T.C.
İNÖNÜ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

UZAKTAN ALGILAMA VE GÖRÜNTÜ İŞLEME TARIMA UYGULANMASI

YÜKSEK LİSANS TEZİ

Ahmet Yaşar BALKESEN

Fizik Anabilim Dalı

Tez Danışmanı: Dr. Öğr. Üyesi Tuncay ÖZDEMİR

OCAK - 2023

T.C
İNÖNÜ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

UZAKTAN ALGILAMA VE GÖRÜNTÜ İŞLEME TARIMA UYGULANMASI

YÜKSEK LİSANS TEZİ

Ahmet Yaşar BALKESEN
(36203612003)

Fizik Anabilim Dalı

Tez Danışmanı: Dr. Öğr. Üyesi Tuncay ÖZDEMİR

OCAK - 2023

TEŞEKKÜR VE ÖNSÖZ

Bu tez çalışmasında uzaktan algılama ve görüntü işleme yöntemlerinin tarım üzerine uygulanması ile rekolte hesaplanması sağlanmıştır. Bu çalışmada ilgi, yardım ve desteklerinden dolayı danışman hocam Sayın Dr. Öğr. Üyesi Tuncay ÖZDEMİR'e

Ayrıca hayatımda en büyük destekçim olan karım ve çocuklarıma,

teşekkür ederim.

ONUR SÖZÜ

"Uzaktan Algılama ve Görüntü İşleme Tarıma Uygulanması" isimli yüksek lisans çalışmasında tez içerisinde sunduğum verileri, bilgileri etik kurallar içerisinde bilimsel ahlak kurallarına uygun bir şekilde sunduğumu, yararlandığım tüm eserlere atıfta bulunarak kaynakçada uygun bir biçimde gösterdiğimi ve bu tezde sunduğum çalışmanın özgün olduğunu belirtir ve bunu doğrularım.

Ahmet Yaşar BALKESEN

İÇİNDEKİLER

ÇİZELGELER DİZİNİ

ŞEKİLLER DİZİNİ

SEMBOLLER VE KISALTMALAR

UA **:** Uzaktan Algılama

YSA **:** Yapay Sinir Ağı

BG **:** Bilgisayarlı Görü

YOLO **:** You Only Looked Once

OPENCV **:** Open Source Computer Vision Library

ÖZET

Yüksek Lisans Tezi

Uzaktan Algılama ve Görüntü İşleme Tarıma Uygulanması

AHMET YAŞAR BALKESEN

İnönü Üniversitesi
Fen Bilimleri Enstitüsü
Fizik Anabilim Dalı

2023

Danışman: Dr. Öğr. Üyesi Tuncay ÖZDEMİR

Tarım, geçmişten günümüze insanoğlunun en önemli ihtiyaç ve geçim kaynaklarının başında gelmektedir. Tarım daima bir teknolojik değişim ve gelişim içerisinde olmuştur. Günümüzde de teknolojinin hızlıca gelişmesinden tarım sektörüde çok fazla etkilenmektedir. Artık tarım alanında derin öğrenme yöntemleri gibi bilgisayarlı görü teknikleri yaygın olarak kullanılmaya başlanmaktadır.

Elma bahçelerinde ağaçlardaki meyveler olgunlaşıp satışa hazır hale geldiklerinde tarladaki toplam rekolte bu konuda deneyimli kişiler tarafından tahmin edilir ve satışlarda ortalama bu tahminler üzerinden yapılır. Bu araştırma ile elma bahçelerinden alınan görüntüler bilgisayar ortamına aktarılıp derin öğrenme teknikleri ile elde edilen sayısal veriler kullanılarak yapay zeka yardımıyla meyve rekoltesi tahmini yapılması sağlanmıştır. Bu sayede deneyimli kişilere ihtiyaç kalmadan yapay zeka yardımıyla elde edilen verilerle sağlıklı bir rekolte tahmini yapılması sağlanarak daha güvenilir bir ticaret yapılması amaçlanmaktadır.

Tez araştırmamızda, Şanlıurfa Küçük Akziyaret köyünde bulunan elma ağaçlarının görüntüleri alınarak bilgisayar ortamına aktarılmıştır. Elma rekoltesi tahmini için gerekli veriler Phyton proglama dili kullanılarak Anaconda programı ve Anaconda programının alt programı olan Spyder kullanılarak kodlanmıştır. Kodlamada OpenCV, Numpy kütüphanelerinden faydalanılmıştır. YOLOv3 yapay sinir ağı kullanılarak nesne tanıma yapılacaktır ve bu bilgiler doğrultusunda ağaçtaki tahmini elma miktarı hesaplanıp rekolte hesaplanacaktır.

Anahtar Kelimeler: Yapay Zeka, Görüntü İşleme, YOLO, Bilgisayarlı Görü.

ABSTRACT

Master Thesis

Remote Sensing and Image Processing Application to Agriculture

AHMET YAŞAR BALKESEN

Inonu University
Institute Of Science
Department of Physics

2023

Supervisor: Dr. Öğr. Üyesi Tuncay ÖZDEMİR

Agriculture is one of the most important needs and livelihoods of human beings from past to present.Agriculture has always been in a technological change and development.Today, the agricultural sector is greatly affected by the rapid development of technology. Computer screening techniques such as deep learning methods are now widely used in agriculture.

In apple orchards, when the fruits on the trees are ripe and ready for sale, the total yield in the field is estimated by experienced people and sales are averaged over these estimates. With this research, the images taken from the apple orchards were transferred to the computer environment and fruit yield was estimated with the help of artificial intelligence by using the numerical data obtained from deep learning techniques. With this way, it is aimed to make a more reliable trade by providing a healthy yield estimation with the help of artificial intelligence instead of the need for experienced people.

In our thesis research, images of apple trees in ŞanlıurfaKüçükAkziyaret village were taken and transferred to the computer environment.The data required for apple yield estimation were coded using the Phyton programming language and using the Anaconda program and Spyder, the subprogram of the Anaconda program.OpenCV, Numpy libraries were used for coding.Object recognition will be made using the YOLOv3 artificial neural network and with the life of this information, the estimated amount of apples in the tree so yield, will be calculated.

Keywords: Artificial Intelligence, Image Processing, YOLO, Computer Vision.

1. GİRİŞ

1.1 Araştırmanın Amacı

Tarım, canlıların temel ihtiyacı olan gıdaların temin edildiği, tarihi çok eskilere dayanan önemli bir uğraş alanıdır. Yıllarca insan ve hayvan gücü kullanılarak basit el aletleri ile yapılmakteyken günümüzde artık makineleşme yardımıyla daha kolay ve daha verimli ekimler ve hasatlar yapılabilmektedir. Günümüzde artık modern tarımın amacı birim alandan alınan ürünlerin hem kaliteli hemde daha yüksek verim alınabilmesi olmuştur.

Tarımda meyve üretimide büyük öneme sahiptir. Ülkemizde ve Dünyada meyve üretimi ithalat ve ihracat bakımından ticari pazarda büyük bir önemi vardır. Türkiye gerek iklimi gerekse toprağı bakımından birçok meyve yetiştirmeye elverişlidir. Meyvelerin ekim ve olgunlaşma süreçlerinden sonra hasat işlemi yapılmaktadır. Hasat işlemi yapıldıktan sonra yıllık üretim miktarı yani rekolte ortaya çıkmaktadır. Çiftçiler bazen rekolte hesaplamadan satış yapmakta bazende deneyimli kişiler tarafından hasat işlemi yapılmadan hemen önce ağaçlardaki meyveye bakarak kabataslak bir hesap yapmaktadır ve ticareti bu değerler üzerinden yapmaktadırlar. Yinede çiftçiler yaptıkları üretim miktarını bu şekilde tam olarak bilememektedir.

Günümüzde uzaktan algılama ve görüntüleme sistemleri birçok uygulama alanında çalışmalar gerçekleşmektedir.Görüntülerin algılanması, işlenmesi, analizi ve yorumlanması sayesinde işlerimiz biraz daha kolaylaşmaktadır.

Uzaktan algılama “Herhangi bir temas olmaksızın bir obje hakkında birşeyler söyleme sanatı ve bilimidir.” (Fischer vd.1976) Uzaktan algılama sistemlerinde yeryüzündeki herhangi bir cismin yayınladığı veya yansıttığı elektromanyetik enerji dijital bir şekilde kayıt altına alınır. Kayıt altına alınan bu veriler bilgisayar ortamında sayısal hale getirilip, analiz edilerek yeryüzünün veya nesnenin dijital bir şekilde gözlenmesi sağlanır. Uzaktan algılama sistemleri yardımıyla daha hızlı gözlemler ve daha ayrıntılı veriler elde edilerek hem zamandan hem de ekonomik yönden kazançlar sağlanabilmektedir. Cisimlerin fiziksel özellikleri cisimler ile elektromanyetik dalga

arasındaki etkileşimin dalga boyuna göre belirlenir. Elde edilen veriler farklı görüntü işleme teknikleri ile görüntü haline getirilip yorumlanarak anlamlı hale getirilir. Bu sebeple uzaktan algılama sistemleri bilgisayar teknolojisinin gelişmesi ile birlikte hızlı veri sağlama özelliğinden dolayı birçok alanda bilgi edinmek amacıyla kullanılabilmektedir.

Tez çalışmasında geliştirilen yazılım ile tarım alanında elma hasadına yönelik yeni bir yaklaşım oluşturma amaçlanmıştır. Elma ağaçlarından elde edilen dijital görüntüler bilgisayar ortamına aktarılacaktır. Geliştirilecek olan yazılım ile alınacak dijital görüntüler üzerinden elma tanıma yapılacak ve elde edilen verilere gore rekolte tahmininde bulunulacaktır. Böylece insan tahminine dayalı rekolte tahmini yerine bilgisayarlı görü ve derin öğrenme modeline dayalı bir rekolte tahmin uygulamasının yapılması amaçlanmıştır.

