinmektedir. İleri beslemeli ve çok katmanlı bir ağ mimarisi gerektirmektedir. Geri yayılım algoritmasında hata geriye doğru yayılır (Anonim, 2017). Hata değeri ağdaki ağırlıkların bir fonksiyonu gibi görülür ve hataların kareleri ortalaması delta kuralında olduğu gibi dereceli azaltma (gradient descent) yöntemi kullanılarak minimuma indirmeye çalışılır. Danışmalı öğrenme yöntemini kullanan geri yayılım algoritmasında örnekler ağa öğretilir ve ağa hedef bir değer verilir. Öğrenme işlemi her bir örnek için ağın çıktı değeri ile hedef değerini karşılaştırır. Elde edilen hata değeri büyük ise ağa tekrar geri besleme şeklinde verilir ve en aza indirgenmeye çalışılır (Curram ve Mingers, 1994).

5. GÖRÜNTÜ İŞLEME İLE AĞAÇ MEYVE YÜKÜNÜN HESAPLANMASI

5.1. Donanım ve Yazılımlar

Görüntü işleme ile ağaç meyve yükünün hesaplanması isimli çalışmada materyal olarak Matlab 2016a ve Android Studio yazılım ortamları kullanılmıştır. Çalışmada donanım olarak Android işletim sistemli, 1.2 GHz dört çekirdekli işlemciye, 1.5 GB RAM'e, 16 GB dahili hafızaya, 8 MP arka kamera ile 5 MP ön kameraya sahip Samsung Galaxy E5 cep telefonu ve Windows işletim sistemine sahip bilgisayar kullanılmıştır.

5.1.1. Matlab

Matlab (Matrix Laboratory), özellikle mühendisler ve bilim insanları için tasarlanmış bir programlama platformudur. Programlama dili olarak, matematiksel ifadelerin kullanımına izin veren matematiğe dayalı Matlab dilini kullanır. Matlab direkt olarak matris ve dizi matematiğini ifade eden bir programlama dili ile iteratif analiz ve tasarım süreçleri için ayarlanmış olan masaüstü ortamını birleştirir (Anonymous, 2018**)** .

Matlab; nümerik hesaplama, grafiksel veri gösterimi ve programlamayı içeren teknik ve bilimsel hesaplamalar için yazılmış yüksek performansa sahip ve dördüncü nesil programlama dilidir. İlk kez James Martin'in "Application Development without Programmers" adlı kitabında 1982 yılında dördüncü nesil programlama ifadesinden bahsedilmiştir. Dördüncü nesil programlar, kullanış açısından kullanıcıya çok daha kolaylık sağlayan, yine kullanıcının daha az kod yazmasını sağlayan yönergeler içeren, hazır şablonları ve sihirbazları sayesinde belirlenen ihtiyaçlarda kolay ve pratik çözümler geliştirmeye yönelik olan dillerdir. Matematik ve hesaplama işlemleri, algoritma geliştirme ve programlama problemleri, verilerin analizi, GUI uygulamaları, modelleme, simülasyon çalışmaları ve ürün örnekleme gibi kullanım alanlarına sahiptir.

Matlab, MathWorks şirketi tarafından geliştirilmektedir. Matlab, grafik kullanıcı ara yüzüyle görsel programlama imkanı vermektedir. C, C++, Fortran, Java gibi programlama dilleri ve MS Excel ile tümleşik çalışma özelliği bulunmaktadır. Gerçek dünya ile veri alışverişi yapılabilir ve kontrol sistemleri geliştirilebilir. Ayrıca, teknik hesaplama açısından yüksek düzeyli bir dil olduğundan dolayı kod, dosya ve veri yönetimi için geliştirme uygun bir geliştirme ortamıdır. Matlab, belirlenen bir problemin mimarisi ve bu problemin çözümünü elde etmek için çeşitli interaktif araçlar sunmaktadır.

Örneğin, lineer cebir, istatistik, fourier analizi, optimizasyon, integral vb. gibi konularda matematiksel fonksiyonlara ve veri gösterimi için 2 ve 3 boyutlu grafik fonksiyonlarına sahiptir (Özcan T., 2018).

Matlab programlama dilinin kullandığı çalışma ortamının dört temel bölümden oluşur.

1. Kullanılan Klasör (Current Folder)
2. Komut Ekranı (Command Window)
3. Çalışma Alanı (Workspace)
4. Komut Geçmişi (Command History)

Kullanılan Klasör: Projeler ve uygulamalar bu klasörde kaydedilir. Bir .m uzantılı dosya veya .m fonksiyon'u çağrıldığında, Matlab bunları bu klasörde arar. Bu klasöre resim, text gibi dosyalar da eklenebilir.

Komut Ekranı: Komutların girildiği ekran olan bu ekranda Matlab ile kullanıcı arasında bir ara yüz oluşturulur.

Çalışma Alanı: Matlab ile uygulama geliştirirken tanımlanan sabitler ve değişkenler çalışma alanında depolanır. Çalışma alanındaki sabit ve değişkenler Matlab'da çağrıldığı zaman çalışma boyunca değişkenler burada tutulur. Matlab programı sonlandırılırsa değişkenler silinir.

Komut Geçmişi: Bu ekran Matlab'da program yazılmasını basitleştiren bir pencere olarak tanımlanabilir. Daha önce çalışılan uygulamalarda var ise açılan komutlar bu ekranda gösterilmektedir. Tekrar tekrar aynı komutun kullanılması durumu oluşursa veya istenirse komut geçmişinde görülebilir ve o komut defalarca kullanabilir (Özel, H., 2018).

Gerçekleştirilen çalışmada sağladığı yüksek performans, kullanım kolaylığı, pratik ve hızlı çözüm geliştirme, görüntü işleme araçları bakımından zengin olması, ihtiyaçlarımıza kolay ve etkili bir etkili bir şekilde cevap vermesi gibi sebeplerden dolayı son zamanlarda yazılımcılar tarafından çok fazla tercih edilen ve her güncellemesi ile kendini geliştirmeye devam eden Matlab program dili tercih edilmiştir.

5.1.2. Android studio

Android 2003 yılında bir şirket olarak kurulduktan sonra günümüzde en çok kullanılan mobil işletim sistemi haline gelmiştir. Android işletim sistemi özellikle

dokunmatik mobil cihazlar, tabletler ve bilgisayarlar için Linux çekirdeğini kullanan ücretsiz ve açık kaynak kodlu bir platformdur. Android uygulamalar Java dili ile geliştirilmektedir. Uygulama uzantısı .apk olarak elde edilmektedir. Android sistemi içerisinde C dili yazılmış olan kütüphaneler, grafik işlemlerini yürüten OpenGL, ses ve görüntü işlemleri ile veri yapılarının düzenlenmesini sağlayan yapılar bulunmaktadır.

Bir android işletim sisteminde çalışan uygulama geliştirmek için özel olarak hazırlanmış Android Studio, Eclipse vb. ortamlar kullanılabilir. Android Studio ortamı Google tarafından resmi olarak duyurulan hızlı ve çok çeşitli araçlara sahip geliştirme ortamıdır.

Android uygulamalar geliştirmek için özel olarak hazırlanmış bir tümleşik geliştirme ortamı ve resmi programlama aracı olan Android Studio yaygın şekilde uygulama geliştirmek için kullanılmaktadır. Bu ortam sayesinde yazılım geliştiriciler kolaylıkla proje oluşturabilir, kod yazımı sırasında destek alabilir ve hata durumları vb. konularda daha basit bir çözüm üretebilmektedirler. Android Studio platformunda uygulamalar "Android yazılım geliştirme kiti (SDK)" kullanılarak Java dilinde yazılmaktadır. SDK, yazılımcıya hata ayıklayıcı, yazılım kütüphaneleri ve emülatör gibi yardımcı araçlar sunmaktadır.

Android Studio geliştirme ortamının kullanıcılara sağladığı temel özellikler şunlardır (Işıngör, 2018) :

- Gradle tabanlı olup esnek proje inşa sistemi
- Hızlı ve zengin özellikli bir emülatör
- Farklı özellik ve sürümlere göre çoklu APK çıktısı
- Genel uygulama özelliklerini oluşturmaya ve örnek kodu içe aktarmaya yardımcı olacak kod şablonları ve GitHub entegrasyonu
- C ++ ve NDK (Native Development Kit) desteği
- Sürükle-bırak özelliği ile ekran tasarımlarını kolaylaştıran zengin editör
- Uygulamanın performansı, kullanılabilirliği, farklı sürümlerde çalışabilirliğinin kontrol edilebileceği test araçları ve frameworkler
- Kolay ve güvenli APK imzalamasına sahip olması
- Ek uygulamalara gerek olmadan Google hizmetlerini uygulamaya ekleyebilme

Gradle, Android uygulaması geliştirme aşamalarını otomatize eden açık kaynak kodlu Android Studio üzerinde çalışan bir yapı sistemidir. Android Studio üzerinde bir proje açıldığında otomatik olarak gradle build sistemi devreye girmektedir ve derleme işlemini gerçekleştirmektedir (Beyler, 2017).

Android Studio ile ilgili en önemli özelliklerden bir tanesi sanal cihazlardır. Bu sanal cihazlar emülatör olarak da isimlendirilmektedir. Herhangi bir sistem üzerinde başka bir işletim sisteminin kullanılmasına emülasyon denir, buna imkan veren cihazlar da emülatör olarak adlandırılmaktadır.

Android uygulamalarını yüklemek ve dağıtımını yapmak için kullanılan dosya biçimine APK (Android Uygulama Paket Dosyası) (Android Application Package File) denilmektedir. Bu dosya biçimi sayesinde geliştirilen uygulamalar istenilen cihaza yüklenebilir ve uygulamaların yayılımı gerçekleştirilebilir.

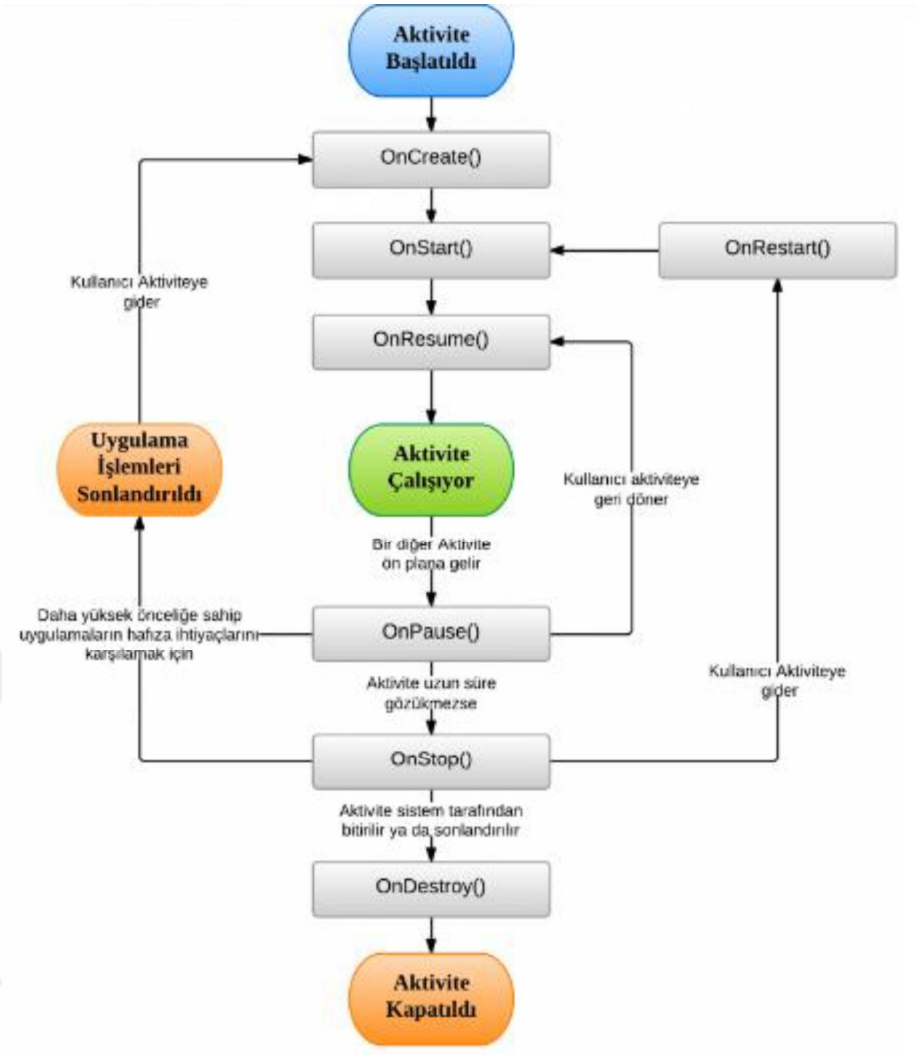
NDK, Android platformuna Native uygulamalar yazabilmek için hazırlanmış olan bir destektir. Android platformunda uygulamalar çoğunlukta Java dili kullanılarak yazılır ancak Android belirli oranda Native (C/C++) desteği de sağlamaktadır (Anonim, 2016).

Android Studio'da uygulama oluştururken kullanıcı ara yüzü oluşturmayı sağlayan sınıflara "aktivite" denilmektedir. Her uygulamanın *LAUNCHER* olarak tanımlı bir aktivite sınıfı olmalıdır. Bu şekilde tanımlanmış olan aktivite sınıfı uygulama ilk olarak açıldığı zaman kullanıcı ekranında görünmektedir. Örnek *LAUNCHER* tanımı verilmiştir . Bu bilgiler AndroidManifest.xml dosyasında tutulmaktadır.

```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER"
        </intent-filter>
    </activity>
</application>
```

Bir aktivite sınıfının işleyiş algoritması Şekil 5.1 ile verilmiştir (Yazar, 2013).

