T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

BSM 498 BİTİRME ÇALIŞMASI

RAYLI SİSTEMLERDE HAYATİ TEHLİKELERİN YAPAY ZEKA İLE ÖNGÖRÜLMESİ

G171210044-Nisanur Karatepe
G171210095-Zeynep Sena Nur Tekin

Fakülte Anabilim Dalı	**:**	**BİLGİSAYAR MÜHENDİSLİĞİ**
Tez Danışmanı	**:**	**Prof. Dr. Cemil Öz**

2020-2021 Güz Dönemi

T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

RAYLI SİSTEMLERDE HAYATİ TEHLİKELERİN YAPAY ZEKA İLE ÖNGÖRÜLMESİ

BSM 498 - BİTİRME ÇALIŞMASI

Nisanur KARATEPE

Zeynep Sena Nur TEKİN

Fakülte Anabilim Dalı : BİLGİSAYAR MÜHENDİSLİĞİ

Bu tez .. / .. / … tarihinde aşağıdaki jüri tarafından oybirliği / oyçokluğu ile kabul edilmiştir.

…….	…….	…….
Jüri Başkanı	**Üye**	**Üye**

ÖNSÖZ

Günümüzde birçok işletme rekabet üstünlüğünü elde etmede Bulut Bilişim teknolojisinin önemini anlamış ve gerek tedarikçileri gerekse müşterileriyle olan ilişkilerini karşılıklı işbirliği ve menfaat esasına bağlı olarak yeniden yapılandırmaya başlamışlardır. Özellikle tedarikçilerle geliştirilen teknolojik işbirliğinin veri hızının arttırılması, iletişim maliyetinin düşürülmesi, veri güvenliğinin sağlanması ve müşteri memnuniyetinin arttırılması gibi konularda son derece olumlu katkılar sağladığı görülmektedir.

İÇİNDEKİLER

ÖZET

Anahtar kelimeler: Görüntü İşleme, Raylı Sistemler, Yolcu Güvenliği

Günümüzde insan nüfusunun artması ulaşımda yoğunluğun artmasına neden olmuştur. Ulaşımın yoğunluğunun artması ise raylı sistemlerde insan güvenliğini tehdit edebilecek durumlara yol açmaktadır. Tehdit durumlarında güvenlik görevlileri tarafından bu durum algılanıp acil durum geçilmektedir. Bu durum ise zaman alabilmekte veya tehdit durumu fark edilene kadar kazalar oluşabilmektedir. İstasyonlarda bulunan kamera sistemleri ray ve güvenli çizgileri görüntülenmektedir. Bu çizgi ve ray sisteminin algılanıp güvenlik durumlarını tehdit eden durumlarda en yakın metro veya trene acil durum bilgisi gönderen otomatik bir sistem yapılması hedeflenmiştir. Bu hedef doğrultusunda öncelikle görüntü işleme ve makine öğrenmesi metotları kullanarak çizgi ihlali yapan veya raylı sistem üzerinde bulunan canlılar algılanacaktır. Tren ve metrolarda bulunan tren kontrol sistemine(Train Control & Management System,TCMS) belirli protokoller kullanılarak en yakında bulunan trene ve güvenlik görevlisine acil durum bilgisi gönderilecektir. Ayrıca kamera ile kişi yoğunluğu algılanması gibi durumlarda yapılarak nesnelerin interneti ile veri tabanında saklayacaktır. Böylece özellikle pandemi dönemi gibi dönemlerde bu sistem akıllı ulaşımı sağlayacaktır.