1.2 Araştırmanın Kapsamı

Geliştirilecek olan yazılım uzaktan algılama ve bilgisayarlı görü teknikleri ile derin öğrenme yöntemi kullanılarak yapılacaktır. Elma ağacındaki olgunlaşmış meyve yüklü bir ağaçtan elde edilen dijital görüntüler bilgisayara aktarılıp, OpenCv ve Numpy kütüphanesi yardımıyla YOLOv3 yapay sinir ağı kullanılarak görüntüler üzerinden veriler elde edilecektir. Elmalardan elde edilen veriler derin öğrenme modeli ile YOLOv3 yapay sinir ağı kullanılarak eğitilecektir. Eğitimi tamamlanan derin öğrenme modeli dijital görüntülerden elde edilen elma verileri ile rekolte tahmininde bulunacaktır. Uygulama Python programlama dili ile Anaconda ve Spyder programları kullanılarak OpenCv, Numpy, kütüphaneleri ve YOLOv3 aracılığı ile kodlanacaktır.

2. UZAKTAN ALGILAMA VE ÇALIŞMA PRENSİBİ

2.1 Uzaktan Algılamanın Tanımı

Uzaktan algılama sistemlerinin birbirine benzer bir kaç tanımlaması mevcuttur. Bunlar şu şekildedir:

- "Dünya üzerinde belirli objelerde anlamlı bilgiler üretmek üzere elde edilmiş görüntüleri yorumlayabilmek ve dünyayı belirli bir uzaklıktan izlemeyi sağlayan araç, teknik ve metotlardır." (Buiten ve Clevers, 1973)
- Yeryüzündeki coğrafi varlıkları herhangi bir fiziksel temasa geçmeksizin, algılayıcılar ile elde edilen görüntülerin analiziyle cografi varlıklar hakkında bilgi sahibi olma bilimi, tekniği ve sanatıdır." (Lillesand vd. 2004)

Tanımların ortak yanı yeryüzündeki herhangi bir nesneye fiziki temas olmaksızın teknoloji sayesinde veriler elde ederek nesne hakkında yorum yapabilmektir.

2.2 Uzaktan Algılamanın Tarihi Gelişimi

Uzaktan algılamanın tarihi gelişimi 1800'lü yıllara uzanmaktadır. 1800'lü yılların sonu 1900'lü yılların başına doğru fotoğraf makineleri kuşlara, balonlara, uçaklara takılarak yeryüzündeki nesneler hakkında veriler elde edilmeye başlanmıştır. Daha sonra uzaktan algılama sistemleri uçaklara takılan kameralar yardımıyla 1. ve 2. Dünya Savaşı sırasında askeri istihbarat amaçlı kullanılmıştır. Uzaktan algılama sistemleri bu dönemde çok sık kullanılmaya başlanmıştır. Bu tecrübelerle savaş sonrasıda askeri amaçlı kullanımlarla beraber sivil amaçlarlada bilgi edinme amaçlı kullanılmaya başlanılmıştır. 1970'lerde Amerika Birleşik Devletleri uzaktan algılama sistemlerini geliştirmek amacıyla Landsat uydusunu fırlatmış ve sonrasında bir çok devlette bu şekilde çalışmalar yapmıştır. Uydular yardımıyla artık kameralardan alınan verilerden daha ayrıntılı ve kaliteli görüntüler elde edilmeye başlanmıştır. (Campbell ve Wynne, 2011) Teknolojinin hızlı

gelişmesi ile birlikte artık günümüzde bilgisayar ve yazılımlar kullanılarak nesne tanıma sistemleri geliştirilmiştir.

2.3 Uzaktan Algılamanın Temel Bileşenleri

Görüntünün üretilmesinden işlenmesine kadar olan süreçte yapılan aşamalar uzaktan algılamanın temel bileşenleridir. Bunlar enerji, atmosfer (ortam), algılanan, algılayan (sensor), işleme (analiz) ve uygulamadır. Enerji kaynağı uzaktan algılamada en başlıca elemadır ve tespit edilmesi istenilen objelere yollanmak üzere elektromanyetik enerji sağlar. Gönderilen enerji hedef cisme çarpmadan önce bulunduğu ortamdan geçer bundan dolayı bulunduğu ortamın özellikleri cisme gelen ve yansıyan dalganın özelliğini değiştirebilmektedir. Bu sebeple ortamda uzaktan algılamanın bir bileşenidir. Diğer bir bileşen algılanandır. Yeryüzündeki bütün cisimler bu gruba dahildir. Cisimlerin fiziksel, kimyasal, boyutsal özellikleri birbirinden farklı olduğundan dolayı bu cisimlerden yansıyan enerjiler farklı olmaktadır. Bu farklılıklar sensörler tarafıdan ölçülerek cisim hakkında bilgi edinilmesini kolaylaştırır. Sensörler (algılayıcılar) elektromanyetik salınımın nesnelerle yaptığı etkileşimi ve geri yansımasını veriler halinde kaydeden cihazlardır. Kaydedilen veriler sayısal olarak işlenerek görüntü oluşturmak üzere bilgisayarlara gönderilir ve bilgisayarlı görüntü teknikleriyle işlenip analiz edilir ve sonunda yorumlanırlar. Yorumlanmalar sonucu elde edilen veriler bize cisimler hakkında bilgi sağlarlar.

2.4 Uzaktan Algılamanın Avantajları

Herhangi bir nesnenin tespit edilmesinde uzaktan algılama sistemleri önemli bir yere sahiptir. Sahip olduğu avantajlar sayesinde bir çok yöntemden daha kullanışlı hale gelmiş ve ön plana çıkmaya başlamıştır. Bu avantajlar sayısallık, hızlılık, ucuzluğu, küreselliği, güncellenebilirliği, ayrıntılı olma, güvenirliliği ve esnekliktir.(Özdemir, 2017)

- **Sayısaldır:** Görüntüler dijital veriler halinde gelir. Bu dijital veriler bir çok yazılımda işlenebilir.

- **Hızlıdır:** Uzaktan algılama ile dünyanın birçok yerindeki noktalar için görüntüsü çalışmaları yapılabilir. Bu noktalara gidilerek harcanacak zaman ile

görüntüsünün temini arasındaki geçen zaman kıyaslandığında sürenin ne kadar kısaldığı görülecektir.

- **Ucuzdur:** Tasarlanacak olan yazılım ile başka başka nesnelerin tespiti sürekli yapılabilir, yeni yeni maliyetler çıkarmaz.

- **Küreseldir:** Uydular aracılığı ile çok küçük yerlerin görüntü tespiti yapılabildiği gibi tüm dünyayı kapsayabilen görüntü çekimleride yapılabilmektedir.

- **Güncellenebilir:** Nesneye ait görüntüler gelişmelere bağlı olarak sürekli güncellenebilir. Böylece nesneye ait değişimler farklı zamanlarda elde edilen görüntülerle izlenebilir ve bu görüntüler incelenerek farklı yorumlar üretilebilir.

- **Ayrıntılıdır:** Yapılan çalışmalara bağlı olarak nesnelerin yüksek çözünürlüklü görüntüleri temin edilebilir. Bu çözünürlüklere göre nesnelerin tespit oranları daha güvenilir sonuçlar verebilir.

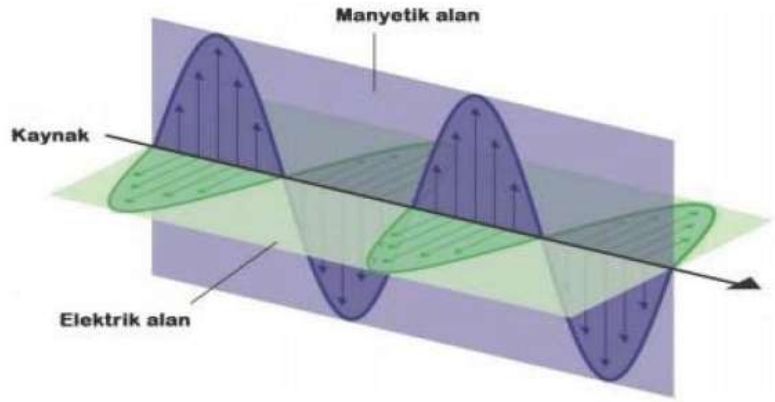
- **Güvenilirdir:** Nesne görüntüleri cisimlerden yansıyan elektromanyetik ışınların sensörler tarafından kaydedilmesi sonucu elde edildiği için nesnenin durumunu doğru bir şekilde yansıtırlar. Bu sebepten dolayı görüntülerde herhangi bir değişiklik yapılması mümkün olamaz.

- **Esnektir:** İşlenen nesne verileri kullanıldıktan sonra işlevi bitmez. Her zaman başka bir sebeple de sürekli kullanılabilir.

3. ELEKTROMANYETİK ENERJİ

3.1 Elektromanyetik Enerjinin Tanımı

Elektromanyetik enerji Şekil 3.1'deki gibi periyodik hareket yapan ve ışık hızında hareket eden bir enerjidir. Elektromanyetik enerji türleri radyo, mikro, kızıl ötesi dalgalar, görünür bölge ışınları, mor ötesi ışını, x ışınları ve gama ışınıdır.



Şekil 3.1: Elektromanyetik salınım

Elektromanyetik dalgalarda elektrik alan ile manyetik alan hem birbirlerine dik hemde yayılma doğrultularına diktir. Elektromanyetik dalgalarda dalgaların yayıldığı ortamın özellikleri önemlidir. Dalgalar katı, sıvı ve gaz gibi ortamlardan geçerken kırılmaya uğrayarak doğrultularını değiştireblir, şiddet, polarizasyon ve dalgaboylarında değişime uğrayabilirler. Uzaktan algılama uygulamalarında bu değişimler belirlenerek kaydedilirler. Kaydedilen veriler ve görüntüler nesne tespiti için yorumlanır. (Örmeci, 1987)

Elektromanyetik ışınım, geçirilebilir, yutulabilir, yansıtılabilir, yayılabilir ve saçılabilirler. Işınım tüm bu fiziksel olayların etkisindedir. Enerji ile nesne arasındaki bu durumlar uzaktan algılamanın en temel kısmını oluşturur ve ışıyan enerji şiddeti iki duruma bağlıdır. Bunlar dalgaboyunun uzunluğu ve cismin ısı miktarıdır. (Örmeci 1987).

Esas önemli olan cisimden yayılabilen enerjinin algılanabilmesidir. Uzaktan algılama sistemlerinin dört temel bileşeni vardır. (Lillesand et al. 2001, Jehnsen 1996).

Bunlar;

1. Kaynak: Güneşte gelen enerjiler, Dünyanın kendi ısısı, radar algılamada yapay elektromanyetik ışınımlar uzaktan algılamada bir kaynak türüdür.