Bir Android projesinin en önemli dosyası AndroidManifest.xml dosyasıdır. Uygulamanın tüm ana bilgileri AndroidManifest.xml dosyası içerisinde belirtilir. Bu dosya .xml formatında olduğu için hem makina hem de insan tarafından okunabilme özelliğine sahiptir.



Şekil 5.1. Activity işleyiş algoritması

Android Studio'da Aktivite başlatıldığı zaman ilk olarak çalışan metod *onCreate (Bundle)* metodudur. Bu metod içinde *setContentView()* metodu kullanarak sınıf tasarımı belirtilmesi gerekmektedir. Uygulamalarda hazırlanan tasarımları (textview, label, button vs.) kullanmak için *R.java* sınıfı içerisinde oluşturulan id'ler kullanılır. Kaynakların erişimini sağlamak için id'leri *findViewById()* metoduna parametre olarak verilerek kaynaklara kazandırılmak istenen işlevsellikler bu kısımda tanımlanır (Yazar, 2013).

Android uygulamalarında ekran tasarımları "layout" dosyaları ile oluşturulur. Layout dosyaları ".xml" formatında oluşturulan dosyalardır ve Android uygulamalarına özel etiketler kullanarak görsel öğelerin yerleşimlerini ve özelliklerini sağlarlar. Örnek olarak "Görüntü işleme ile ağaç meyve yükünün hesaplanması" adlı tez çalışmasında kullanılan layout kullanım şekli aşağıda verilmiştir.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context="com.gamzeyasar.portakalbulma.MainActivity">
    <org.opencv.android.JavaCameraView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/cameraview"
         />
    <Button
        android:text="HESAPLA"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/button"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true" />
    <TextView
        android:text="Portakal Kilogram="
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/textView"
        android:layout_marginLeft="11dp"
        android:layout_marginStart="11dp"
        android:layout_alignBaseline="@+id/button"
        android:layout_alignBottom="@+id/button"
        android:layout_toRightOf="@+id/button"
        android:layout_toEndOf="@+id/button" />
    <TextView
        android:text="TextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/textView2"
        android:layout_marginLeft="17dp"
        android:layout_marginStart="17dp"
        android:layout_alignBaseline="@+id/textView"
        android:layout_alignBottom="@+id/textView"
        android:layout_toRightOf="@+id/textView"
        android:layout_toEndOf="@+id/textView" />
    />
</RelativeLayout>
```

Uygulama geliştirirken genel olarak ekranın görüntüsü genellikle iki çeşit yerleşim tipi kullanılarak belirlenir. Bu yerleşim tipleri “RelativeLayout” ve “LinearLayout” olarak ikiye ayrılır. LinearLayout yerleşiminde öğeler sırasıyla ekrana yerleşirler ve ekranda yerleşimleri ekranın en üst kısmından en alt kısmına doğru sırasıyla gerçekleşir. RelativeLayout yerleşiminde ise görsel öğeler diğer öğelerin konumlarını referans alarak dizilim gerçekleştirirler. RelativeLayout tasarımında ilk olarak eklenen öğe ekranın en üst noktasında yerini alır (Anonim, 2016).

Uygulama tasarlarken gereken öğeler Android Studio tarafından sağlanmaktadır. Android SDK ile gelen ve layout dosyasında kullanılan temel öğeler TextView, Button, EditText, ImageView, JavaCameraView, ListView vb. olarak sayılabilir.

TextView, bir yazıyı veya metni kullanıcıya göstermek için kullanılmaktadır. Bu yazı uygulamayı geliştiren tarafından sabit olarak belirlenebileceği gibi, uygulama içerisinde elde edilen bir değerin veya yazının gösterilmesi şeklinde de belirlenebilir. Görüntü işleme ile ağaç meyve yükü hesaplaması isimli çalışmada kullanılan TextView'lerden biri için sabit bir yazı değeri belirlenmiş olup, diğeri uygulama içerisinde hesaplanan portakal yükü değerini ekranda kullanıcıya göstermektedir.

Button, tıklandığı zaman istenilen bir işlevi yerine getirecek kodu çağıran görsel öğedir. Uygulama geliştiricilerin çok sık kullandığı Button tıklamaları dinleyiciler tarafından yakalanır ve tıklama işlemi gerçekleştiği zaman yapılması gereken işlemler dinleyici metotları içerisine tanımlanır.

JavaCameraView, OpenCV ve Java Camera arasında köprü kurarak kamera uygulamasının ekranda gösterilmesini sağlar. İçerisine yazılacak olan komutlar ile kameranın durdurulması, başlatılması, bitirilmesi gibi işlemleri gerçekleştirir (Anonymous, 2018).

Android işletim sistemini kullanarak geliştirilen uygulamalar insanın her türlü ihtiyacına yönelik olarak gelişmeye devam etmektedir. Gün geçtikçe kullanımı artan ve aktif bir şekilde kullanılan akıllı telefonlar ile insanlar günlük hayatta karşılaştıkları problemleri ve gereksinimleri geliştirilen çeşitli uygulamalar ile gidermektedir. Tez kapsamında gerçekleştirilen "Görüntü İşleme ile Ağaç Meyve Yükünün Hesaplanması" isimli uygulama günlük yaşamda tarım sektörü ile uğraşan insanların karşılaştıkları bir problemi ele almış ve bu problemi aşabilmek için çeşitli çözüm önerileri sunmuştur. Çalışmada uygulanan en önemli yenilik; görüntü işleme ile bulunan meyve yükünün android işletim sistemlerinde de kullanılabilir olmasıdır. Bu sayede android işletim sistemine sahip akıllı telefon veya tableti olan herkes gerçekleştirilen çalışmayı kullanabilecektir.

5.2.3. OpenCV

OpenCV (Open Source Computer Vision) açık kaynak kodlu bilgisayar görme ve makine öğrenimi yazılım kütüphanesidir. OpenCV bilgisayar görme uygulamaları için ortak bir alt yapı sağlamak ve ticari ürünlerde makine kullanımını hızlandırmak için

geliştirilmiştir. Kullanıcılar açık kaynak kodlarını kullanarak değişiklik yapabilirler. OpenCV kütüphanesi, 2500'den fazla hem klasik hem de son teknoloji görüntü işleme ve makine öğrenimi algoritmalarından oluşan kapsamlı ve optimize edilmiş algoritmalar içerir. Bu algoritmalara örnek verilecek olunursa; yüzleri algılamak ve tanımlamak, nesneleri tanımlamak, videolarda insan eylemleri sınıflandırmak, kamera hareketlerini izlemek, hareketli nesneleri izlemek, nesnelerin 3B modellerini ayıklamak, stereo kameralardan 3B noktalar üretmek, görüntüleri yüksek çözünürlükte birleştirmek, tüm bir blok görüntüsünü bulmak veya bir görüntü veritabanından benzer görüntüleri bulmak, flaş kullanarak çekilen görüntülerden kırmızı gözleri kaldırmak, göz hareketlerini takip etmek, manzarayı tanımak vb. 'dir. C++, Java, Python ve Matlab geliştirme ortamlarına entegre olabilir ve Windows, Linux, Android, Mac OS işletim sistemlerinde kullanılabilir. OpenCV çoğunlukla gerçek zamanlı uygulamalarda kullanılmaktadır. C++ dilinde yazılmış bir görüntü işleme kütüphanesidir (Anonymous, 2018).

OpenCV'i oluşturan en önemli bileşenler aşağıda verilmiştir. Bunlar ayrıca meyve yükü hesaplama uygulamasında kullanılan bileşenlerden bazılarıdır.

Core : Burada OpenCV'nin temel fonksiyonları ve matris, point, size gibi veri yapıları bulunur. Görüntü üzerinde çizim yapabilmek için metotları ve XML işlemlerini barındırır.

HighGui: Bu bileşen resmi ekranda gösterme, pencereleri yönetme ve grafiksel kullanıcı arabirimleri için gerekli olabilecek metotları barındırır.

Imgproc: Imgproc paketi, filtreleme operatörleri, kenar ve nesne belirleme, renk uzayı işlemleri, renk denetimleri ve eşikleme gibi hemen hemen tüm görüntü işleme fonksiyonlarını içine alan bir pakettir.

OpenCV ile görüntü işleme yapabilmek için alınan görüntünün *Mat* sınıfında tutulması gerekmektedir. Renk değerleri renk uzayı ile beraber sayısal olarak alınıp görüntü matris haline getirilerek bu sınıf içerisinde tutulur. *Mat* değişkeni oluşturma örneği aşağıda verilmiştir. Burada *public*, bir değişkeni, metodu veya sınıfı niteleyebilir. *Public* olarak tanımlanan öğeye kodun iç kısımları veya dışarıdaki farklı sınıflar da erişilebilir. Uygulamalarda *main()* metodu çoğunlukla *public* olarak tanımlanır. Benzer şekilde *private, protected* gibi tanımlanan değişken veya metotlar vardır. *Void*, main metodunun ekrana yazma işlemi haricinde herhangi değişik bir sonucu geri döndürmeyen metotları belirtmek için kullanılır. Adından da anlaşılacağı gibi *onCameraViewStarted* ile kamera uygulaması başlatılır. *Mat* içerisinde belirtilen width ve height değişkenleri,

kamera açıldığı zaman ekranda kapladığı en ve boy kadar görüntüyü tutacağını ifade etmektedir. *Cv.Type.CV_8UC4* olarak belirtilen değer ise veri tipini belirtir.