2. Yeryüzü ile etkileşim: Yeryüzündeki nesneden yansıyan veya yayılabilen enerji miktarı ile ışınımların özelliklerine bağlıdır.

3. Atmosfer ile etkileşim: Elektromanyetik dalga atmosferde ilerlerken atmosferin özelliğine göre bozulma ve saçılmaya uğrayabilirler.

4. Algılayıcı: Elektromanyetik dalga herhangi bir obje ile bir etkileşime girebilir ve bu veriler uzaktan algılama algılayıcısı ile kayıt altına alınırlar.

4. GÖRÜNTÜ VE GÖRÜNTÜ İŞLEME

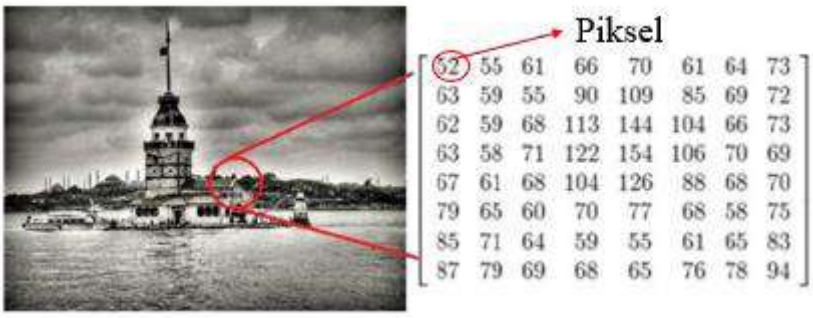
Işığın bir cisme çarpıp göze gelmesiyle görme olayı gerçekleşir. Görme duyusu insanların en önemli duyularından birisidir. Gözümüz nesneleri tanımamızı ve ayırt etmemizi sağlar. Kameralarda insanların göz yapısından ilham alınarak tasarlanmış cihazlardır. Bir nesnenin kameralar ile elde edilmiş resmine görüntü denilmektedir. Görüntü işleme tekniklerinde bazı algoritmalar, yazılımlar ve yazılımlara ait kütüphaneler kullanılmaktadır. Bu yazılımlar ile görüntü üzerine işlemler yapılarak görüntülerdeki nesnelerin ayırt edilebilmesi ve yorumlanması yapılabilmektedir. (Okur, 2015)

4.1 Görüntü İşleme

Görüntü işleme; kamera gibi aygıtlar tarafından kaydedilmiş olan görüntülerin bilgisayar ortamında özel programlar ile amaca uygun bir şekilde değiştirilerek incelemelerin yapıldığı işlemlerdir. (Demirbaş ve Dursun, 2007; Çomak vd., 2011). Son yıllarda kullanımı gittikçe artan görüntü işlemenin tasarım ve imalat alanlarında, savunma sanayi ve güvenlikte, kontrol sistemlerinde, tıp, mimari uygulamalarda, harita ve jeodezi uygulamaları gibi pek çok farklı alanda kullanılmaktadır (Samtaş ve Gülesin, 2011). Görüntü işleme de piksel ve sayısal görüntü kavramı önemli bir yer tutmaktadır.

4.1.1 Piksel

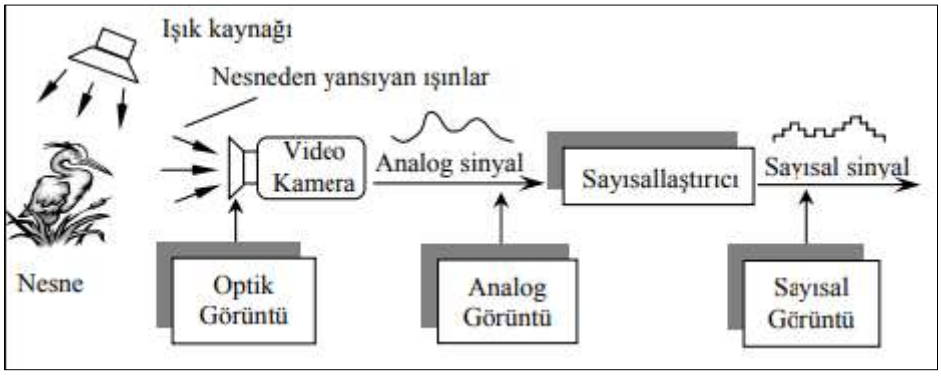
Görüntü işlemede piksel kavramı resimleri oluşturan noktalar olarak tanımlanabilir. Görüntü işleme yöntemlerinde görüntünün boyutuna göre diziler kullanılır. Örneğin iki boyutlu bir resimde kullanılan iki boyutlu dizileri matris kavramına benzetmek mümkündür. Bu nedenle sayısal bir görüntü matrisinde satır ve sütunların kesiştiği noktalar piksel olarak isimlendirilir. Piksel sayısal bir görüntünün en küçük birimidir. Şekil 4.1'de görüntüsünün bir kesiti alınan piksellerin renk karşılıkları verilmiştir (Pişkin, 2016).



Şekil 4.1: Resim içerisinde bulunan pikseller (Pişkin, 2016)

4.1.2 Sayısal görüntü

Sayısal görüntü, gerçek hayatta kamera gibi aygıtlar ile elde edilen görüntülerin sayısallaştırılması olarak ifade edilir. Gerçek hayatta kullanılan kameralardan alınan veriler farklı renk uzaylarına göre farklı boyutlarla depolanabilmektedirler (Pişkin, 2016). Şekil 4.2'de gerçek hayattaki bir görüntünün alınıp sayısallaştırılması işlemi anlatılmaktadır. Nesne ilk olarak ışık kaynağı tarafından aydınlatılır. Daha sonra nesneden yansıyan ışınlar kamera tarafından işaretlere dönüştürür. Böylece analog sinyaller sayısallaştırıcılar aracılığı ile sayısal sinyallere dönüştürülür ve sayısal görüntü oluşturulur. Elde edilen sayısal görüntüler için görüntü işleme uygulamaları gerçekleştirilebilir (Yaman vd., 2001).



Şekil 4.2: Gerçek hayattaki görüntünün alınıp sayısallaştırılması (Yaman vd., 2001)

5. YAPAY ZEKA

Yapay zeka kavramı ilk olarak John McCarthy tarafından 1955'te Honever, New Hampshire'daki Dartmouth College'da düzenlenen bir konferansda kullanılmıştır (McCarthy vd., 2006). Yapay zeka, kendi kendine öğrenme, akıl yürütme ve mantıksal karar verme gibi insan becerilerini makinelere uyarlamayı amaçlayan bir bilim alanıdır. Günümüzde yapay zeka teknolojisi bilgisayar teknolojisindeki gelişmeler ile daha da desteklenmektedir. Makinelerin artık çeşitli algoritmalarla ve yazılımlarla verileri ayrıştırmayı, yorumlamayı ve tahmin etme gibi farklı işlemleri yapabilmeleri sağlanmaktadır. Bu işlemler insan beyninde bulunan sinir ağlarına benzetilerek yapay sinir ağları kurularak yapılmaktadır. Yapay sinir ağları insan beynindeki sinir ağlarının daha az karmaşık şekilde bilgisayar ortamında yapılmış halidir.

5.1 Derin Öğrenme

Günümüzde bilgisayar donanımlarının gelişmesi sonucu grafik kartlarının işlemcilerin yükünü hafifletip çok katmanlı yapay sinir ağları ile çalışabilmek mümkün olabilmektedir. Grafik kartları ile işlemcilerin beraber çalışarak hesaplama yapabilen ve daha kısa sürede işlemler yapabilmeleri sayesinden çok katmanlı ve karmaşık sistemler tasarlanabilmektedir. Bu yeni modele derin öğrenme adı verilmektedir. Derin öğrenme yöntemi, yapay sinir ağları kullanarak çok katmanlı yapılarda büyük boyutlu veriler ile çalışmayı sağlayabilen, makine öğrenmesinde bir alt daldır. Derin öğrenme, nesnelerin özelliklerini öğrenmek için çoklu işlem katmanlarından oluşan hesaba dayalı modeller oluşturabilme imkânı sağlamaktadır. Bu modeller, görsel nesne tanımlayabilme, nesne algılayabilme, konuşma tanıyabilme ve ilaç keşfi gibi diğer birçok farklı alanlarda yapılan çalışma ve uygulamalara büyük katkılarda bulunmuşlardır. Derin evrişimli ağ ve yapılar görüntü işleme, video işleme , konuşma ve ses işlemede büyük gelişmeler yapılmasına olanak sağlarken, tekrarlayan ağlar metin ve konuşma gibi sıralı verilerde sık bir şekilde kullanılabilmektedir. (LeCun vd., 2015).

5.2 You Only Looked Once (YOLO)

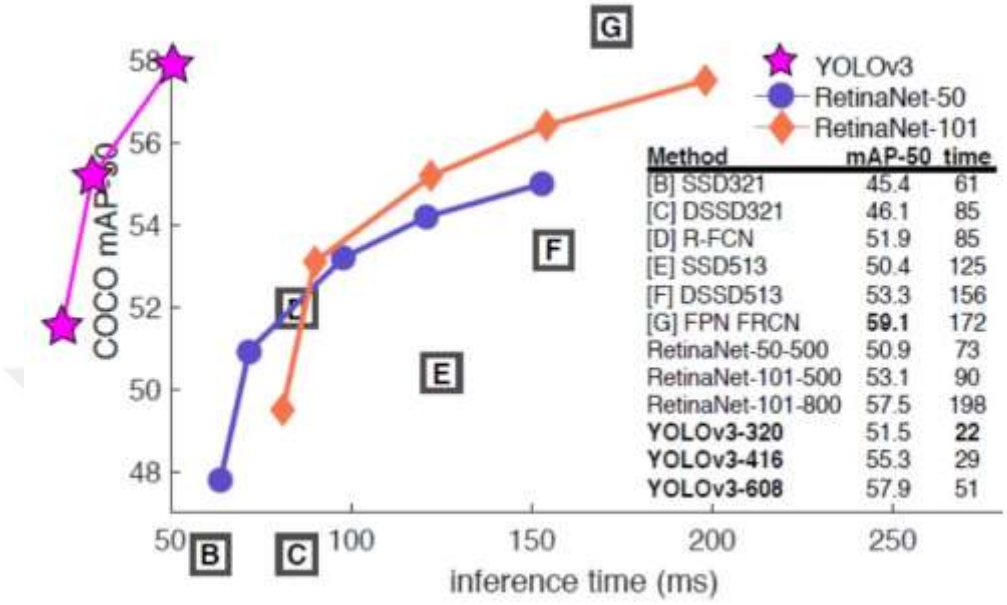
Sağlıklı bir insanın görme duyusu oldukça hızlı ve doğru çalışan, nesneleri tanıyan, birbirleriyle olan konum ilişkisini ayırt edebilen bir sistemdir. Günümüzde de artık insan beynindeki sinir sistemi örnek alınarak geliştirilen algoritmalar yardımıyla çeşitli sensörlere gerek kalmadan nesne tespit uygulamaları tasarlanmıştır. Bu uygulamalarla nesne tespiti daha hızlı ve daha doğru şekilde yapılabilmektedir. Genelde nesne tespit algoritmaları nesneyi tespit edip sınıflandırabilmek için ayrıca işlemler yapmaktadır. Algoritma ilk olarak görüntülerde nesnenin bulunma ihtimali olan bölgeleri belirleyip belirlediği bölgelerde sınıflandırıcı olarak tasarlanan sinir ağları ile her bir bölge için teker teker işlemler yaparak nesne tespit edilmektedir. Yapılan sistemlerde güzel sonuçlar alınmasına rağmen görüntüler farklı işlemlerden geçirildiği için ortaya çıkan değişken sayıları işlem gücünü arttırmaktadır. (Felzenszwalb vd., 2009)

YOLO İngilizce açılımı "You Only Looked Once", türkçe karşılığı "Sadece Bir Kez Bak" dır. Bu ismi alma sebebi algoritmanın nesnenin tespitini oldukça seri ve tek seferde yapmasından kaynaklanmaktadır.

YOLO, standart nesne tespit yöntemlerinden farklı olarak sınırlayıcı bölge kutuların olması, kutulardaki sınıfsal olasılığın hesaplanabilmesi gibi diğer tüm farklı işlem ve uygulamaları tek bir problem olarak ele alarak incelemiş ve nesne tespit sistemlerine yeni bir bakış açısı getirmiştir. YOLO algoritması ile görüntüler üzerinde nesnelerin nerede olduğunu belirleyip tespit etmek için görüntüye sadece bir kez bakılması yeterli olacaktır. Sadece bir ağ ile aynı zamanda çok sayıda sınırlayıcı kutu tahmin edilebilmekte ve her kutu için sınıf ihtimalleri analiz edilip sonuçları tahmin edilebilmektedir. Bu model standart nesne tespit yöntemlerine göre daha faydalıdır.

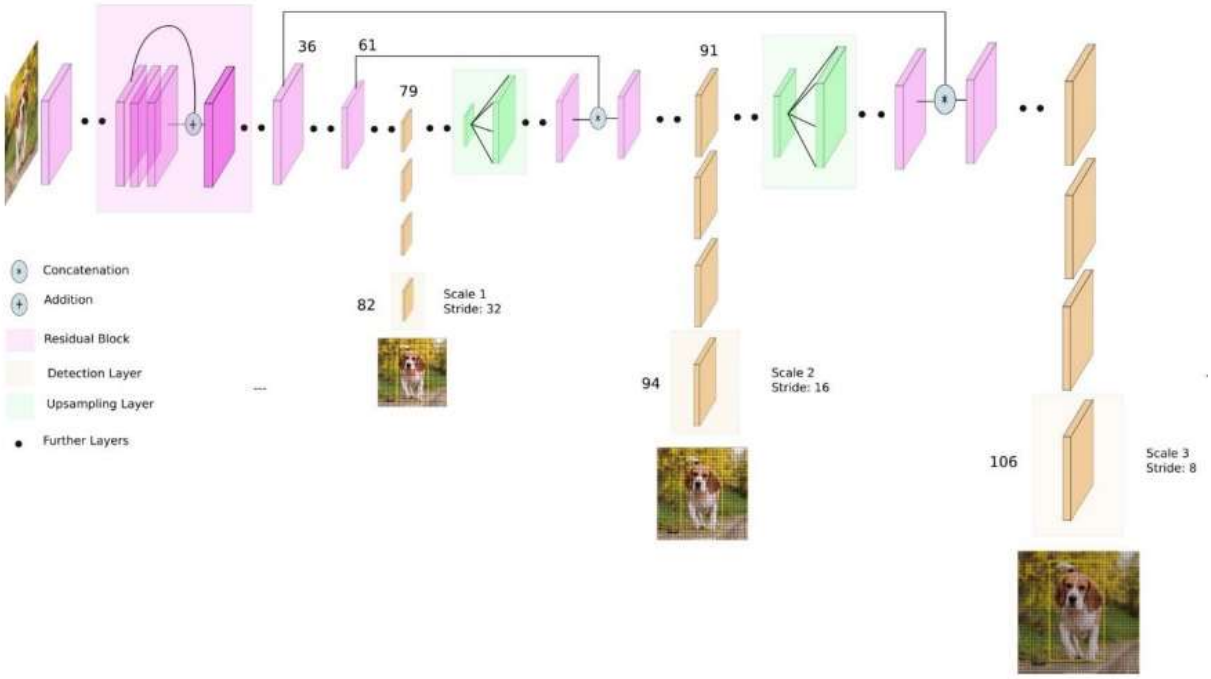
YOLO-V2 modeli YOLO modelinin geliştirilmiş halidir. YOLO-V2'de nesne tespitinde kullanılmak üzere 19 katmanın yanında fazladan 11 katmanı bulunan derin ağ mimarisi kullanılmaktadır. 30 katmanı olan bu modelde genelde küçük nesneleri algılamada sıkıntı yaşanmaktadır. Bunun nedeni görüntülerde resimler küçüldükçe özniteliklerini kaybetmesi ve görüntünün işlenememesinden kaynaklanmaktadır. Bu sıkıntıları gidermek için YOLO-V3 modeli geliştirilmiştir. YOLO-V3 modeli YOLO ve YOLO-V2 kullanılarak geliştirilmiştir. YOLO-V3 mimarisinde eğitilmiş toplam 106 evrişimsel katman bulunmaktadır. (He vd., 2016)

YOLO-V3’ de büyük nesnelerinde küçük nesnelerinde tespit edilmesinde iyi bir performans sağlamaktadır. COCO veri seti kullanılarak yapılan çalışmalarda YOLO-V3 nesne tespit sistemi diğer nesne tespit sistemleriyle karşılaştırıldığında çok daha iyi sonuçlar verdiği Şekil görmek münkündür.



Şekil 5.1: YOLO V3'ün Diğer algoritmalar karşısında hız ve doğruluk açısından karşılaştırması (Redmon ve Farhadi, 2018)

YOLO-V3 nesne tespit ağında ilk olarak görüntüler N x N (13x13, 26x26, 52x52 gibi) ızgaralara ayrılır. Izgaralara ayrıldıktan sonra ilk olarak 82. katmanda büyük boyuttaki nesneler, 94. katmanda orta boyuttaki nesneler ve son kısımda ise 106. katmanda küçük boyuttaki nesnelerin tespiti gerçekleşir. Bu tespit işlemlerinde nesnelerin çevresinde sınır çerçeveleri çizilmesi ızgaraların eşik değerine bağlıdır. Hesaplanan olasılık oranlarının eşik değerini aşmasının ardından ızgaralar içerisinde merkez koordinat değerleri (x ve y) ile genişlik ve yükseklik (w ve h) değerlerine bağlı olarak bir çerçeve çizilir. Böylece en son çıkış katmanında nesnelerin çevresine sınır çerçeveleri ile birlikte nesne tespiti yapılmış görüntü elde edilir. (Yasak, 2021)



Şekil 5.2: YOLO V3 Ağ yapısı (Kathuria, 2018)

6. KULLANILAN YAZILIMLAR

6.1 Open CV

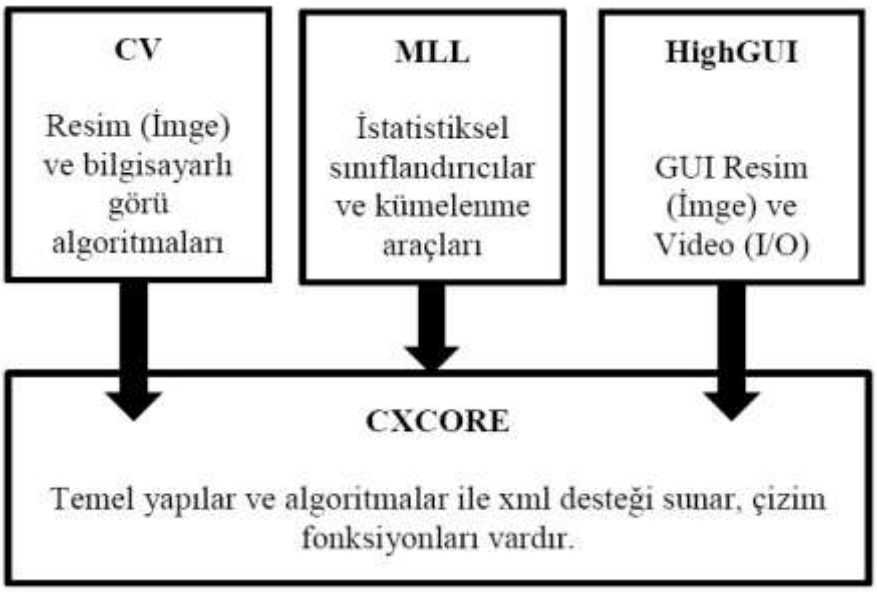
İlk sürümü 1999 yılında intel firması tarafından çıkarılan OpenCv, gerçek zamanlı görüntü işleme ve bilgisayarlı görü yöntemlerinin gerçekleştirildiği bir platformdur. OpenCV'nin anlamı "Open Source Computer Vision Library" dir. Türkçe karşılığı "Açık Kaynak Kodlu Bilgisayarlı Görü" dür. (Boyraz, 2015)

OpenCV, Windows, IOS, Linux, PSP, Android gibi birçok yerleşik veya mobil sistemlerde çalışabilen, C ve C++ yazılım dili ile yazılmış, görüntü işleme uygulamaları ve gerçek zamanlı görme için tasarlanmış açık kaynak kodlu bir kütüphanedir. Açık kaynak kodlu olması, eğitim amaçlı ve ticari kullanımının ücretsiz olması diğer görüntü işleme kutüphanelerine göre daha kullanışlı hale getirmektedir. (Kutlu, 2013)

OpenCV, Willow Garage şirketi tarafından desteklenmektedir. OpenCV kütüphanesi SourceForge web sitesinden indirilebilmektedir. Halihazirda olan C yazılım sürümüne ek olarak OpenCV 2.0 sürümüyle birlikte, C++ yazılımlı ara yüzü de eklenmiştir. OpenCV kütüphanesi kullanılarak görüntü işleme yöntemleri ve bilgisayarlı görü teknikleri kullanılarak;

- ✓ Nesne Tanıyabilme
- ✓ Yüz Tanıyabilme
- ✓ İşaret Dili Tanıyabilme
- ✓ Hareket Yakalayabilme
- ✓ Hareketsel Takip gibi uygulamalar yapılabilmektedir.