```
    public void onCameraViewStarted(int width, int height)
{
imgThresholded=new Mat(width,height,CvType.CV_8UC4);
}
```

Önerilen çalışmada OpenCV'nin sağlamış olduğu açık kaynak kodlu görüntü işleme kütüphanelerinin fazlalığı sebebiyle Android Studio geliştirme platformunun içerisine OpenCV kütüphaneleri eklenmiştir. Bu sayede Android Studio ortamında görüntü üzerinde çeşitli işlemler gerçekleştirilerek Android meyve programı oluşturulmuştur.

5.2. Yöntem

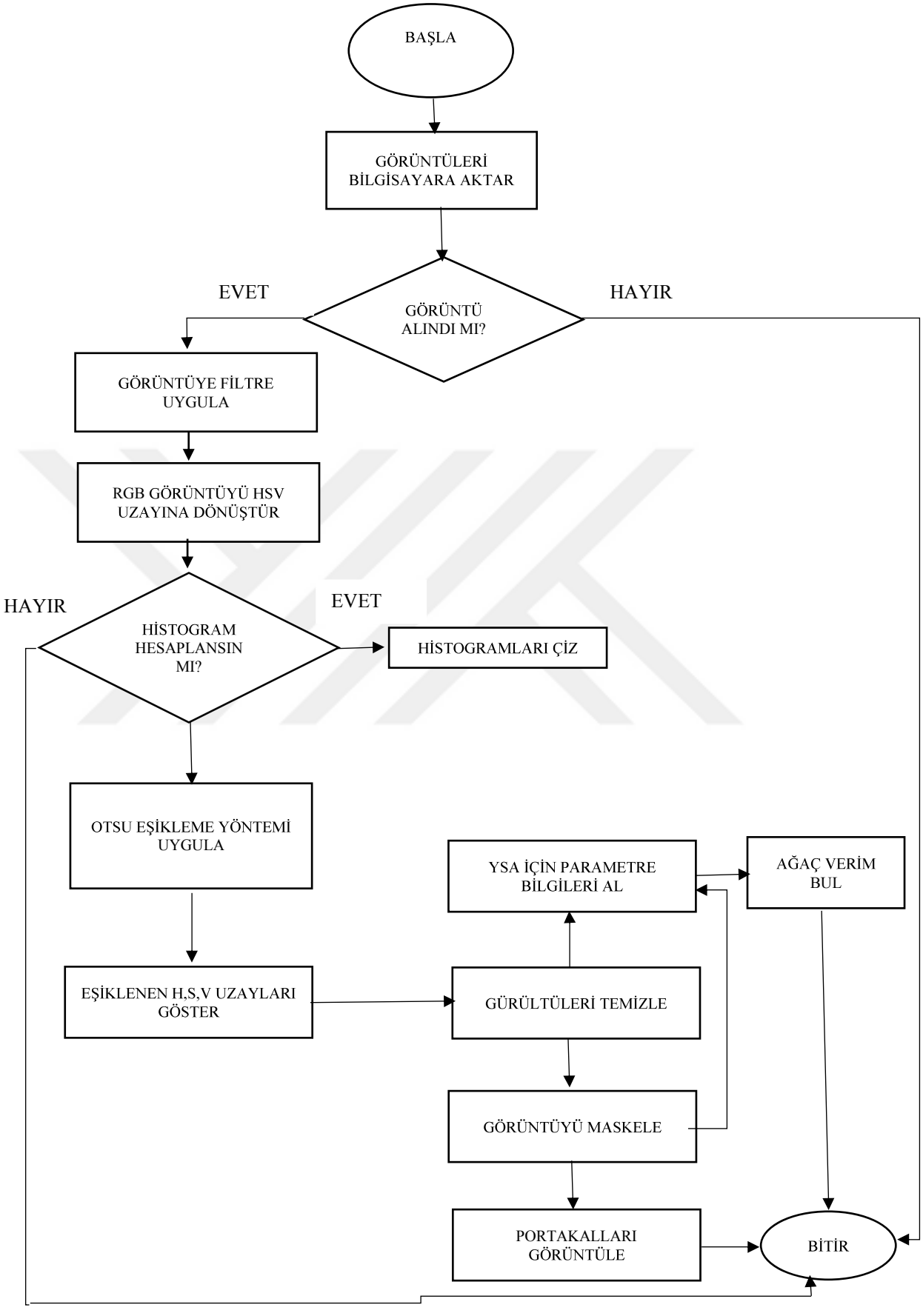
Görüntü işleme teknikleri kullanarak bir ağaç üzerindeki meyvenin hesaplanması mantığına dayanan çalışma iki farklı program geliştirme ortamı kullanılarak gerçekleştirilmiştir. Bunlardan ilki Matlab, ikincisi Android Studio'dur. Matlab ortamında gerçekleştirilen çalışma ile ağaç dalları üzerindeki meyvelerin görüntü işleme metotları ile meyve verim miktarı bulunmuş olup, gerçek hayatta uygulamanın kullanılabilmesini sağlamak amacıyla Android Studio platformunda uygulama OpenCV görüntü kütüphanesi kullanılarak yeniden düzenlenerek ağaç meyve yükü hesaplanmıştır. Bu sayede, daha önce gerçekleştirilmemiş olan, görüntü işleme uygulaması tarım sektörüne entegre haline getirilmiş ve kullanım kolaylılığı sağlaması amaçlanmıştır.

Çalışmada kullanılan görüntüler Antalya Serik bölgesindeki yerel bir çiftçiye ait portakal ağaçlarından farklı açılardan ağaç görüntüleri de olmak üzere 235 adet görüntü alınmıştır. Portakallar henüz dallarında iken bir cep telefonu ile fotoğrafları çekilmiştir. Sonrasında portakallar dallarından toplanmış ve meyvelerin toplam ağırlığı kantar yardımıyla tespit edilmiştir. Alınan görüntüler bilgisayar ortamında Matlab yazılım geliştime platformuna aktarılarak geliştirilen uygulama yardımıyla görüntü üzerinden portakallar segmentasyon yapılarak ayıklanmıştır. Sonrasında toplam ağaç alanı ile portakal meyvesi alanı arasında alan, boyut, ağırlık merkezi, renk vb. parametreler ile bir veritabanı oluşturulmuştur. Daha sonra veritabanı kullanılarak yapay sinir ağları

algoritması ile tahmini olarak hesaplanan meyve yükü ile ağaçtan toplanarak bulunan meyve yükü karşılaştırılmıştır.

İlk aşamada renkli RGB uzayındaki görüntüler HSV renk uzayına dönüştürülmüştür. İkinci aşamada HSV renk uzayları ayrıştırılarak her bir uzay katmanının histogramı hesaplanmıştır. Hesaplanan bu histogramlar grafik üzerinde gösterilerek, görüntünün renk dağılım bilgisi ortaya çıkarılmıştır. Üçüncü aşamada histogram grafiğinden elde edilen piksel bilgileri ile Otsu eşikleme metodu uygulanmıştır. Yeşil olan yaprak alanı ve portakal alanı yaklaşık olarak hesaplanarak YSA veritabanına eklenmiştir. Dördüncü aşamada elde edilen binary görüntüye gürültü temizleme gibi morfolojik işlemler uygulanmış olup, görüntü maskelenmiştir. Bu aşamada görüntü üzerindeki meyve alanı, çevresi, ortalama ağırlık merkezi bilgisi ve sayı bilgisi alınarak yapay sinir ağı yönteminde kullanılacak olan veritabanına eklenmiştir. Son aşamada meyve öznitelikleri ile yaprak alanı arasındaki ilişkiden yola çıkarak tahmini olarak hesaplanan meyve kilogramı bulunmuştur.

Matlab ortamında gerçekleştirilen çalışmanın akış diyagramı Şekil 5.2'de verilmiştir.



Şekil 5.2. Matlab meyve yükü hesaplama akış diyagramı

6. ARAŞTIRMA BULGULARI VE TARTIŞMA

Kültürümüzde bahçe ve tarım alanlarında yetiştiricilik yapan insanların bazı durumlarda ürünlerini tahmini olarak hesaplayıp toplu bir şekilde alıcıya sattığı bilinmektedir. Alıcı ve satıcı arasında tam net kurulamayan ürün çıktı hesabı iki taraf açısından da güvensizliğe sebep olmaktadır. Bu amaç ile ağaç meyve yükü hesaplaması matematiksel boyuta taşınarak hem alıcının hem satıcının güven duyabileceği bir uygulama önerilmiştir. Çalışma, gerçek hayata taşınarak etkili bir şekilde tarım uygulamasına Android sistem ile entegre edilmiştir. Daha önce literatürde bulunmayan bu yaklaşım çalışmanın en önemli noktasını göstermektedir .

Uygulanan çalışmada kullanılan portakal ağacı görüntüleri öğle saatlerinde gölgeli bir ortamda ve ikindi saatlerinde gölgesiz bir ortamda alınan görüntülerden oluşturulmuştur. Bahçe ortamında doğal aydınlanma koşullarıyla alınan görüntüler üzerinde sabit bir aydınlık şiddeti olmadığı için meyveleri %100 doğruluk ile tespit etmek mümkün olamamaktadır. Bu sebeple, Matlab ortamında geliştirilen uygulamada güneşli saatlerde çekilen parlak görüntüler üzerinde işlem yapmadan önce Lab uzayının L katmanında karartma işlemi uygulanmıştır. Aynı şekilde güneş ışığının yetersiz düştüğü fakat gölgenin olmadığı görüntülerde renk parlaklığı arttırılıp portakalların daha çok belirginleşmesi sağlanmıştır. Daha sonra bahsedilen görüntü işleme adımları uygulanarak ağaç yükü hesaplanmıştır. Matlab çalışmasında kullanılan gölgesiz ve karanlık olan görüntüler üzerinde kilo tahmininde, kantar ile ölçülen kilo – uygulama sonucu oranlanarak %89.8 başarı elde edilmiştir. Benzer olarak güneşli ve gölgeli bir görüntü üzerinde gerçekleştirilen kilo tahmininde %76.95 oranında başarı elde edilmiştir.

6.1. Matlab ile Görüntü İşleme

6.1.1. Görüntünün sayısallaştırılması

Bir ışık kaynağından çıkan ışık demetlerinin herhangi saydam olmayan bir nesneyi aydınlatması sonucunda nesneden yansıyan ışınlar bir kamera ile toplanarak analog bir sinyal haline getirilir. Analog sinyaller bir sayısallaştırıcı sistem tarafından sayısal sinyallere dönüştürülür. Böylece elde edilen görüntü bilgisayar ortamına aktarılarak işlenmeye hazır hale getirilmiş olur.

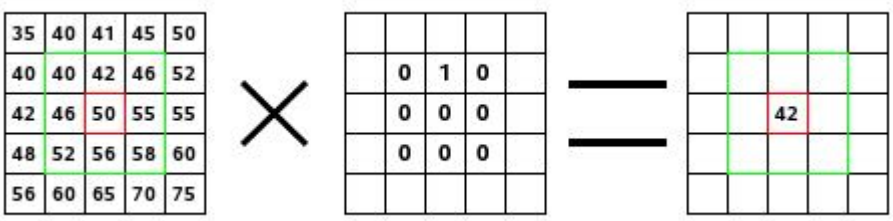
Çalışmada seçilen renkli ağaç görüntüsü bilgisayar ortamına aktarılıp, Matlab platformunda orjinal halde gösterilmiştir (Şekil 6.1). Görüntünün [1600x1200] matris şeklinde alınarak sayısallaştırma işlemi gerçekleştirilmiştir.

Şekil 6.1. Orjinal renkli görüntü

Orjinal renkli görüntü, gölge olmayan bulutlu bir havada ikindi saatlerinde alınmıştır.

6.1.2. Filtreleme İşlemi

Görüntü üzerinde ön işlem olarak gürültüleri temizlemek için bir filtreleme işlemine ihtiyaç duyulmuştur. Filtreleme aşamasında görüntü üzerinde yer alan gereksiz detaylar ve gürültü olarak nitelendirilen küçük pikselli siyah noktalar azaltılmaktadır. Filtreleme işlemleri seçilen bir pikselin ve komşu piksellerinin belirlenen herhangi bir matris ile çarpılması ile gerçekleştirilirler. Bu matrise kernel ismi verilmektedir. Kernel ile çarpım işlemi olan konvolüsyon işlemi Şekil 6.2 ile gösterilmektedir.



Şekil 6.2. Konvolüsyon işlemi

Filtreleme işlemleri için ortalama (mean) filtresi, medyan filtresi, Gaussian filtresi ve Gabor filtresi gibi filtreleme teknikleri kullanılabilir (Chang ve ark., 2003). Birçok filtreleme tekniği boyutsal alandaki (domain) geleneksel tekniklerdir. Filtreleme teknikleri lineer ve lineer olmayan filtreler olarak sınıflandırılabilirler. Bir lineer filtre olan ortalama filtresi, her pikselin komşularına ait piksel değerlerinin ortalamaları ile değiştirilmesi mantığına dayanan bir yöntemdir. Bu yöntem uyuşmazlıkları azaltır ve uygulaması kolay olan bir yöntemdir. Resimde yumuşatma ve bulandırma etkisi yaratır. Lineer olmayan filtrelere örnek olarak da sıra istatistiği (order statistic) filtresi verilebilir. Medyan filtresi, bu filtrenin özel bir durumudur. Kenar seçiciliğini korur ve ortalama filtresine oranla daha az bulanıklığa neden olur (Cheng ve ark., 2010; Güraksın, 2015).

Orjinal görüntü üzerinde gürültüleri azaltmak ve görüntüyü yumuşatmak için çalışmada kullanılan görüntüye ortalama filtresi uygulanmıştır. Ortalama filtresi, görüntünün her bir piksel değerini komşu piksel değerleriyle ve kendisinin dahil olduğu ortalama değer ile değiştirmektedir. Bu sayede çevresindekileri temsil etmeyen piksel değerlerinin ortadan kalkmasına yol açar. Ortalama filtresi bir konvolüsyon filtresidir. Çoğunlukla 3x3'lük bir kernel ile işlem tercih edildiği için uygulamada 3x3'lük bir kernel kullanılmıştır (Şekil 6.3). Ortalama filtresi ile orjinal görüntüye kıyas ile kenar pikselleri daha yumuşak bir görünüm kazanmıştır.

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Şekil 6.3. 3x3 Ortalama kernel filtresi

Uygulamada kenar niteliklerinin belirginliğinin azalması sonucu etkilemeyeceği için istenilen gürültü azaltma elde edilmiştir. Orjinal görüntü ve filtre işlemi uygulanmış görüntü Şekil 6.4 ile verilmiştir.

Şekil 6.4. Solda orjinal renkli görüntü, sağda filtre sonrası görüntü

6.1.3. RGB renk uzayından HSV renk uzayına dönüşüm

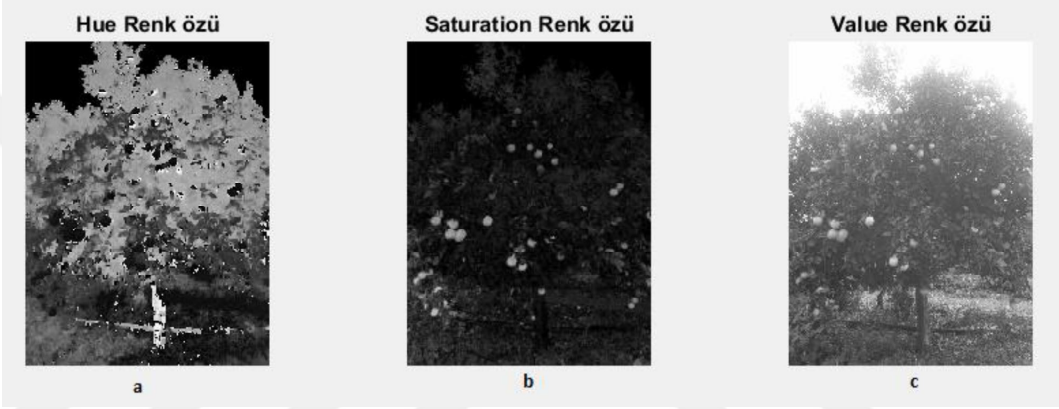
RGB renk uzayı görüntü işleme uygulamalarında yaygın şekilde kullanılan bir renk uzayı olmasına rağmen insan görme mekanizmasının görüş sağlama mantığına uygun değildir. Bu sebepten ötürü farklı iki veya daha fazla rengi RGB renk uzayı koordinatları kullanarak ayırt etmek mümkün olmamaktadır. Ayrıca cihaz bağımlı bir renk uzayı olduğu için her cihazda farklı renk sonuçları gösterir.

HSV renk uzayı insan görme sistemine daha yakın olduğu için görüntü işleme uygulamalarında bir görüntünün filtrelenmesi, renk seçimi gibi uygulamalarda sıklıkla tercih edilmektedir. Bu sayede, örneğin sadece hue bileşeninin değerini kullanarak eşikleme işlemleri gerçekleştirilebilir.

Bir rengin parlaklığı, doygunluğu gerekli ise veya segmentasyon işlemi gibi durumlarda RGB uzayından HSV uzayına dönüşüm yapılması nesnelerin arka plandan ayrılması için kolaylık sağlar. RGB uzayından HSV uzayına dönüşüm işlemleri doğrusal olmayan matematiksel denklemler ile gerçekleştirilir. RGB değerleri 0 ile 255 değerleri arasından 0-1 değerleri arasına normalize edildikten sonra, diğer bir deyişle görüntü tipi “uint8” den “double” tipine dönüştürüldükten sonra dönüşüm işlemi gerçekleştirilir.

Çalışmada seçilen görüntünün HSV renk uzayına çevrilip çevrilmeyeceği kullanıcıya sorulduktan sonra cevap ‘Evet’ ise görüntünün renk uzayı dönüşümü yapılmaktadır. Cevap ‘Hayır’ ise programdan çıkılmaktadır. Kullanıcının yanıtına göre bir döngü oluşturulmasının sebebi, görüntü üzerinde sonraki aşamalarda yapılacak olan işlemlerin zaman alması ve belleği doldurması sebebiyle herhangi bir nedenden ötürü görüntü seçimi değişimi yapılmak istenirse zaman ve bellek kaybı en az düzeye indirilerek kullanıcının vazgeçmesini sağlayabilmektir.

RGB görüntüsü HSV renk uzayına dönüştürüldükten sonra HSV uzayının her bir renk katmanı ayrı ayrı ekranda gösterilmiştir (Şekil 6.5).



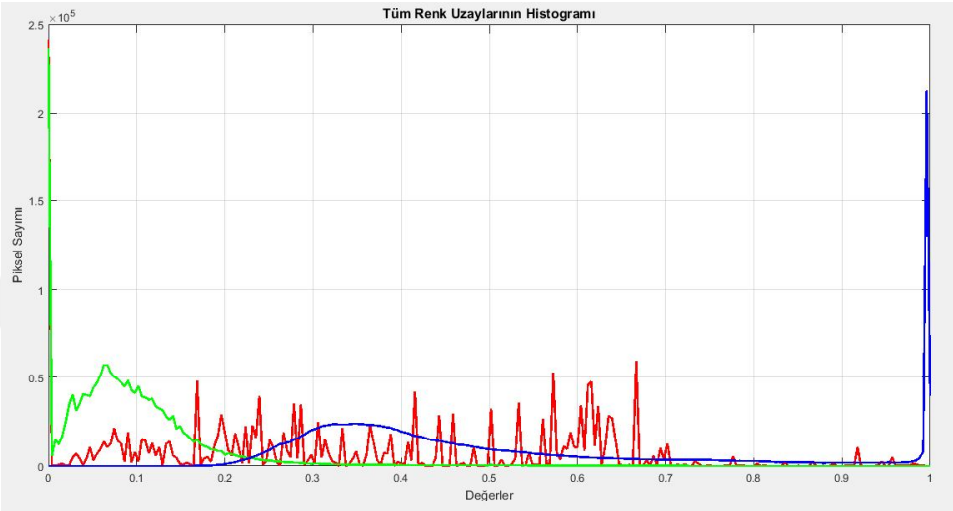
Şekil 6.5. a. Hue renk özü, b. Saturation renk özü, c. Value renk özü

6.1.4. Histogram hesaplama

Histogram, sayısal görüntünün içerisindeki piksellerin adet olarak dağılımını gösteren grafiğe verilen isimdir. Histogram sayesinde görüntüde öne çıkan nesnelerin, parlaklığı ayarlama, kontrast genişletme, görüntü iyileştirme gibi görüntü işlemede sıklıkla kullanılan işlemleri gerçekleştirebilmek için görüntü hakkında bilgi edinilir. Histogram grafiğinde yalnızca piksellerin sayıları gösterilir, koordinat bilgileri saklanmaz.

HSV renk uzayına dönüştürülen görüntünün her bir renk uzayı ayrı şekilde gösterildikten sonra elde edilen bu renk uzayları hakkında renk dağılımı bilgisi edinebilmek için hue, saturation ve value uzaylarının histogramları hesaplanmıştır. Histogramları hesaplanan renk katmanlarının grafikleri Şekil 6.6’da verilmiştir. Ayrı olarak hesaplanan renk uzaylarının histogramı daha sonrasında tek bir grafik üzerinde birleştirilmiştir (Şekil 6.7).

Şekil 6.6. a. Hue histogram grafiği, b. Saturation histogram grafiği, c. Value histogram grafiği



Şekil 6.7. HSV görüntüsünün histogram grafiği

6.1.5. Eşikleme İşlemi

Eşik değeri belirleme, görüntü işleme tekniklerinden en temel ve basit olan yöntemdir. Eşik değer belirleme işlemini gerçekleştirirken, her bir piksel bir nesne gibi değerlendirilir ve gri değerinin belirlenen eşik değerinden büyük olup olmadığı incelenir. Genellikle büyük olması durumunda, 1 yani nesne, olmaması durumunda ise 0, yani arka plan olarak değerlendirilir (Acar, 2011).

Farklı veri türlerini değerlendirmek için farklı eşikleme yöntemleri kullanılmaktadır. Çalışmada görüntüyü eşiklemek için ilk önce manuel olarak klasik eşik değeri belirleme uygulanmıştır. Seçilen eşik değerleri;

```
hueLow=0.18, hueHigh=0.7
saturationLow=0, saturationHigh=0.2
valueLow=0.35, valueHigh=1
```

olarak belirlendiğinde eşiklenen görüntü Şekil 6.8'de verilmiştir. Seçilen eşik değerleri;

```
hueLow=0.1, hueHigh=0.5
saturationLow=0, saturationHigh=0.3
valueLow=0.45, valueHigh=1
```

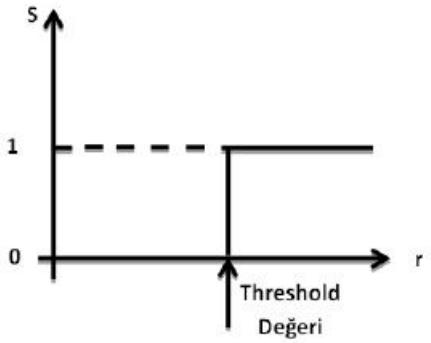
olarak belirlendiğinde ise eşiklenen görüntü Şekil 6.9 ile gösterilmektedir.

Manuel olarak eşik belirlenmesi karmaşık resimlerde zor ve doğru sonuç vermediği görüldüğünden dolayı Nobuyuki Otsu tarafından önerilen 'Otsu Metodu' kullanılmıştır (Otsu, 1979).

Gri görüntünün parlaklık değeri, alındığı aydınlatma koşullarına ve çevresel koşullara göre değişiklik göstereceğinden dolayı eşik seviyesinin her görüntü için farklı şekilde belirlenebiliyor olması önem kazanır. Otsu' nun bölgesel eşikleme algoritması bu değişken eşik seviyesini belirlemek için kullanılmaktadır. Belirlenen eşik seviyesi 0-1 aralığında olan bir parlaklık parametresidir. Bu eşik değerini belirlendikten sonraki adımda gri görüntü siyah beyaza çevrilmektedir (Acar, 2011). Eşik değerinden yüksek gri seviye değeri olan pikseller 1, küçük olan pikseller 0 değerini alır (Şekil 6.10).

Şekil 6.8. Manuel eşik değeri ile eşiklenen binary görüntü-1

Şekil 6.9. Manuel eşik değeri ile eşiklenen binary görüntü-2



Şekil 6.10. Eşik değeri seçimi

HSV renk uzaylarının her renk bandına ayrı olarak alt ve üst seviyeleri belirterek eşik seviyesini bulabilmek için bir Otsu yöntemi olan *graythresh()* komutu kullanılmıştır.