OpenCV'de kütüphanesi CV, MLL, HighGUI, CXCore ve CvAux bileşeni olmak üzere beş bileşenden oluşmaktadır.



Şekil 6.1: Python OpenCV kütüphanesinin bileşenleri

- CV bileşeni, filtre işlemleri, geometric dönüşümler, basit resim işleme, renk dönüşümleri, şekil tanıma işlemleri ve kamera kalibrasyou gibi işlemler yapılabilmektedir.
- MLL bileşeni, derin öğrenme için gerekli sayısal verilere elde etmek ve eldeki verileri sınıflamak için tasarlanan bir kütüphanedir. İstatistiksel model işlemleri, yapay sinir ağları işlemleri gibi birçok işlemi yapabilmektedir.
- HighGUI bileşeni, grafik arabirimidir. Resim ve videoları kaydetme, yükleme ve hafızadan slime işlemlerini yapan bir kütüphanedir.
- CXCore bileşeni, XML desteği sağlayan, matemaktiksel işlemler, matris işlemleri ve 2D grafik çizimi gibi gerçekleştirebilen bir kütüphanedir.
- CvAux bileşeni, şablon ve şekil eşleştirmeleri, bir nesnenin ana hatlarını bulmak, yüz tanıyabilme, vücut hareketlerini tanıyabilme gibi deneysel algoritmaya sahip bir kütüphanedir. (Brodski ve Haehler, 2008)

6.2 Python

Python 1991 yılında Guido Van Rossum geliştirdiği çok güçlü ve yüksek seviyeli, dinamik nesne yönelimli programlama dilidir.(Harwani, 2011) Python platformlardan ayrı çalışan bir yazılım dili olduğundan dolayı Windows, Linux, Mac OS X, gibi bir çok işletim

sistemleri üzerinde çalışabilmektedir. Phyton programlama dili Eric S. Raymond gibi birçok yazılımcı ve Google şirketinin tercih ettiği bir yazılım dili haline gelmiştir (Swaroop, 2003)

Python'da derleme işlemi olmadığı için, bu yazılım dili ile hızlı bir şekilde program geliştirilebilir. Python açık kaynak kodlu, yüksek seviyeli ve nesne yönelimli bir programlama dilidir. (Özgül F., 2016).

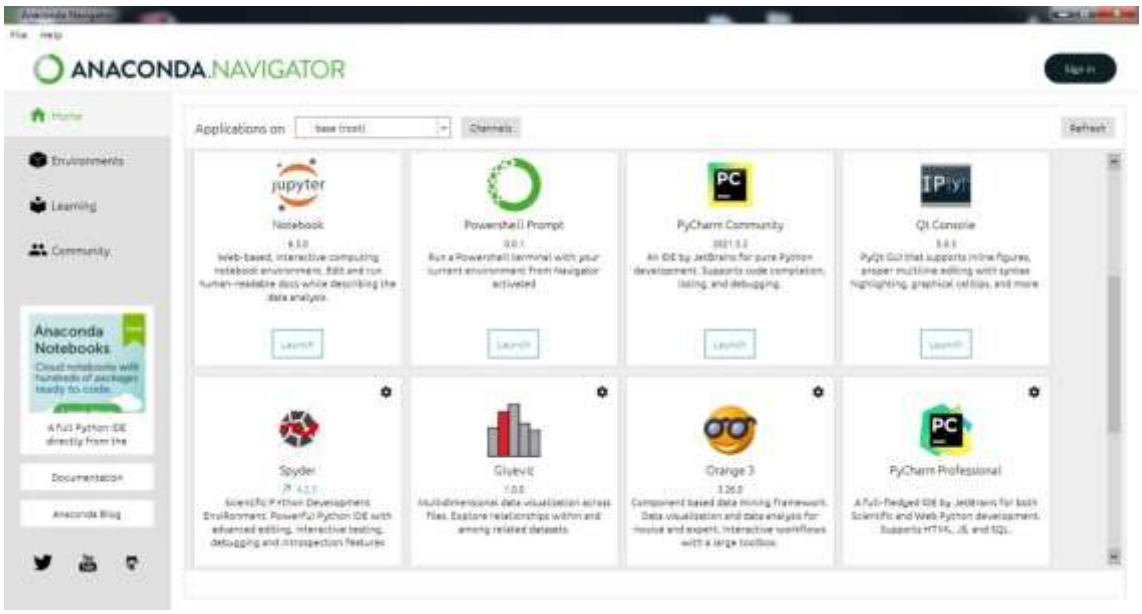
Şekil 6.2: Python dilinin özellikleri (Saralıoğlu vd., 2014)

6.3 NumPy

NumPy kütüphanesi, Python yazılım dilini kullanırken çok boyutlu dizi ve matrislerle çalışma imkanı sağlayan ileri düzey matematiksel işlemler yapabilen bir kütüphanedir. Açık kaynak kodla çalışan, sayısal değerleri işleyebilmesinden kaynaklı yazılıcımların çok fazla kullandığı NumPy kütüphanesi Phyton yazılım dilinin önemli bir kütüphanesidir. Çok katmanlı dizinleri ve matris sistemlerini içermektedir. Dizin ve matrislerde çok sayıda sayısal işlemler yaparak analiz çıktıları sağlayabilmek için kullanılabilmektedir. (İpar, 2022)

6.4 Anaconda

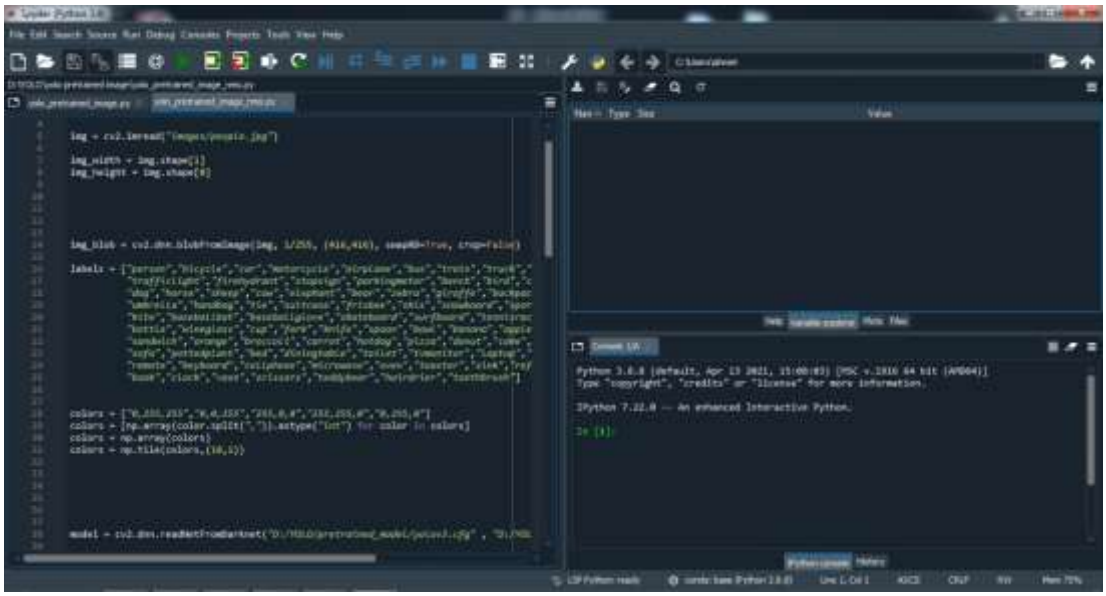
Anaconda programı, paket program yönetim ve dağıtımını daha basitleştirmeyi amaçlayan Python ve R programlama dillerinin bir dağıtımıdır. İçerisinde bir çok kütüphane barındırır ve bu kütüphanelere kolayca erişimi sağlar. Dağıtım, Windows, Linux ve macOS için uygun veri bilimi paketlerini içermektedir.



Şekil 6.3: Anaconda program arayüzü ve kütüphaneler

6.5 Spyder

Spyder, Python yazılım dili ile tasarlanmış ve bilim adamları, mühendis ve veri bilimcileri için tasarlanmış bilimsel hesaplamaların yapılabildiği Python yazılımı için ücretsiz ve açık kaynaklı geliştirilen bilimsel bir ortamdır. Görüntü işlemede, veri madenciliğinde, bilimsel bir dizinin görselleştirmesini test etmede, analiz, hata ayıklayabilme gibi tam özellikli bir geliştirme aracının özelliklerini sunmaktadır.



Şekil 6.4: Spyder program arayüzü

7. YOLO İLE NESNE TESPİT UYGULAMASI

Tez çalışmasında yapılan uygulamada ilk olarak nesne tespiti için gerekli görüntüler Şanlıurfa Küçük Akziyaret köyünde yer alan elma ağaçlarından elde edilmiştir. Şekil 7.1 de daha elma tespit edilmemiş görüntü bulunmaktadır.

Şekil 7.1: Tespit edilmemiş elma görüntüleri

Veri elde edilecek elma ağacı fotoğrafları 2022 yılının Eylül ayında cep telefonu kamerası ile çekilmiştir. Telefondan alınan görüntülerin boyutları 4640 x 2088 pikseldir. Görüntü işleme esnasında programın takılmaması ve hızlı işlenmesi için görüntünün boyutları 640 x288 piksel olarak küçültülmüştür. Görüntülerin tespitinde OpenCV ve NumPy kütüphaneleri kullanılarak, Phyton yazılım diliyle YOLO ağı kullanılacaktır.

7.1 Nesne Tespiti İçin Gerekli Yazılım

Çalışmamızda ilk olarak Yolo'nun resmi sitesi https://pjreddie.com/darknet/yolo/ üzerinden nesne tanıma işlemi için cfg ve weights dosyaları indirilmiştir. Şekil 7.2 de sarı kutu içine alınmış YOLOv3-416 veri seti kullanılmıştır. Bu veri seti 416 x 416 resimlerle

eğitilen bir modeldir. Cfg ve weights dosyaları derin öğrenme için gerekli parametreler ve sayılar bulunduran ve sürekli güncellenen dosyalardır.

Performance on the COCO Dataset

Model	Train	Test	mAP	FLOPS	FPS	Cfg	Weights
SSD300	COCO trainval	test-dev	41.2	-	46		link
SSD500	COCO trainval	test-dev	46.5	-	19		link
YOLOv2 608x608	COCO trainval	test-dev	48.1	62.94 Bn	40	cfg	weights
Tiny YOLO	COCO trainval	test-dev	23.7	5.41 Bn	244	cfg	weights
SSD321	COCO trainval	test-dev	45.4	-	16		link
DSSD321	COCO trainval	test-dev	46.1	-	12		link
R-FCN	COCO trainval	test-dev	51.9	-	12		link
SSD513	COCO trainval	test-dev	50.4	-	8		link
DSSD513	COCO trainval	test-dev	53.3	-	6		link
FPN FRCN	COCO trainval	test-dev	59.1	-	6		link
Retinanet-50-500	COCO trainval	test-dev	50.9	-	14		link
Retinanet-101-500	COCO trainval	test-dev	53.1	-	11		link
Retinanet-101-800	COCO trainval	test-dev	57.5	-	5		link
YOLOv3-320	COCO trainval	test-dev	51.5	38.97 Bn	45	cfg	weights
YOLOv3-416	COCO trainval	test-dev	55.3	65.86 Bn	35	cfg	weights
YOLOv3-608	COCO trainval	test-dev	57.9	140.69 Bn	20	cfg	weights
YOLOv3-tiny	COCO trainval	test-dev	33.1	5.56 Bn	220	cfg	weights
YOLOv3-spp	COCO trainval	test-dev	60.6	141.45 Bn	20	cfg	weights

Şekil 7.2: Coco Date seti verileri

Gerekli dosyalar indirildikten sonra kodlama kısmında ilk olarak kütüphaneler eklenmiştir. Görüntü işleme iki kütüphane kullanılmıştır. Bunlar Opencv ve Numpy'dır. Kütüphanelerin eklendiği kodlar Şekil 7.3 de belirtilmiştir.

```
import cv2
import numpy as np
```

Şekil 7.3: Kütüphane için gerekli kodlar

Nesne tanıma yapılacak resmin okunabilmesi için cv2.imread fonksiyonu kullanılmıştır. Bu fonksiyon ile resim çağırılarak okuma işlemi yapılacaktır. Yazılan kodlar Şekil 7.4 te belirtilmiştir.

```
img = cv2.imread("images/people.jpg")
```

Şekil 7.4: Resim okuma için gerekli fonksiyon

Yolo algoritmasında resmin okunabilmesi için resmin eni ve boyunu bilmek gerekiyor. Bu oranları bulabilmek için numpy kütüphanesinin sağladığı datashape metodu kullanılmıştır. Datashape metodu kullanılarak oluşturulan değişkenlerin yazıldığı kodlar Şekil 7.5 da belirtilemiştir.

```
img_width = img.shape[1]
img_height = img.shape[0]
```

Şekil 7.5: Resimlerin en ve boy oranları için gerekli fonksiyon

Resimlerin en ve boy oranlarına ulaştıktan sonra nesne tespiti yapabilmek için görüntüyü yolo algoritmasına vermek gereklidir. Bunun içinde resmi blob formatına çevirmek gereklidir. Blob formatı resmin dört boyutlu tensörlere çevrilmiş halidir. Tensörler vektörlerin ve matrisler sizin istediğiniz sayıda boyuta genelleştirilmesidir. Coco veri seti olarak 416x416 resimlerle eğitilen modeli kullanacağımız için resmimizi blob formatına çevirirken resmin ebatlarını 416x416 olarak ayarlamamız gerekiyor. Resmi blob formatına dönüştürmek için gerekli kodlar Şekil 7.6 de belirtilmiştir.

```
img_blob = cv2.dnn.blobFromImage(img, 1/255, (416,416), swapRB=True, crop=False)
```

Şekil 7.6: Resmin blob formatına dönüştürülmesi için gerekli fonksiyon

Resmimizde uygulamamızın tanıyacağı nesneleri tanımlaması ve belirtmesi için etiketleme (labels) işlemi yapılmıştır. Her nesne için farklı renkte yapılarak birbirinden ayırt edilmesi sağlanmıştır. Bu işlem için yine numpy kütüphanesi kullanılmıştır. Bunun için gerekli kodlar Şekil 7.7 de belirtilmiştir.

```
labels = ["person","bicycle","car","motorcycle","airplane","bus","train","truck","boat",
          "trafficlight","firehydrant","stopsign","parkingmeter","bench","bird","cat",
          "dog","horse","sheep","cow","elephant","bear","zebra","giraffe","backpack",
          "umbrella","handbag","tie","suitcase","frisbee","skis","snowboard","sportsball",
          "kite","baseballbat","baseballglove","skateboard","surfboard","tennisracket",
          "bottle","wineglass","cup","fork","knife","spoon","bowl","banana","apple",
          "sandwich","orange","broccoli","carrot","hotdog","pizza","donut","cake","chair",
          "sofa","pottedplant","bed","diningtable","toilet","tvmonitor","laptop","mouse",
          "remote","keyboard","cellphone","microwave","oven","toaster","sink","refrigerator",
          "book","clock","vase","scissors","teddybear","hairdrier","toothbrush"]


colors = ["0,255,255","0,0,255","255,0,0","255,255,0","0,255,0"]
colors = [np.array(color.split(",")).astype("int") for color in colors]
colors = np.array(colors)
colors = np.tile(colors,(18,1))
```

Şekil 7.7: Resimlerin etiketleri

Oluşturacağımız modeli çalışmaya dahil edebilmek için readNetFromDarknet metodu kullanarak daha önce Yolo resmi sitesinden indirdiğimiz veri setinin cfg ve weight dosyalarını değişkene tanımlamamız gerekiyor. Bunun için gerekli kodlar Şekil 7.8 da belirtilmiştir.

```
model = cv2.dnn.readNetFromDarknet("D:/YOLO/pretrained_model/yolov3.cfg" ,
                                   "D:/YOLO/pretrained_model/yolov3.weights")
```

Şekil 7.8: Cfg ve weight dosyalarının tanımlanması

Cfg ve weight dosyalarını programa dahil ettikten sonra resimlerde tanıma işlemi yapabilmek için ve tanıma işlemi sonrası tanınan resimlere etiket verebilmesi için model.getLayerNames metodu kullanılmıştır. Bunun için gerekli kodlar Şekil 7.9 da belirtilmiştir.

```
layers = model.getLayerNames()
output_layer = [layers[layer -1] for layer in model.getUnconnectedOutLayers()]

model.setInput(img_blob)

detection_layers = model.forward(output_layer)
```

Şekil 7.9: Resimlere etiket verebilmesi için gerekli fonsiyon

Resimlerden tanıma yapılırken nesneyi ne oranda tanıyabildiğini belirleyebilmek için numpy kütüphanesinden np.argmax metodu kullanılarak bir güven skoru oluşturuldu. Bu güven skoru bizim istediğimiz oranlarla ayarlanarak belirlediğimiz skora göre resimde belirlenen nesnelere çerçeve (sınırlayıcı kutu) çizilip çizilmemesine karar verilebilir. Çerçevenin çizileceği resmin merkezi, eni ve boyu ve çerçevenin rengi belirlenebilir. Çerçevede üzerinde güven skoruda gösterip nesneyi yüzde kaç oranında tanıdığı belirlenebilir. Bunlar için gerekli kodlar Şekil 7.10 de belirtilmiştir.

```
for detection_layer in detection_layers:
    for object_detection in detection_layer:

        scores = object_detection[5:]
        predicted_id = np.argmax(scores)
        confidence = scores[predicted_id]

        if confidence > 0.10:

label = labels[predicted_id]
bounding_box = object_detection[0:4] * np.array([img_width, img_height, img_width, img_height])
(box_center_x, box_center_y, box_width, box_height) = bounding_box.astype("int")


            start_x = int(box_center_x - (box_width / 2))
            start_y = int(box_center_y - (box_height / 2))

    end_x = start_x + box_width
    end_y = start_y + box_height

    box_color = colors[predicted_id]
    box_color = [int(each) for each in box_color]

    label = "{}: {:.2f}%".format(label, confidence * 100)
    print("predicted object {}".format(label))
```

Şekil 7.10: Çerçeve özellikleri ve güven skoru için gerekli kodlar

Yolo algoritması kullanılarak yapılan nesne tanıma işlemlerinde bazen aynı nesneye birden fazla çerçeve oluşabiliyor. Bu çerçevelerde istenilen çerçeveyi göstermek mümkündür. En yüksek güven skoru olan yani en iyi tanıma işlemi yapılan nesneye sadece çerçeve oluşturtulabilir ve güven skoru düşük olanları göstermeyebiliriz. Bunu yapabilmemiz için Non – Maximum Suppression (Maximum olmayanı bastırma) yöntemini kullanmamız gerekiyor. Bu yöntemle resimde güven skoru düşük tanımalar için çerçeve hiç çizilmeyecektir. Non – Maximum Suppression metodu üç işlemden oluşmaktadır.