```
hueLow = 0;
hueHigh = 0.3569;
saturationLow = 0.15;
saturationHigh = 1.0;
valueLow = 0.6471;
valueHigh = 1.0;
```

Yukarıda verilmiş olan alt ve üst sınır eşik değerlerine göre elde edilen her bir renk uzayının binary görüntüsü birleştirilerek HSV görüntünün uint8 tipindeki görüntüsü elde edilmiş olup Şekil 6.11 ile gösterilmektedir.

Şekil 6.11. Otsu yöntemi ile eşiklenmiş görüntü

Elde edilen eşiklenmiş görüntü üzerinde saptamak istediğimiz nesneler haricinde küçük nesneler de olduğu görünmektedir. Bir görüntü üzerinde segmentasyon işlemi yapmak istediğimiz zaman nesneler tam olarak arka plandan belirgin şekilde ayrılmıyorsa ve üzerinde çalışılan görüntü doğal aydınlanma koşulları altında elde edilmiş ise bu durum ile karşılaşılması çok muhtemeldir. Doğal bahçe ortamında gerçek bir görüntü üzerinde işlem yaptığımız için portakal piksellerinin gün ışığının yansımasıyla yaprak alanlarında görülmesi ve küçük gürültülerin olması durumunun üstesinden gelebilmek için görüntüye bazı morfolojik işlemler uygulamamız gerekmiştir.

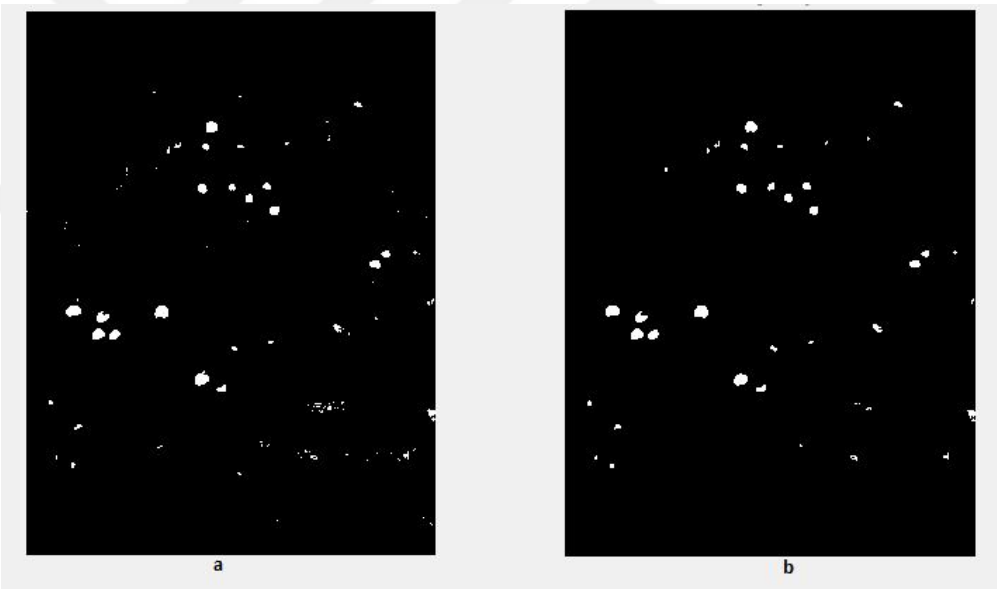
6.1.6. Morfolojik işlemler

Matematiksel bakımdan morfoloji, lineer olmayan komşuluk işlemlerinde etkili olan görüntü işleme analizi olarak tanımlanabilir. Morfolojik görüntü işlemlerinde genişletme ve aşındırma adında temel iki işlem sıklıkla kullanılır. Bilinen açma ve kapama işlemleri gibi diğer tüm görüntü işleme yöntemleri genişletme ve aşındırma işlemlerini referans alarak gerçekleştirilir. Daire, kare, oval gibi geometrik şekiller ile yapısal filtre uygulanan görüntü üzerinde açma veya kapama gibi morfolojik işlemler uygulanabilir. Fakat bu işlem için öncelikle genişletme veya aşındırma uygulanacak

görüntü ikili (binary) formata çevrilmelidir. İkili formattaki görüntüler üzerinde istenilen türde aşındırma ve genişletme işlemleri gerçekleştirilebilir (Atalı ve ark.).

Görüntü işlemede morfolojik işlemler görüntünün kusurlarını düzeltmek, nesneleri belirginleştirmek gibi amaçlar ile kullanılmaktadır. Bu sebeple uygulamada elde ettiğimiz binary görüntüdeki portakal olarak belirlenmeyen küçük nesneleri temizlemek için görüntüye morfolojik işlemlerden gürültüleri yok etme, genişletme ve aşındırma, kapama, boşlukları doldurma morfolojik işlemleri uygulanmıştır.

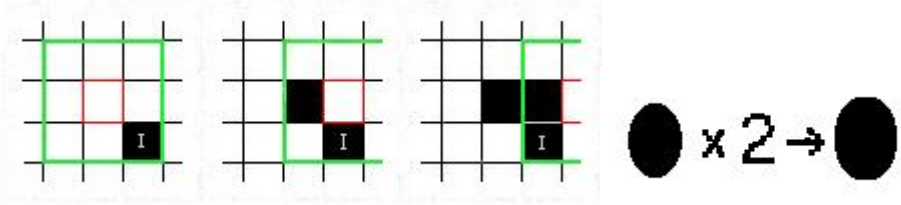
Çalışmada en son elde edilen binary görüntünün temizlenmesi için belirli bir piksel değeri seçilerek portakal olarak görünmeyen nesneler ortadan kaldırılmıştır. Piksel değerinin belirlenmesi deneme yanılma yolu elde edilmiştir. En doğru sonuca göre piksel değeri belirlendikten sonra morfolojik bir işlem olan *bwareaopen()* komutu ile binary görüntü birbirine bağlı olmayan küçük nesnelerden temizlenerek daha temiz bir görüntü elde edilmiştir. İşlem öncesi ve sonrasına ait görüntüler Şekil 6.12'de verilmiştir.



Şekil 6.12. a. Görüntü morfolojik temizleme öncesi, b. Görüntü morfolojik temizleme sonrası

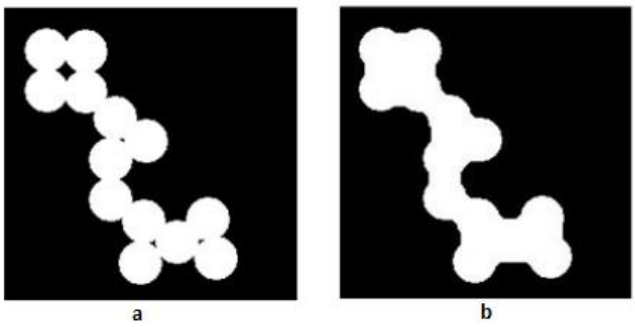
Genişletme (dilation) işlemi ikili görüntüdeki bulunan nesneleri büyütmek veya ortaya çıkmasını sağlamak için kullanılan bir yöntemdir. Uygulamada binary görüntü üzerine kernel matrisi yerleştirilmiştir. Kernel merkezi binary görüntüde herhangi bir piksele denk gelirse kernelin tüm elemanları binary görüntüye yerleştirilir. Kernel tüm olası pozisyonlara çekildikten sonra binary görüntü ile örtüşen pikseller varsa o pikseller genişlemenin içerisine alınır.

Şekil 6.13 ile verilen örneğe bakacak olursak, görüntünün incelenen pikselinin kırmızı bir sınırı, 3x3 boyutlu kernelin yeşil sınırı vardır. “I” pikseli yeşil sınırın yani kernelin içerisinde olduğunda incelenen piksel 0 değerini alır. Böylece nesnelerin genişletme işlemi gerçekleştirilir (Anonymous, 2018).

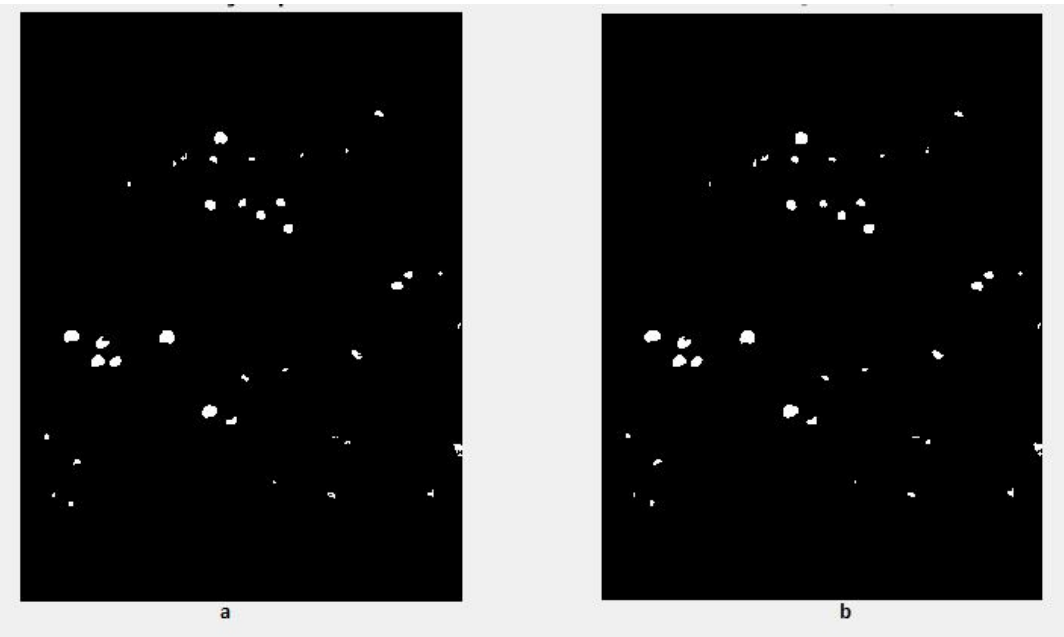


Şekil 6.13. Genişletme işlemi

Genişletilmiş olan binary görüntünün aşındırılması işlemine kapama işlemi denilmektedir (Şekil 6.14). Elde edilen görüntüdeki küçük boşlukları doldurmak için bu yöntem kullanılmıştır. Kapama yöntemi disk şeklindeki kernel ile uygulanmıştır. Bu sayede nesneler yumuşatılmış, uzun ve ince olan boşluklar doldurulmuş, küçük boşluklar kapatılmıştır. Kapama işleminden önce ve sonra maskelenen görüntü karşılaştırılması Şekil 6.15’te gösterilmektedir.

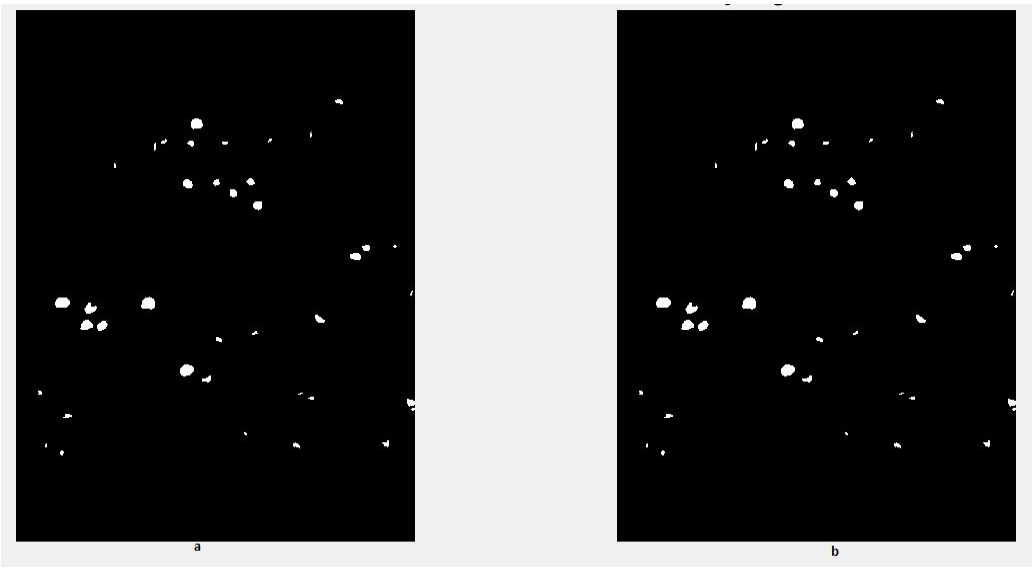


Şekil 6.14. a. Kapama işlemi öncesi bir görüntü, b. Kapama işlemi sonrası bir görüntü



Şekil 6.15. a. Genleşme ve kapama öncesi, b. Genleşme ve kapama sonrası

Genleşme ve kapama işleminden sonra görüntü üzerinde belirlenen nesnelerde herhangi bir boşluk kalması durumunun önüne geçebilmek için bu çukur olarak isimlendirilen boşlukları doldurmak için yeniden morfolojik bir işlem olan *imfill()* komutu uygulanmıştır. Bu sayede hem belirlenecek olan kenarların tespiti kolaylaşmış, hem de görüntü üzerinde boşluğa sahip bir nesne kalmamıştır. Bu işlem öncesinde ve sonrasında maskelenen görüntü Şekil 6.16'da gösterilmektedir.



Şekil 6.16. a. Çukurlar doldurulmadan önce, b. Çukurlar doldurulduktan sonra

6.1.7. Yapay sinir ağları ile kilogram tahmini

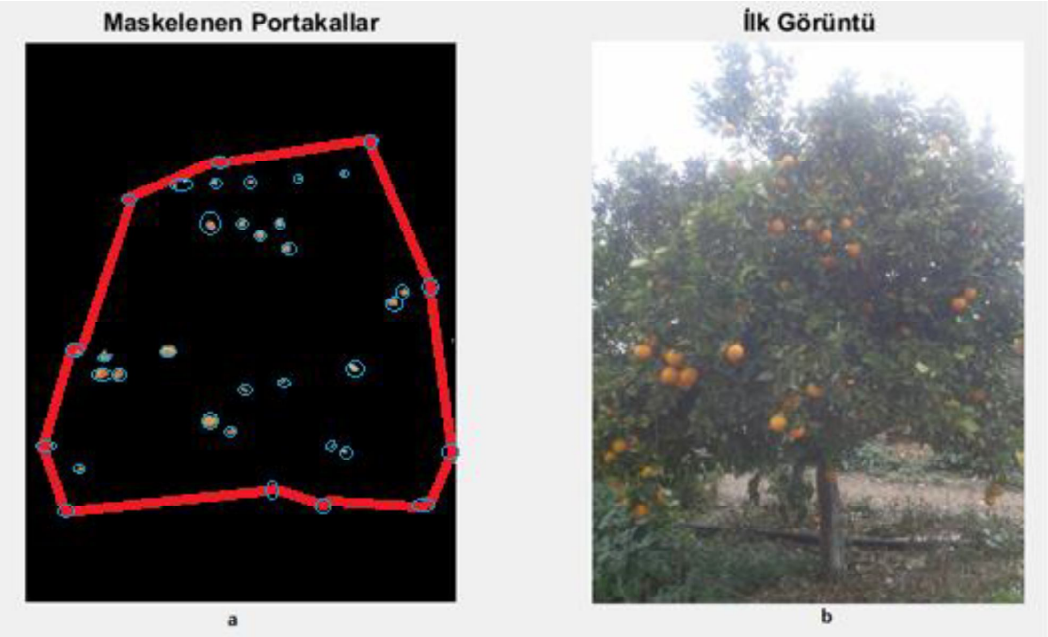
Yapay sinir ağları öğrenebilme ve genelleme özellikleri sayesinde görüntü ve ses tanıma, tahmin etme, haberleşme, medikal alan vb. matematiksel olarak problemlerin ifade edilebildiği hemen hemen her alanda geniş bir uygulama sahasına ulaşmıştır. YSA, kendisine gösterilen örnek veri setleri ile eğitilir. Bu örnek veri setlerinin bir kısmı ile öğrenme işlemi gerçekleşir. Diğer kısmı ise öğrenme işlemini test etmek amacı ile kullanılır. Genel olarak bu oran %80 eğitim seti, %20 test seti olarak ayrışır. Verilen test seti ile giriş olarak verilen eğitim seti uyuşmuş ise ağ eğitilmiş demektir.

YSA, sinir hücrelerinin birleşmesi ile elde edilir. Bu birleşme katmanlar halinde ve her katman içinde paralel bir şekilde gerçekleşir. Giriş katmanında, kullanıcıdan giriş verileri alınıp ara katmana iletilir. Bu zorunlu bir uygulama olmamakla birlikte çoğunlukla giriş katmanında veri veya veriler olur. Ara katmanda, giriş katmanından gelen veriler işlenerek çıkış katmanına iletilir. Yapılacak olan işlemlerin tamamı bu katmanda gerçekleşir. Uygulamanın ihtiyacına göre katman sayısı belirlenir. Çıkış katmanında ara katmandan işlenmiş olarak gelen bilgiler, kullanıcıya eğitim setine göre çıktıları işlenerek gönderilirler.

YSA dış ortamdan aldıkları verilere cevap olarak gösterdikleri davranışlarını değiştirebilirler. Öğrenmede sinir ağına bir giriş değeri verildiği zaman ağ sabit cevaplar üretebilmek için kendini yenilemek durumundadır. Bunun için YSA'yı eğitebilmek adına çeşitli öğretme algoritmaları geliştirilmiştir. Geliştirilen algoritmaların her birinin kendine göre iyi ve eksik olduğu noktalar bulunmaktadır. Öğrenme işlemi tamamlanmış bir YSA bir noktaya kadar ona verilen girdilerdeki ufak değişikliklere karşı duyarsız kalabilir. Bu durum ağın sürekli benzer çıktığı göstereceğini ifade eder ve gerçek dünyadaki çevreden gelen faktörlerle ufak bozulmalara uğramış girdileri farkedebilmek açısından ağın genellemesi için önemlidir. Bu yaklaşım klasik bilgisayarlarda kullanılan temel mantığı aşarak içinde yaşadığımız mükemmel olmayan dünyayı anlayabilmek için geliştirilmiş bir sistemdir. YSA genelleme işlemini kendi yapısal özelliklerinden dolayı otomatik olarak gerçekleştirirler (Akkaya, 2007).

YSA modeli oluştururken ara katman sayısı, bu katmanda yer alacak olan nöron sayısı, aktivasyon fonksiyonu seçimi gibi parametrelerin belirlenmesi için kesin bir yöntem olmadığından dolayı deneme yanılma yolu ile bu parametreler belirlenmektedir (Öztemel, 2003).

Çalışmada kameradan alınan görüntüler üzerinde arka plandan portakallar ayrıştırılarak görüntü ikili formata çevrilmiştir. Bu işlem sonrasında ikili görüntü üzerinde bulunan portakalların bazı parametreleri elde edilerek YSA'ya giriş olarak verilmiştir. Bu parametreler toplam kapalı alan bilgisi, portakal adeti, toplam portakal alanıdır. Bulunan portakal sayısına göre parametlerin sayısı değişmektedir. Bu parametreler sütun matris haline getirildikten sonra ağa giriş olarak tanımlanmıştır.



Şekil 6.17. a. En dış portakal sınırları belirlenmiş görüntü, b. Orjinal görüntü

Şekil 6.17 ile portakal ağacının bulunan toplam kapalı alanı ve toplam portakal alanı ile orjinal görüntü verilmiştir. Mavi çemberler portakal büyüklüklerini ve portakalların toplam alanını belirleyecek şekilde YSA'a verilmiştir. Şekil 6.17.a'da kırmızı ile belirlenen sınırlar YSA'a uygulanan ağaç sınırlarını belirleyen çizgilerdir. Bu sınır en dıştaki portakalların oluşturduğu kapalı bir eğridir. YSA'dan beklenen dış alan ile portakalların oluşturduğu toplam alan ile bir ilişki kurarak meyve ve yaprak yükünün ayırt etmesi ve olası meyve ağırlığını tahmin etmesi beklenmektedir.

Ağ çıkış hedef parametresi olarak, bir ağaçtan manuel olarak toplanıp elde edilen ortalama kilogram bilgileri verilmiştir. Oluşturulan ağ için ileri beslemeli YSA kullanılmış olup, ağın öğrenmesi için danışmalı öğrenme algoritması olan Levenberg Marquardt öğrenme algoritması seçilmiştir. Uygulamada öğrenme hatasının düşük olması nedeniyle bu algoritma tercih edilmiştir. İleri beslemeli yapay sinir ağlarında en düşük hata ile çıkış değerleri tahmin edilmeye çalışılır.

Levenberg Marquardt öğrenme algoritması maksimum komşuluk fikri üzerine kurulmuştur ve basit gradyant azalan metodu ve Newton iterasyonunu kullanmaktadır (Ranganathan, 2004). Hata geriye yayılım algoritmasından farklı olarak parametre güncelleme işlemleri için tüm girdi verileri için oluşturduğu hata vektörünü ve Jacabian matrisini kullanarak yapmaktadır (Çavuşlu ve ark.).

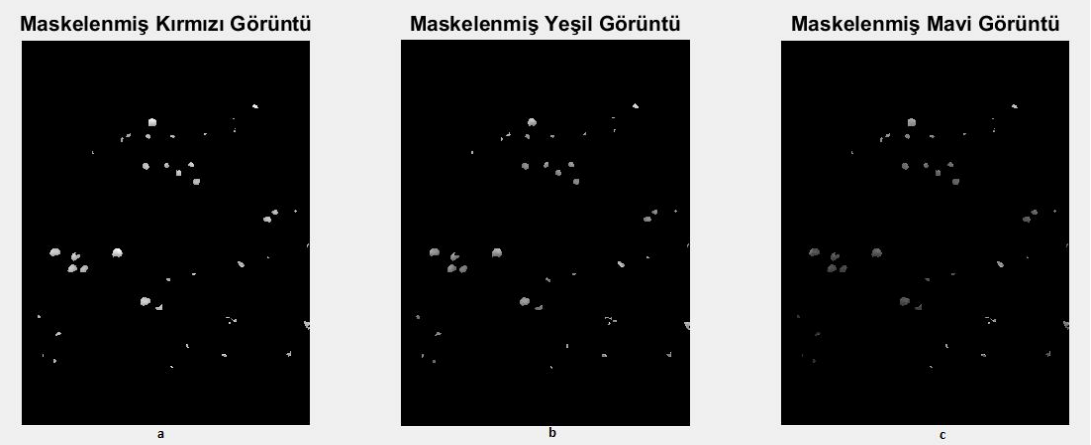
Matlab programında YSA toolbox'ı kullanarak oluşturulan YSA modeli için, bir giriş katmanı, bir gizli katman ve bir çıkış katmanı belirlenmiş olup ağın performansının düşük kalma durumuna göre gizli katman sayısı arttırılması hedeflenmiştir. Transfer fonksiyonu olarak PURELIN doğrusal aktivasyon fonksiyonu kullanılmıştır ve gizli katmandaki nöron sayısı 10 olarak belirlenmiştir. Kilogram tahmini için öncelikli olarak bir ağacın üzerinde bulunan 39 adet portakalın alan, piksel yoğunluk gibi parametreleri giriş parametreleri olarak verilmiştir. Aynı ağacın arka tarafından elde edilmiş olan görüntüsünden elde edilen 25 adet portakalın parametreleri ise test seti olarak belirlenmiştir. Hedef seti olarak manuel olarak belirlenen portakal parametreleri girilmiştir. YSA verilen eğitim girdilerini kullanarak gerekli öğrenmeyi sağlandıktan sonra öğrenme setindeki girdilerin ağa uygulanması ile öğrenme işlemi gerçekleşmektedir. Öğrenme esnasında ağırlıklar rastgele alınmıştır. Öğrenmeye bağlı olarak ağırlıklar güncellenerek belirlenen hedef hata düzeyine veya maksimum iterasyon sayısına ulaşıldığı zaman öğrenme ve ağın eğitilmesi bitmektedir.

6.1.7. Maskelenen görüntüyü RGB görüntüye dönüştürme

Morfolojik işlemler uygulanan binary görüntü sadece portakal nesneleri olduğunu bildiğimiz beyaz yuvarlaklar olarak görünen nesneler ile maskelenmiştir. Çalışmada, binary olarak maskelenmiş görüntü yeniden RGB renk uzaylı olarak görüntülenmiştir. Bu işlemi gerçekleştirebilmek için bir takım dönüşüm fonksiyonları kullanılmıştır.

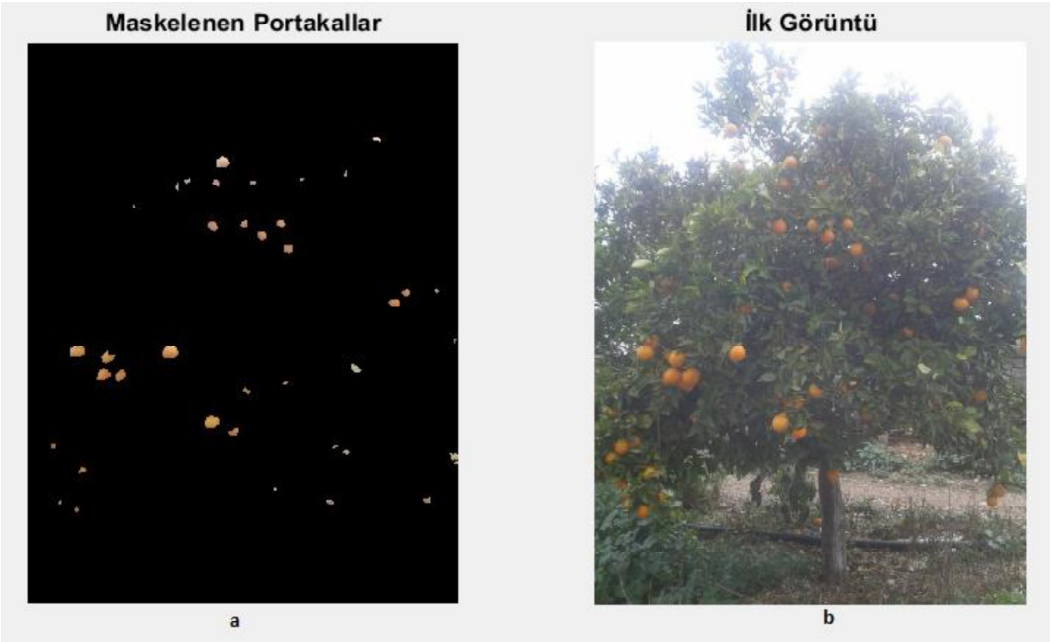
Binary formatta maskelenmiş olan görüntü 0 ve 1 değerlerinden oluşan bir mantıksal dizidir. Öncelikle bu lojik ifadenin RGB görüntüsü ile aynı türe dönüştürülmesi gerekmektedir. Bu dönüşümün sebebi aynı türden değişkenler ile matematiksel işlem yapıldığından dolayıdır. Maskeleme işleminin RGB görüntü üzerinde yapılabilmesi için çarpma işlemi gerçekleştirilmiştir. Bu işlem öncesinde de maskelenen görüntü uint8 tipine dönüştürülmüştür. Bu ayrıca verileri daha hızlı işlemeye ve daha az hafızada bellek tutulmasını sağlamıştır. uint8 veri tipi 0-255 arasında değişen tam sayı değerleri alan bir görüntü tipidir. İsmindeki int kısaltması tam sayı içerdiğini, 8 rakamı 8 bitlik değer

saklandığını gösterir. Bu şekilde pikseller maksimum 255 değerine kadar gri seviye tam sayı ile ifade edilirler. Gri seviye görüntü üzerine maskelenmiş olarak resmin her bir R, G ve B katmanlarının çıktıları Şekil 6.18 ile verilmiştir.



Şekil 6.18. a. Maskelenmiş R uzayı, b. Maskelenmiş G uzayı, c. Maskelenmiş B uzayı

Maskelenmiş görüntüler birleştirilerek tek bir görüntü haline getirilmiştir. Bu işlem sonrasında belirlenen portakallar ve orjinal ilk renkli görüntü Şekil 6.19 ile verilmiş olup bulunan nesnelerin özelliklerinin alınması aşamasına geçilmiştir.



Şekil 6.19. a. Maskelenen portakallar, b. İlk görüntü

Sonuç olarak güneşli ortamda alınan görüntüler üzerinde bazı yaprak alanlarının renk pikselleri portakal alanı pikselleri ile benzerlik göstermiş ve hesaplama içerisinde dahil edilmiştir. Bu durum portakal bölgelerini belirlemek için kullanılan renk tabanlı yöntemin hatalı sonuçlar üretmesine neden olmuştur. Ayrıca, görüntüler doğal aydınlatma koşullarında ve çok kaliteli olmayan bir akıllı cep telefonu ile alındığı için görüntüler üzerinde parlama dışında çok fazla gürültü de meydana gelmiştir. Bu gürültülerin büyük bir çoğunluğu uygulanan algoritmalar ile temizlendiğinde küçük pikseller olarak görünen (örneğin yapraktan dolayı çok az bir kısmı görünen portakallar) portakalların bir kısmı da temizlenmiştir.

Uygulamada karşılaşılan bir diğer zorluk ise ağaç dalında meyvelerin üst üste gelerek birbirlerini kapatmalarıdır. Bu durum sonucunda aynı dal üzerinde yan yana olan birden fazla portakal tek bir portakal olarak değerlendirilmiş olup sonucu etkilemektedir. Benzer durum yaprakların kapattığı portakallar için de geçerlidir. Yaprakların bazıları portakal meyvelerini kapattıkları için kamera tarafından görüntülenememiştir. Ayrıca, çekilen portakal ağaçları ile kamera arasında sabit bir mesafe olmaması sebebi ile çalışmalar her ağaç için benzer sonuçlar üretmemiştir.

Tüm bu sonuçlar çerçevesinde Matlab ile geliştirilen uygulamada portakalların tamamı belirlenememiştir. İstenmeyen yanlış çıktılar meydana gelmiştir.

6.2. Mobil Görüntü İşleme

Mobil görüntü işleme, taşınabilir cihazlar üzerindeki kameralar ile alınan görüntülerin bir dizi işlemlerden geçirilmesi olarak tanımlanabilir. Nesne izleme, görüntü iyileştirme, nesne sınıflandırma ve yorumlama, çok kullanılan yüz tanıma sistemi, barkod okuma gibi kullanım alanları oldukça geniştir. Mobil görüntü işleme uygulamalarından daha karmaşık ve gelişmiş olanları ise gerçek zamanlı olarak yürütülen uygulamalardır. Kameradan alınan görüntünün gerçek zamanlı olarak işlenmesi, daha fazla gürültüye, ışık kontrolüne ve işlemci yoğunluğu gibi olumsuz durumlara sebep olduğundan dolayı ileri algoritmalar ile işlemlerin gerçekleştirilmesi gerekmektedir.

Tez kapsamında yürütülen çalışmada günümüzde kullanım alanı oranında büyük bir paya sahip olan Android işletim sistemi tercih edilmiştir. Android kullanımı sadece akıllı telefonlar ve tabletlerle sınırlı olmamakla birlikte, akıllı kitaplar, televizyonlar, kameralar, saatler, kulaklıklar ve daha pek çok teknolojik alet Android işletim sistemi kullanmaktadır. Yaşamımıza bu denli entegre olmuş bir sistemi hayatın her alanında

kullanabilirliği düşüncesinden yola çıkarak, bilgisayar ortamında geliştirilen uygulamamızı Android platformuna taşıyıp tarım sektöründe etkin bir şekilde kullanılması amaçlanmıştır. Bu doğrultuda uygulama yeniden düzenlenerek Android Studio programlama aracı ve OpenCV görüntü işleme kütüphanesi yardımıyla geliştirilmiştir.

Tez kapsamında gerçekleştirilen uygulamada öncelikle yeni bir proje oluşturularak projeye OpenCV kütüphanesi eklenmiştir. Proje oluşturulduğunda aktivite metodu çağırılmıştır ve *onCreate* metodu devreye girmiştir. Bu metotta genel olarak *setContentView* metodu hazır olarak gelir ve çalıştırılarak *layout* dosyasından ekran tasarımı yüklenmektedir. Çalışmada, ekran ilk oluştuğunda *onCreate* metodu içerisinde tanımlanması gereken değerler tanımlanarak bu metot içerisine yazılan kodlar aşağıda verilmiştir.