Non – Maximum Suppression metodu ilk işlem for döngüsünden önce üç adet boş liste oluşturulması gerekli. Bunlardan biri tanımaların (tahmin kimlikleri) tutulduğu bir liste, ikincisi çerçeve özelliklerinin tutulduğu bir liste, üçüncüsüde güven skorlarını tutulduğu bir listedir. Bu listeler için gerekli kodlar Şekil 7.11 de belirtilmiştir.

```
ids_list = []
boxes_list = []
confidences_list = []
```

Şekil 7.11: Non – Maximum Suppression yöntemi için gerekli listeler

Non – Maximum Suppression yönteminde ikinci işlem ise yapılan tanıma işlemleri ile nesnelerin tahmin kimlikleri, çerçeve özellikleri ve güven skorları ile oluşturulan boş listeler doldurulacaktır. Bunun için for döngüsü içerisinde .append metodu kullanılacaktır. Bu metod listelerin sonuna değerler eklemeyi sağlar. Bu işlemler için gerekli kodlar Şekil 7.12 de belirtilmiştir.

```
ids_list.append(predicted_id)
confidences_list.append(float(confidence))
boxes_list.append([start_x, start_y, int(box_width), int(box_height)])
```

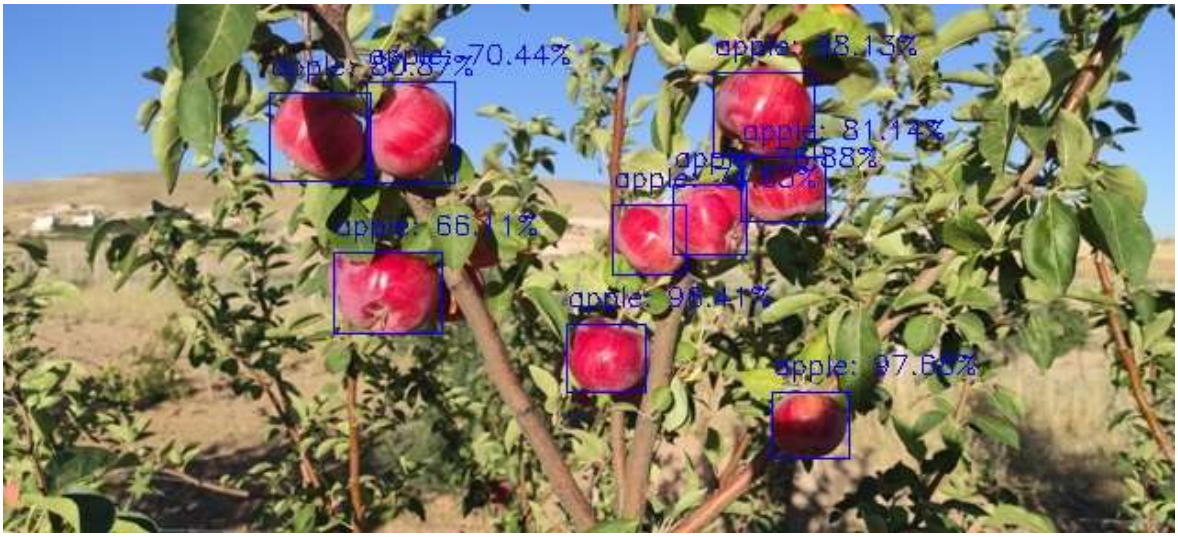
Şekil 7.12: Boş listelere değerler eklemek için gerekli kodlar

Non – Maximum Suppression yönteminde üçüncü işlem tanıma işlemi yapılarak doldurulan boş listelerin içerisinde cv2.dnn.NMSBoxes metodu kullanılarak en yüksek güvenilirliğe sahip tahmin kimliklerine çerçeveleri çizmek olacaktır ve çizilen çerçevelere yine etiketler yazdırılacaktır. Bu işlemler için gerekli kodlar Şekil 7.13 de belirtilmiştir.

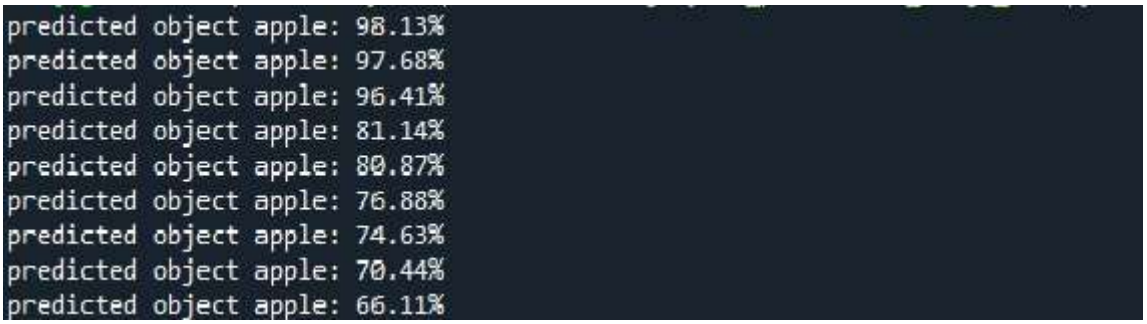
```
max_ids = cv2.dnn.NMSBoxes(boxes_list, confidences_list, 0.5, 0.4)

for max_id in max_ids:

    max_class_id = max_id
    box = boxes_list[max_class_id]

    start_x = box[0]
    start_y = box[1]
    box_width = box[2]
    box_height = box[3]

    predicted_id = ids_list[max_class_id]
    label = labels[predicted_id]
    confidence = confidences_list[max_class_id]
```

Şekil 7.13: En yüksek güvenilirliğe sahip tanımalar için gerekli kodlar

Yazılan kodlar sonucu program çalıştırıldığında programın başarılı bir şekilde çalıştığı görülmüştür. Şekil 7.14 de görüldüğü üzere resimde bulunan dokuz adet elmanın dokuzunun da tanıma işlemi yapıldığı görülmüştür. Üzerlerine çizilen çerçeverler de görüldüğü üzere güven skorları çok yüksek değerlerdedir. Şekil 7.15 de resimde bulunan dokuz adet elmanın yüzde olarak güven skorlarları bulunmaktadır.



Şekil 7.14: Tespit edilmiş elma görüntüleri



Şekil 7.15: Elma tanıma işlemi güven skorları

7.2 Rekolte Hesaplama

Tarladaki elma ağaçlarının rekoltesini hesaplayabilmek için üç ağaç seçilip bunlar üzerinden Yolo yazılımı ile bir veri eğitim seti çıkarılmıştır. Bu veri eğitim setinde ağaçdaki gerçek elma sayısı ile uygulamamızın tespit ettği elma sayısı ve doğruluk güven skorları bulunmaktadir. Bu örnek ağaçlara göre yazılımımızın doğru çalışma olasılığı hesaplanmıştır. Uygulamamız ağaçlardaki elmaların çok büyük bir kısmını tanımıştır fakat bazı elmaları tanıyamamıştır. Tanıyamadığı elmalar sarı halkalar içine alınarak belirtilmiştir. Elmaları tanıyamamasındaki sebepler: bir başka elmanın arkasında kalıp elmanın bir kısmının görünmesi, çekilen fotoğrafların çözünürlüğünün düşük olması, elma boyutlarının küçük olması, fotoğrafın uzaktan çekilmesinden kaynaklanmadır.

Şekil 7.16: Birinci ağaçtaki tespit edilen ve tespit edilemeyen elmalar

Çizelge 7.1: Birinci ağacın veri eğitim seti.

Toplam Elma Sayısı	Tespit Edilen Elma Sayısı	Güven Skoru(%)
15	12	89.81 87.24 80.13 80.06 77.04 75.13 72.51 52.85 37.07 35.32 35.06 32.80

Şekil 7.17: İkinci ağaçtaki tespit edilen ve tespit elemeyen elmalar

Çizelge 7.2: İkinci ağacın veri eğitim seti.

Toplam Elma Sayısı	Tespit Edilen Elma Sayısı	Güven Skoru(%)
12	8	95.92 85.35 67.10 48.56 31.30 29.31 24.31 21.65

İkinci ağaçta bir yaprağı renginden dolayı elma olarak yanlış tespit etmekte ve bu ağaçtaki elmalar çok sık olduğundan dolayı bir kısmı tespit edilememiştir.

Şekil 7.18: Üçüncü ağaçtaki tespit edilen ve tespit elemeyen elmalar

Çizelge 7.3: Üçüncü ağacın veri eğitim seti.

Toplam Elma Sayısı	Tespit Edilen Elma Sayısı	Güven Skoru(%)
13	9	95.29 94.65 80.32 75.67 64.35 57.02 39.84 35.78 30.21

Üç veri eğitimin seti bir üzerinden yapılan hesaplamalarla toplamda 40 elmadan 29 tanesi uygulama tarafından tespit edilmiş, 11 tanesi tespit edilememiştir. Uygulamamızın üç ağaç üzerinden güven skoru %72.5 olarak çıkmıştır.

8. SONUÇ VE ÖNERİLER

8.1 Sonuç

Bu tez çalışmasında phyton diliyle yazmış olduğumuz uygulama ile yapay zeka yardımıyla elma ağaçlarından elde edilen görüntülerden gerçek zamanlı elma tespiti yapmaktayız. Uygulamamız %72.5 güven skoru ile çalışmaktadır.

Çizelge 8.1: Uygulama Başarı Durumu.

Toplam Elma Sayısı	Tespit Edilen Elma Sayısı	Güven Skoru(%)
40	29	72.5

Bazı elmaların tespit edilememesinin sebepleri, uzaktan çekilmesi, elmaların birbirini engellemesi, yaprakların elmaları kapatması, yeterli ışık miktarının olmaması gibi olumsuz nedenler bulunmaktadır. Ağaç üzerinde yapılan elma rekoltesi tahmini için yapılan çalışmada farklı programlar ve yazılım dilleri kullanılmış ve uygulama geliştirilmiştir. Bu tez çalışması ile yapılan uygulama tamamen tamamen açık kaynak kodlu programlar ve kütüphaneler ile oluşturulmuştur.

8.2 Öneriler

Rekolte tahmini yapılacak ağacın görüntüleri elde edilirken yeterli ışık ayarlanıp, daha az yapraklı ağaçların seçilmesi ve uygun mesafeden resimlerin çekilmesiyle daha olumlu sonuçlar elde edilebilir. Ayrıca yakın mesafeden daha kaliteli fotoğraf makineleri ile çözünürlüğü daha yüksek fotoğraflarda uygulamanın daha iyi sonuçlar vermesini sağlayabilir. Bu çalışmada coco data veri setinin 416 x 416 resimlerin eğitildiği model kullanılmıştır, gerekli bilgisayar altyapısı sağlanırsa coco data serisinin 608 x 608 ebatlarında eğitilmiş veri seti ile çok daha iyi sonuçlar elde edilebilir ve eğitimin kalitesi arttırılabilir. Yapılan uygulamanın daha da geliştirilmesi ile nesne tespiti ve rekolte hesaplaması artık bilgisayarlı görü ile daha hızlı ve daha güvenilir bir şekilde yapılabilir.