```
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
sc1=new Scalar(iLowH,iLowS,iLowV);
sc2=new Scalar(iHighH,iHighS,iHighV);
cameraView=(JavaCameraView)findViewById(R.id.cameraview);
cameraView.setCameraIndex(0); // 0 arka, 1 ön kamera için.
cameraView.setCvCameraViewListener(this);
cameraView.enableView();
button=(Button)findViewById(R.id.button);
tv1=(TextView)findViewById(R.id.textView2);}
```

Uygulamanın ekran tasarımında *RelativeLayout* tercih edilmiştir. Bir adet Button, iki adet TextView ve bir adet JavaCameraView görsel öğesi kullanılmıştır. Uygulamanın basit ekran tasarımı Şekil 6.20 ile gösterilmektedir.

Şekil 6.20.Uygulamanın ekran tasarımı

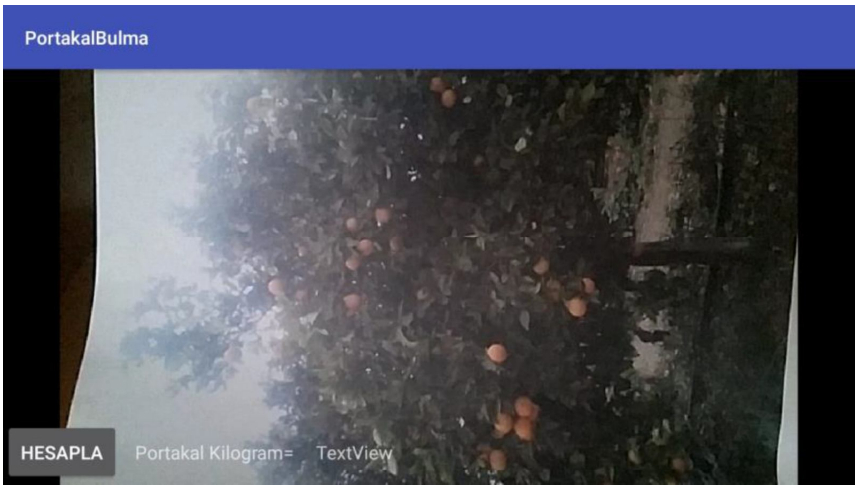
Ekranda gösterilen "HESAPLA" isimli Button, *onCreate* metodu içerisinde tanımlanmıştır ve *findViewById*() metoduna parametre olarak id'si verilerek buton kaynağına erişim sağlanmıştır. Aynı şekilde TextView, JavaCameraView görsel ögesi de *onCreate* içerisinde tanımlanarak *findViewById()* metoduna parametre olarak id'leri verilip erişimleri gerçekleştirilmiştir. Butona tıklanması ile ekran üzerinde algılanan portakal meyvelerinin kilogram hesabı TextView içerisinde gösterilmektedir. "Portakal Kilogram=" ile ifade edilen TextView sabit olup, *cameraView* komutu ile isminden de anlaşılacağı üzere kamera ayarları ile ilgili düzenleme gerektiren kodlar eklenmiştir. Arka kameranın açılması sağlanıp, kameranın çalışması aktif hale getirilmiştir.

Android ortamında geliştirilen çalışmada, Matlab uygulamasında geliştirdiğimiz algoritmaya benzer şekilde bir algoritma oluşturulmuştur. Matlab uygulamasından farklı olarak tek bir görüntü üzerinde çalışılmamış, gerçek zamanlı olarak görüntü işleme adımları gerçekleştirilmiştir.

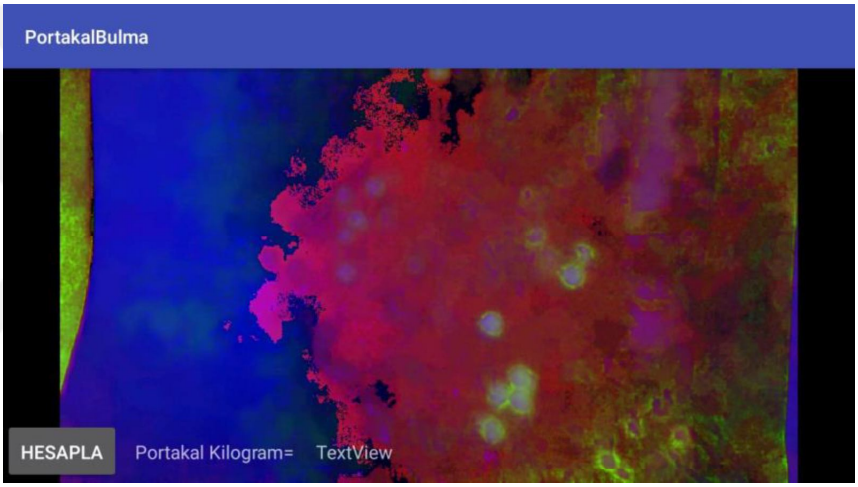
Çalışmada öncelikle kameradan renkli görüntülerin alınması sağlanmıştır. Alınan renkli görüntüler Şekil 6.21 ile verilmiştir. Ardından kameradan alınan RGB renkli görüntüler bir filtre işleminden geçirilmektedir. Bu işlem ile görüntüde istemediğimiz gürültüler yok edilmektedir (Şekil 6.22). Sonrasında renkli RGB görüntü HSV renk uzayına çevrilerek histgram hesaplması yapılıp eşikleme işlemi için uygun değerlerin

görülmesi sağlanmıştır (Şekil 6.23). HSV renk uzayında her bir renk uzayının ayrı ayrı histogram değerleri elde edildikten sonra eşik değeri manuel olarak seçilmiştir. Deneme yanılma yöntemi ile en uygun eşik değeri belirlendikten sonra *onCreate* metodu içerisinde "sc1,sc2" olarak en üst sınır ve en alt sınır eşik değerleri tanımlanmıştır. Tanımlanan bu eşik değerlerine göre eşikleme işlemi gerçekleştirilerek ikili görüntü oluşturulmuştur (Şekil 6.24). Oluşan ikili görüntüde portakallar beyaz olarak, arka plan siyah olarak görünmektedir. Filtre ile arka plandan tam olarak temizlenemeyen ve nesneler üzerinde kalan boşlukları doldurmak için morfolojik işlemler uygulanmıştır. İşlem sonucunda portakallar daha temiz ve görünür hale gelmiştir. Daha sonra arka plandan ayrılan portakallarımızın etrafı bir dikdörtgen ile çizilerek (Şekil 6.25), alan bilgileri, sayıları, yükseklikleri ve genişlikleri gibi parametreleri alınarak tahmini bir kilogram hesabı yapılmış olup, ekrandaki "HESAPLA" butonuna basılarak bulunan değer TextView'e basılmıştır. Uygulamanın çıktı ekran görüntüsü Şekil 6.26 ile verilmiştir.

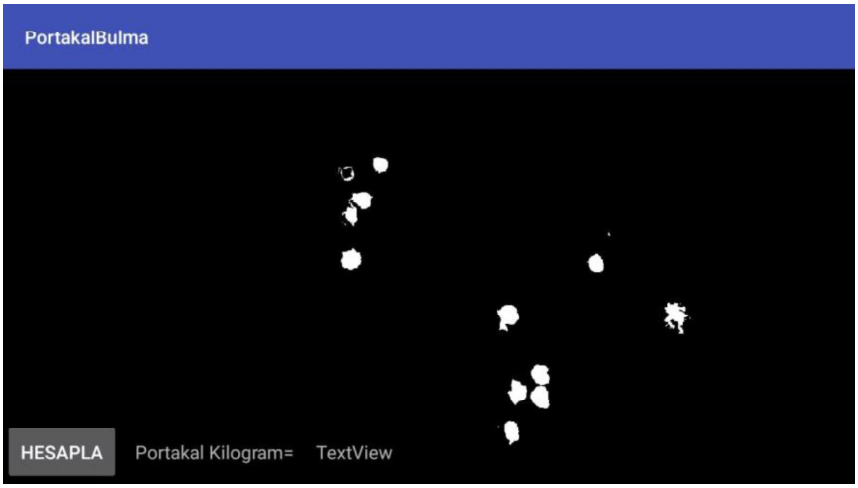
Şekil 6.21. Telefon kamerası ile alınan renkli gerçek zamanlı görüntüler



Şekil 6.22. Görüntülerin filtre işleminden geçirilmiş hali



Şekil 6.23. Görüntülerin RGB'den HSV uzayına geçirilmesi sonucu elde edilen görüntüler



Şekil 6.24. Eşiklenen görüntüler

Şekil 6.25. Portakal belirlenmiş görüntüler

Şekil 6.26. Portakal kilogramı hesaplanmış görüntüler

Android Studio ile gerçekleştirilen çalışmada, doğal bahçe koşullarında çekilen görüntülerin fotokopi çıktıları kullanılmıştır. Portakal meyvesinin yetişme şartları yaşadığımız bölgenin iklim şartları uymadığı için yetişmediğinden dolayı gerçek zamanlı olarak bahçe ortamında portakal ağaçlarını gözlemleme şansımız olmamıştır. Gerçekleştirilen uygulama ile gerçek zamanlı olarak kameradan görüntü alımı başlatılıp portakal ağaç çıktı resimlerinin üzerine getirilmiştir. Gerçek zamanlı olarak çalışılması, çok fazla gürültüye sebep olduğu için görüntü üzerindeki gürültülerin tamamı temizlenememiştir. Gürültülerin tamamı temizlenmek istendiğinde görüntüde bozulmalar meydana geldiği için farklı bir yöntem ile bu olumsuz durumun önüne geçilmeye çalışılmıştır. Arka plandan ayrılan portakalların en küçük alana sahip olanı belirlendikten sonra bu değer altında kalan nesneleri temizlemek üzere bir algoritma gelitşirilmiştir. Bu sayede doğruluk payı arttırılmıştır.

Tüm bu olumsuz durumlara rağmen önerilen iki yöntemde de istenmeyen durumlar minimum seviyelere çekildiği için gölgesiz ve parlak görüntüde sırasıyla %89.8 ve %76.95 oranında başarı sağlanmıştır. Android ortamında, Matlab ile çalışılan gölgesiz ve nispeten karanlık olan görüntü üzerinde %71.88 doğruluk payı ile meyve yükü hesaplanmıştır. Güneşli ve parlak olan görüntü üzerinde ise %65.04 oranında başarı elde edilmiştir.

7. SONUÇLAR VE ÖNERİLER

7.1 Sonuçlar

Çoğu uygulamada yaygın şekilde kullanılan görüntü işleme tekniklerinin tarım alanında da kullanımı gün geçtikçe artmaktadır. Yapılan literatür çalışmalarının son on yılda artışı, çıktı sonuçlarının doğruluğu ve güvenirliliğiyle beraber görüntü işleme tekniklerinin tarım sektöründe çok tercih edildiğini göstermektedir. Gerçekleştirilen uygulamalar yabancı otlarla müdahale, hasat işlemlerinin robotikleşmesi, meyve kusurlarının belirlenmesi, meyve boyutlarına göre sınıflandırma yapılması, olgunlaşma süreleri vb. konular olup, ağaç meyve yükü hesaplamasından yola çıkarak verim tahmini uygulaması literatürde daha önce örneği olmayan bir çalışmadır.

Önerilen çalışmada doğal bahçe ortamının olumsuz etkileri (güneş ışığı, havanın bulutlu olması gibi), meyve algılama sürecinde yaşanan negatif durumlar (yaprakların meyveleri kapatması, birbirine yakın olan meyvelerin tek meyve olarak algılanması, kamera mesafesi, kamera kalitesi vb.) portakalların tam olarak belirlenememesine neden olmuştur. Gerçekleştirilen çalışma ile tüm bu olumsuz şartlar minimum seviyelere indirilerek algoritmaların geliştirilmesi hedeflenmiştir. Özellikle çalışmanın Android işletim sistemleri ile uyumlu hale getirilmesi probleme bambaşka bir bakış açısı getirmiş olup, elde edilen veriler uygulamanın geleceği olduğunu göstermektedir. Görüntü işleme uygulamalarında ışık şiddeti koşullarının etkisini ve sınıflandırma problemlerinin azaltılması ile ilgili yaklaşımlar devam etmektedir.

7.2 Öneriler

Görüntü işleme teknikleri kullanılarak ağaç meyve yükü hesaplaması uygulamasında kullanılan teknikler ile portakallar tam olarak belirlenememekle birlikte iyi sonuçlar elde edilmiştir. Uygun ışık altında, kaliteli bir lens ile alınan görüntüler üzerinde çalışılması, çıktı hesaplamalarının doğruluk payını arttıracaktır. Özellikle çok sık yaprak alanına sahip olmayan bir ağaç daha doğru sonuç verecektir. Ayrıca portakalların kilo hesabında kullanılan YSA dışında diğer yöntemler ile hesaplamalar gerçekleştirilebilir ve sonuçlar tartışılabilir. Android Studio ile gerçekleştirilen uygulama görüntüler üzerinde iyileştirmeler arttırılarak, aydınlatma olumsuz etkeni azaltma durumu

gerçekleşirse ve doğal ortam şartlarında uygulamayı test etme imkanı olursa daha doğru sonuçlar elde edilebilir.

KAYNAKLAR

Acar, U., 2011, Uydu görüntüleri ve tıbbi görüntülerden benzer görüntü işleme teknikleriyle bilgi çıkarımı, *YTÜ Fen Bilimleri Enstitüsü*.

Akkaya, G., 2007, Yapay sinir ağları ve tarım alanındaki uygulamaları, *Atatürk Üniversitesi Ziraat Fakültesi Dergisi*, 38 (2), 195-202.

Angel, E. ve Shreiner, D., 2011, Teaching a shader-based introduction to computer graphics, *IEEE Comput Graph Appl*, 31 (2), 9-13.

Atalı, G., Özkan, S. S. ve Karayel, D., Morfolojik Görüntü İşleme Tekniği ile Yapay Sinir Ağlarında Görüntü Tahribat Analizi, *Akademik Platform Mühendislik ve Fen Bilimleri Dergisi*, 4 (1).

Ataseven, B., 2013, Yapay sinir ağları ile öngörü modellemesi.

Aygören, H., Saritaş, H. ve Morali, T., 2012, İMKB 100 Endeksinin Yapay Sinir Ağları ve Newton Nümerik Arama Modelleri ile Tahmini, *Journal of Alanya Faculty of Business/Alanya Isletme Fakültesi Dergisi*, 4 (1).

Aytan, A. E., Öztürk, Y. ve Örgev, E. K., 1993, GÖRÜNTÜ İŞLEME-DIGITAL IMAGE PROCESSING, *Journal of Istanbul University Faculty of Dentistry*, 27 (4), 273-277.

Bayır, F., 2006, Yapay sinir ağları ve tahmin modellemesi üzerine bir uygulama, *Yüksek Lisans Tezi, İstanbul Üniversitesi Sosyal Bilimler Enstitüsü İşletme Ana Bilim Dalı Sayısal Yöntemler Bilim Dalı, İstanbul*.

Bg, F. ve Wa, C., 1954, Simulation of self-organizing systems by digital computer, p.

Carpenter, G. A. ve Grossberg, S., 1988, The ART of adaptive pattern recognition by a self-organizing neural network, *Computer*, 21 (3), 77-88.

Chang, C.-H., Hsieh, C.-W., Jong, T.-L. ve Tiu, C.-M., 2003, A fully automatic computerized bone age assessment procedure based on phalange ossification analysis, *Proc IPPR*, 16, 463-468.

Cheng, H.-D., Shan, J., Ju, W., Guo, Y. ve Zhang, L., 2010, Automated breast cancer detection and classification using ultrasound images: A survey, *Pattern recognition*, 43 (1), 299-317.

Civalek, Ö., 2004, Dikdörtgen plakların doğrusal olmayan analizinde yapay sinir ağı yaklaşımı, *Teknik Dergi*, 15 (72).

Cooley, J. W. ve Tukey, J. W., 1965, An Algorithm for the Machine Calculation of Complex Fourier Series, *Mathematics of Computation*, 19 (90), 297-301.

Crane, R., 1997, A simplified approach to image processing : classical and modern techniques in C, *Upper Saddle River, N.J.*, Prentice Hall PTR, p.

Curram, S. P. ve Mingers, J., 1994, Neural networks, decision tree induction and discriminant analysis: An empirical comparison, *Journal of the Operational Research Society*, 45 (4), 440-450.

Çavuşlu, M. A., Becerikli, Y. ve Karakuzu, C., Levenberg-Marquardt algoritması ile YSA eğitiminin donanımsal gerçeklenmesi, *Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi*, 5 (1).

Çelik, E., 2014, Görüntü İşlemeye Dayalı Avuç İçi İzinin Yapay Sinir Ağı ile Tanınması.

Demir, Y., 1997, Yapay Sinir Ağları ile Ulaştırma Taleplerinin Modellenmesi, *İTÜ Fen Bilimleri Enstitüsü, Yüksek Lisans tezi, İstanbul*.

Demirel, S. O., 2008, Görüntü işleme teknikleri ve medikal uygulamaları, *Ege Üniversitesi*.

Elmas, Ç., 2003, Yapay sinir ağları:(kuram, mimari, eğitim, uygulama), Seçkin Yayıncılık, p.

Fırat, M. ve Güngör, M., 2004, Askı madde konsantrasyonu ve miktarının yapay sinir ağları ile belirlenmesi, *Teknik Dergi*, 15 (73).

Fukushima, K., 1975, Cognitron: A self-organizing multilayered neural network, *Biological cybernetics*, 20 (3-4), 121-136.

Gonzalez, R. C. ve Woods, R. E., 2002, Digital image processing, *Upper Saddle River, N.J.*, Prentice Hall, p.

Gonzalez, R. C. ve Woods, R. E., 2008, Digital image processing, *Upper Saddle River, N.J.*, Prentice Hall, p.

Gül, Ç., 2016, Renk uzayları [online], http://www.caglargul.com/2016/11/renk-uzaylari.html [Ziyaret Tarihi: 13 Kasım 2018].

Güraksın, G. E., 2015, Yapay zeka teknikleri kullanarak kemik yaşı tespiti, *Selçuk Üniversitesi Fen Bilimleri Enstitüsü*.

Hannan, M., Burks, T. ve Bulanon, D. M., 2010, A machine vision algorithm combining adaptive segmentation and shape analysis for orange fruit detection, *Agricultural Engineering International: CIGR Journal*.

Hannan, M. W. ve Burks, T. F., 2004, Current developments in automated citrus harvesting, *2004 ASAE Annual Meeting*, 1.

Haralick, R. M. ve Shapiro, L. G., 1992, Computer and robot vision, Addison-wesley, p. 346-351.

Hwang, J. G. ve Ding, A. A., 1997, Prediction intervals for artificial neural networks, *Journal of the American Statistical Association*, 92 (438), 748-757.

Işıngör, A., Android Stduio'yu tanıyalım [online], https://gelecegiyazanlar.turkcell.com.tr/konu/android/egitim/android-201/android-studioyu-taniyalim, 02.12.2018.

Jähne, B., 1997, Digital image processing : concepts, algorithms, and scientific applications, *Berlin ; New York*, Springer, p.

Jähne, B., 2005, Digital image processing, *Berlin ; New York*, Springer, p.

Jang, J., Sun, C. ve Mizutani, E., 1997, Neuro-Fuzzy and Soft Computing, Prentice-Hall, upper Sanddle River.

Kahya, E. ve Arın, S., 2014, Görüntü Renk Kod Analizi İle Meyvenin Yerinin Tespiti Üzerine Bir Araştırma A Research On Image Color Code Analysis With Fruit Locating, *JOTAF/Tekirdağ Ziraat Fakültesi Dergisi*, 11 (2), 110-118.

Kanimozhi, B. ve Malliga, R., 2017, Classification of Ripe or Unripe Orange Fruits Using the Color Coding Technique, *vol*, 1, 43-47.

Karakuş, D., 2006, Görüntü analiz yöntemleri ile kayaçların yapısal özelliklerinin tanımlanması, *DEÜ Fen Bilimleri Enstitüsü*.

Kaynak, O. ve Efe, M., 2000, Yapay Sinir Ağları ve Uygulamaları, *BOĞAZİÇİ ÜNİVERSİTESİ*.

Klopf, A. H., 1972, Brain function and adaptive systems: a heterostatic theory, *AIR FORCE CAMBRIDGE RESEARCH LABS HANSCOM AFB MA*.

Kohonen, T., 1988, An introduction to neural computing, *Neural Networks*, 1 (1), 3-16.

Kondo, N., Ahmad, U., Monta, M. ve Murase, H., 2000, Machine vision based quality evaluation of Iyokan orange fruit using neural networks, *Computers and Electronics in Agriculture*, 29 (1), 135-147.

Köybaşı, E., 2013, Dijital kamera kullanılarak robotun bilgisayar sistemiyle birleşik çalıştırılması.

Kuncan, M., Ertunç, H. M., Küçükyıldız, G., Hızarcı, B., Ocak, H. ve Öztürk, S., 2013, Görüntü işleme tabanlı zeytin ayıklama makinesi, *Otomatik Kontrol Ulusal Toplantısı*, 459-464.

KURTULMUŞ, F., VARDAR, A. ve Kavdır, İ., 2013, Bahçe Koşullarında Alınmış Renkli Görüntülerde Doku ve Şekil Öznitelikleriyle Genç Şeftali Meyvelerinin Saptanması, *Tarım Makinaları Bilimi Dergisi*, 9 (2).
McCulloch, W. S. ve Pitts, W., 1943, A logical calculus of the ideas immanent in nervous activity, *The bulletin of mathematical biophysics*, 5 (4), 115-133.
Metlek, S., Uzunkavak, M. ve Dalı, S. D. Ü. F. B. E. E. B. E. A., 2009, Üretim bandı üzerindeki renkli silindir parçalarının makine görme sistemiyle tanımlanması ve sınıflandırılması, SDÜ Fen Bilimleri Enstitüsü, p.
Minsky, M. ve Papert, S., 1969, Perceptron Expanded Edition, MIT Press, Cambridge, MA. Original edition published in.
Morrissey, B. M., 1988, (HSI) Color Processing On Personal Computers, *Applications of Digital Image Processing XI*, 173-177.
Niblack, W., 1986, An introduction to digital image processing, *Englewood Cliffs, N.J.*, Prentice-Hall International, p.
Nygren, K., 2004, Stock prediction–a neural network approach, *Royal Instiute of Technology*, 1-34.
Otsu, N., 1979, A threshold selection method from gray-level histograms, *IEEE transactions on systems, man, and cybernetics*, 9 (1), 62-66.
Özcan T., 2018, Matlab Nedir? [online], https://www.heryerdeyazilim.com/matlab-nedir/ [Ziyaret Tarihi: 27 Kasım 2018].
Özel, H., 2018, Matlab Nedir? [online], https://medium.com/@halilozel1903/matlab-nedir-91a904a74f45 [Ziyaret Tarihi: 27 Kasım 2018].
Öztemel, E., 2003, Yapay Sinir Ağlari, *PapatyaYayincilik, Istanbul*.
Özveren, U., 2006, Pem yakıt hücrelerinin yapay sinir ağları ile modellenmesi.
Payne, A. B., Walsh, K. B., Subedi, P. ve Jarvis, D., 2013, Estimation of mango crop yield using image analysis–segmentation method, *Computers and Electronics in Agriculture*, 91, 57-64.
Pratt, W. K., 2001, Digital image processing : PIKS inside, *New York*, Wiley, p.
Ranganathan, A., 2004, The levenberg-marquardt algorithm, *Tutoral on LM algorithm*, 11 (1), 101-110.
Rosenblatt, F., 1958, The perceptron: A probabilistic model for information storage and organization in the brain [J], p.
Rosenfeld, A., 1969, Picture processing by computer, *ACM Computing Surveys (CSUR)*, 1 (3), 147-176.
Sağıroğlu, Ş., Beşdok, E. ve Erler, M., 2003, Mühendislikte yapay zeka uygulamaları-1: Yapay sinir ağları, Ufuk Kitap Kırtasiye-Yayıncılık, p.
Samtaş, G. ve Gülesin, M., 2012, Sayısal Görüntü İşleme ve Farklı Alanlardaki Uygulamaları, *Ejovoc (Electronic Journal of Vocational Colleges)*, 2 (1).
Saraç, T., 2004, Yapay Sinir Ağlan Seminer Projesi, *Gazi Üniversitesi Endüstri Mühendisliği Bölümü, Ankara*.
Senthilkumaran, N. ve Vaithegi, S., 2016, Image segmentation by using thresholding techniques for medical images, *Computer Science & Engineering: An International Journal*, 6 (1), 1-13.
Slaughter, G. E. ve Hobson, R. S., 2006, Artificial Neural Network for Temporal Impedance Recognition of Neurotoxins, *IJCNN*, 2001-2008.
Şen, Z., 2004, Yapay sinir ağları ilkeleri, Su Vakfı, p.
Taşova, O., 2011, Yapay sinir ağları ile yüz tanıma, *DEÜ Fen Bilimleri Enstitüsü*.
Tatlı, M. ve Üncü, İ. S., 2014, Mobil cihazlarda görüntü işleme için bir çözüm önerisi, *Akademik Bilişim Konferansı*, 05-07.

Thomas, W. ve Connolly, C., 1986, Applications of colour processing in optical inspection, *Automatic Optical Inspection*, 116-123.
Tonguç, G., 2007, Görüntü işleme teknikleri kullanılarak meyve tasnifi, *Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü*.
Türkiye istatistik kurumu, 2018, meyveler, içecek ve baharat bitkilerin üretim miktarları (seçilmiş ürünlerde), *TÜİK*, http://www.tuik.gov.tr/UstMenu.do?metod=temelist
Vashisth, R. ve Chandra, A., 2010, Predicting stock returns in Nifty index: An application of artificial neural network, *International Research Journal of Finance and Economics*, 49, 15-24.
Werbos, P., 1974, Beyond Regression:" New Tools for Prediction and Analysis in the Behavioral Sciences, *Ph. D. dissertation, Harvard University*.
Yaman, K., 2000, Görüntü işleme yönteminin Ankara hızlı raylı ulaşım sistemi güzergahında sefer aralıklarının optimizasyonuna yönelik olarak incelenmesi. Yayınlanmamış Yüksek Lisans Tezi, Gazi Üniversitesi, *Fen Bilimleri Enstitüsü*.
Yaşar, G. H. ve Akdemir, B., 2017, Estimating Yield for Fruit Trees Using Image Processing and Artificial Neural Network, *Int'l Journal of Advances in Agricultural & Environmental Engg. (IJAAEE)*, 4 (1).
Yavuz, S. ve Deveci, M., 2012, İstatiksel Normalizasyon Tekniklerinin Yapay Sinir Ağin Performansina Etkisi, *Erciyes Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi* (40), 167-187.
Young, I. T., Gerbrands, J. J. ve Van Vliet, L. J., 1998, Fundamentals of image processing, Delft University of Technology Delft, p.
Zhao, Y., Wang, D. ve Qian, D., 2009, Machine Vision Based Image Analysis for the Estimation of Pear External Quality, *2009 Second International Conference on Intelligent Computation Technology and Automation*, 629-632.
https://tr.wikipedia.org/wiki/Resim:RGB_farbwuerfel.jpg [Ziyaret Tarihi: 11 Mart 2017].
https://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html [Ziyaret Tarihi: 18 Nisan 2017]
http://gdt.itu.edu.tr/wp-content/uploads/Teknik_Raporlar/Teknik_Rapor_04/Teknik_Rapor_04_html/mobile/index.html#p=29 [Ziyaret Tarihi: 11 Mart 2017]
Uğurlu B., 2011, http://members.comu.edu.tr/boraugurlu/
http://documentation.statsoft.com/STATISTICAHelp.aspx?path=SANN/Overview/SANNOverviewsNetworkTypes [Ziyaret Tarihi: 18 Mart 2017]
https://uk.mathworks.com/discovery/what-is-matlab.html [Ziyaret Tarihi: 27 Kasım 2018].
https://gelecegiyazanlar.turkcell.com.tr/konu/android/egitim/android-201/android-ndk [Ziyarte Tarihi: 20 Ekim 2018]
https://gelecegiyazanlar.turkcell.com.tr/konu/android/egitim/android-201/layout [Ziyaret Tarihi: 20 Ekim 2018]
Yazar M., 2013i http://kod5.org/activity-sinifi/ [online] [Ziyaret Tarihi: 20 Ekim 2018]
https://docs.opencv.org/java/2.4.9/org/opencv/android/JavaCameraView.html [Ziyaret Tarihi: 1 Aralık 2018]
https://opencv.org/about.html [Ziyaret Tarihi: 1 Aralık 2018]
https://docs.gimp.org/en/plug-in-dilate.html, [Ziyaret Tarihi :30 Kasım 2018].
Beyler A., 2017, https://www.mobilhanem.com/android-dersleri-gradle/ [Ziyaret Tarihi: 30 Kasım 2018].

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı	**:**	Gamze Hikmet YAŞAR
Uyruğu	**:**	Türkiye Cumhuriyeti (T.C.)
Doğum Yeri ve Tarihi	**:**	Konya- 05.04.1992
Telefon	**:**	05307628916
Faks	**:**	-
e-mail	**:**	gamzeozteris@gmail.com

EĞİTİM

Derece		**Adı, İlçe, İl**	**Bitirme Yılı**
Lise	:	Konya Anadolu Lisesi	2010
Üniversite	:	Selçuk Üniversitesi Elk. Elt. Müh. Bölümü	2014
Yüksek Lisans	:	Selçuk Üniversitesi Elk. Elt. Müh. Bölümü	Devam ediyor
Doktora	:	**-**	

İŞ DENEYİMLERİ

Yıl	**Kurum**	**Görevi**
2014-2018	Elit Mühendislik İnş. Ve Elk.Ltd.Şti.	Koordinatör Yrd.

YABANCI DİLLER

İngilizce, Almanca

YAYINLAR

Yaşar, G. H. ve Akdemir, B., 2017, Estimating Yield for Fruit Trees Using Image Processing and Artificial Neural Network, *Int'l Journal of Advances in Agricultural & Environmental Engg. (IJAAEE)*, 4 (1).

Akdemir B. Ve Yaşar G. H., 2017, The Estimation of Yield in Orange Trees Using Image Processing, 2. International Conference on Advances in Natural and Applied Sciences. (ICANAS).