KAYNAKLAR

Campbell , JB. And Wynne, RH., 2011. Introduction to Remote Sensing, fifth edition, The Guilford Press, New York.

Fischer, WA., Hepmhill, WR. And Kover, A., 1976. Progress in Remote Sensing, Photogrometria, Vol. 32, pp. 33-72.

Buiten, H.J. and Clevers, JGPW. 1993. Land Observation by Remote Sensing: Theory and Applications, Vol. 3 of Current Topics in Remote Sensing. Gorden&Breach

Lillesand, TM., Kiefer, RW. And Chipman, JW., 2004. Remote Sensing and Image Interpratation, fifth ed. Jhon Wiley & Sond, New York, NY.

Jehnsen J.R., (1996) *Introductory Digital Image Processing*: A Remote Seıısing Perspective.3 rd edition, Pearson Prentice Hall, USA, 1-526 pp.

Örmeci C, (1987) Uzaktan Algılama: Temel Esaslar ve Algılama Sistemleri. İstanbul Teknik Üniversitesi Matbaası, İstanbul, 1-112 s.

McCarthy, J., Minsky, M. L., Rochester, N., & Shannon, C. E., (2006). A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955. AI magazine, 27(4), 12.

LeCun, Y., Bengio, Y., & Hinton, G., (2015). Deep learning. Nature, 521, 436. doi:10.1038/nature14539

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D., (2009). Object detection with discriminatively trained part-based models. IEEE transactions on pattern analysis and machine intelligence, 32(9), 1627-1645.

He, K., Zhang, X., Ren, S., & Sun, J., (2016). Deep residual learning for image recognition. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.

Redmon, J. ve Farhadi, A., (2018). Yolov3: An Incremental Improvement. arXiv preprint arXiv:1804.02767.

Harwani, B. M., 2011, Introduction to Python Programming and Developing GUI Applications with PyQT, Cengage Learning

Brodski G. And Haehler A., "Learning OpenCV: Computer Vision With the openCV Library", O'Reilly Media, Amerika Birleşik Devletleri, 16-17 (2008).

Özdemir, H., 2010, Uzaktan algılama ders notu.pdf, *İstanbul Üniversitesi*, 20.

Özgül, F., (2020). *Her Yönüyle Python.* Kodlab Yayınları, İstanbul.

Pişkin, M., (2016). *Opencv ile görüntü işleme.* https://mesutpiskin.com/blog/wp-content/uploads/2017/01/OpenCV%20Kitap.pdf (Son erişim tarihi:25.11.2022)

Swaroop, C. H., 2003 A Byte of Python (Son erişim tarihi:05.12.2022)

Okur S., "Görüntü İşleme Yöntemleri Kullanılarak Gözdeki Damarların Tespit Edilmesi" Yüksek Lisans Tezi, T.C. Fırat Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Bilgisayar Eğitimi Anabilim Dalı, Bilgisayar Sistemleri Eğitimi, Elazığ 2015

Yasak S.S., "Coğrafyada yapay zeka uygulamaları: YOLO V3 ile gerçek zamanlı kayaç tespit uygulaması örneği" Yüksek Lisans Tezi, T.C. Marmara Üniversitesi Sosyal Bilimler Enstitüsü Coğrafya Anabilim Dalı, Coğrafya, İstanbul 2021

Boyraz Ö.F., "Mobil Damar Görüntüleme Cihazı Tasarımı", Yüksek Lisans Tezi, T.C. Sakarya Üniversitesi, Fen Bilimleri Enstitüsü, Elektrik-Elektronik Mühendisliği Anabilim Dalı, Sakarya, 2015

Kutlu H., "İnsan Bilgisayar Etkileşimli Görüntü İşleme Uygulamaları", Yüksek Lisans Tezi, T.C. Fırat Üniversitesi, Fen Bilimleri Enstitüsü, Elektronik ve Bilgisayar Eğitimi Anabilim Dalı, Bilgisayar Sistemleri Eğitimi, Elazığ 2013.

İpar M., "Makine Öğrenimi Yardımıyla Hasta Dökümanları İçerisindeki Laboratuvar Değerlerinin Yakalnıp İlişki Kurularak Otomatik Bir Şekilde Eşleştirilmesi", Yüksek Lisans Tezi, T.C. Maltepe Üniversitesi, Lisansüstü Eğitim Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, İstanbul 2022

Demirbaş, H. Y. & Dursun, İ., (2007). Buğday tanelerinin bazı fiziksel özelliklerinin görüntü işleme tekniğiyle belirlenmesi. *Journal of Agricultural Sciences, 13*(03), 176-185.

Samtaş, G., & Gülesin, M., (2012). Sayısal görüntü işleme ve farklı alanlardaki uygulamaları. *Ejovoc (Electronic Journal of Vocational Colleges), 2*(1), 85-97.

Yaman, K., Sarucan, A., Mehmet, A. & Aktürk, N., (2001). Dinamik çizelgeleme içingörüntü işleme ve arıma modelleri yardımıyla veri hazırlama. *Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, *16*(1), 19-40.

Saralıoğlu, E. , Yıldırım, D. , Güngör, O., 2014, Uzaktan algılama araştırmacılarına yönelik python ara yüzü ve arcgıs yazılımı eklentisi, 5. Uzaktan Algılama-Cbs Sempozyumu (UZAL-CBS 2014)

Kathuria, A. (2018), *What's New in YOLO v3*, https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b Son Erişim Tarihi: 05.12.2022

EKLER

EK 1: Python Dili İle Geliştirilen Uygulamaya Ait Kodlar

EK 1: Python Dili İle Geliştirilen Uygulamaya Ait Kodlar

```
import cv2
import numpy as np
img = cv2.imread("images/apple.jpg")
img_width = img.shape[1]
img_height = img.shape[0]

img_blob = cv2.dnn.blobFromImage(img, 1/255, (416,416), swapRB=True, crop=False)

labels = ["person","bicycle","car","motorcycle","airplane","bus","train","truck","boat",
        "trafficlight","firehydrant","stopsign","parkingmeter","bench","bird","cat",
        "dog","horse","sheep","cow","elephant","bear","zebra","giraffe","backpack",
        "umbrella","handbag","tie","suitcase","frisbee","skis","snowboard","sportsball",
        "kite","baseballbat","baseballglove","skateboard","surfboard","tennisracket",
        "bottle","wineglass","cup","fork","knife","spoon","bowl","banana","apple",
        "sandwich","orange","broccoli","carrot","hotdog","pizza","donut","cake","chair",
        "sofa","pottedplant","bed","diningtable","toilet","tvmonitor","laptop","mouse",
       "remote","keyboard","cellphone","microwave","oven","toaster","sink","refrigerator",
        "book","clock","vase","scissors","teddybear","hairdrier","toothbrush"]

colors = ["0,255,255","0,0,255","255,0,0","255,255,0","0,255,0"]
colors = [np.array(color.split(",")).astype("int") for color in colors]
colors = np.array(colors)
colors = np.tile(colors,(18,1))

model = cv2.dnn.readNetFromDarknet("D:/YOLO/pretrained_model/yolov3.cfg" ,
"D:/YOLO/pretrained_model/yolov3.weights")
layers = model.getLayerNames()
output_layer = [layers[layer -1] for layer in model.getUnconnectedOutLayers()]
model.setInput(img_blob)

detection_layers = model.forward(output_layer)
```

```
ids_list = []
boxes_list = []
confidences_list = []

for detection_layer in detection_layers:
    for object_detection in detection_layer:
        scores = object_detection[5:]
        predicted_id = np.argmax(scores)
        confidence = scores[predicted_id]

        if confidence > 0.10:

            label = labels[predicted_id]
            bounding_box = object_detection[0:4] * np.array([img_width, img_height,
img_width, img_height])
            (box_center_x, box_center_y, box_width, box_height) =
bounding_box.astype("int")

            start_x = int(box_center_x - (box_width / 2))
            start_y = int(box_center_y - (box_height / 2))
            ids_list.append(predicted_id)
            confidences_list.append(float(confidence))
            boxes_list.append([start_x, start_y, int(box_width), int(box_height)])

max_ids = cv2.dnn.NMSBoxes(boxes_list, confidences_list, 0.1, 0.1)
for max_id in max_ids:

    max_class_id = max_id
    box = boxes_list[max_class_id]
    start_x = box[0]
    start_y = box[1]
    box_width = box[2]
    box_height = box[3]
```

```
    predicted_id = ids_list[max_class_id]
    label = labels[predicted_id]
    confidence = confidences_list[max_class_id]
    end_x = start_x + box_width
    end_y = start_y + box_height
    box_color = colors[predicted_id]
    box_color = [int(each) for each in box_color]
    label = "{}: {:.2f}%".format(label, confidence * 100)
    print("predicted object {}".format(label))

    cv2.rectangle(img, (start_x, start_y), (end_x, end_y), box_color, 1)
    cv2.putText(img, label, (start_x, start_y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5,
box_color, 1)
cv2.imshow("Detection Window", img)
```

ÖZGEÇMİŞ

Ad-Soyad **:** Ahmet Yaşar BALKESEN

ÖĞRENİM DURUMU:

- **Lisans** **:** 2008, İnönü Üniversitesi, Fen Edebiyat Fakültesi, Fizik
- **Tezsiz Yüksek Lisans** **:** 2010, Harran Üniversitesi, Fen Bilimleri Enstitüsü, Fen ve Matematik Eğitimi Fizik Öğretmenliği

MESLEKİ DENEYİM:

- 2008 - 2011 Mef dershanesinde çalıştı.
- 2011 – 2013 Birey dershanesinde çalıştı.
- 2013 – 2020 Özel şanlıurfa saraç ilgi okullarında çalıştı.
- 2020 – 2021 Özel şanlıurfa mem kolejinde çalıştı.
- 2021 – …… Karakörü ismail nazif bayraktar m.t.a.l. çalışmakta.