BÖLÜM 1. GİRİŞ

Yolcu güvenliği, demiryollarının varlığından beri her zaman çok önemli bir konu olmuştur. Demiryolu, kullanışlı ve verimli bir toplu taşıma yöntemidir fakat platformlara düşen yolcular, nesneler veya tren kapısına sıkışan yolcular gibi durumlar sıklıkla izlenebilmektedir. Bu sorunları çözmek için çeşitli uygulamalar hayata geçirilmiştir. CCTV, acil durum butonları bunlara örnektir. Ancak CCTV vb. uygulamalar sürekli insan kontrolünde izlendikleri için her zaman yeterli desteği sağlayamazlar ve bu yüzden acil durumlarda müdahale etmek zorlaşır. Gelecek trene bunu haber vermekte zaman alır. Bu yüzden bu proje demiryolu platformunda yolcu güvenliği için vizyona dayalı bir nesne algılama algoritması önermektedir.

1.1 RAYLI SİSTEMLERİN ÖNEMİ

Tüm dünyanın vazgeçemediği ulaşım şekli olan raylı sistemler gerek yolcu gerekse yük bakımında oldukça ehemmiyetli bir ulaşım şeklidir. Dünyanın nüfusunun artmasıyla kalabalık şehirlerde raylı sistemler büyük bir hızda çoğalmaktadır. Trafik yükünü büyük oranda azaltan raylı sistemler gittikçe gelişmeye başlamıştır. Büyük şehirlerde özellikle çok katlı metro ağları trafiği büyük oranda azaltıcı etkiye sahiptir. Raylı sistemlerin trafiği azaltıcı yönünün dışında güvenli olması da büyük bir artıdır. Karayolu ulaşımında trafiğin içinde olmak yolcular için tehlike arz edebilirken raylı sistemlerde bu oran çok daha az olmaktadır. İnsanlar kendilerini daha güvende hissetmektedir.

Raylı sistemler sadece yolcu taşımanın dışında yük taşınımı da sağlar. Hatta sadece yük taşıyan hatlar bile bulunmaktadır. Dünyada raylı sistemlerin tarihçesi epey eskiye dayanmaktadır. 1800'lü yılların başlarına dayanan sistem hala önemini korumaktadır.

Günümüzde kent içi toplu taşımacılıkta kullanılan raylı ulaşım sistemleri; hafif raylı sistemler, tramvaylar, metrolar, banliyö trenleri, manyetik yataklı sistemler ve üst yollu elektrikli, toplu taşıma sistemi olan monoraydan oluşmaktadır. Bu sistemlerden ilk dördü ülkemizin değişik kentlerinde etkin olarak kullanılmaktadır.[1]

1.1.2 Raylı sistemlerde yolcu güvenliğinin önemi

Ulaşımda her şeyden önce güvenlik olmak zorundadır. Güvenli olmayan ulaşım türleri varlığını devam ettiremez. Raylı sistemler yolculara diğer ulaşım şekillerinden kat kat daha fazla güvenlik sağlamaktadır.

İstasyonların turnike alanlarında, nokta nöbeti tutan resmi üniformalı güvenlik görevlileri; yolcuya açık diğer alanlarda ise devriye hizmeti veren güvenlik görevlileri bulunmaktadır. Yanıcı, parlayıcı, patlayıcı vb. maddelerle istasyon ve araçlara giriş yapılması yasak olduğundan güvenlik personelleri, detektör ile üst araması, çanta ve paket kontrolü yapmaktadır. İstasyonlarda resmi üniformalı güvenlik görevlilerinin yanı sıra, sivil olarak görev yapan güvenlik görevlileri de mevcuttur. İstasyonlarda güvenlik personeli dışında zaman zaman da resmi polis memurları bulundurmakta ve koordineli bir şekilde çalışılmaktadır.

Acil durumlarda, güvenlik görevlilerinin yanı sıra istasyonlarda bulunan diğer personeller de (istasyon amirleri, bakım ekipleri, temizlik personelleri vb.) bulundukları alanları kontrol etmekle sorumludurlar. Olası bir acil durumda söndürme, tahliye vb. işlemlerini yapacak personel ve görev dağılımı belirlenmiştir. Konularına göre yapılan tatbikat ve eğitimler de belirli periyotlarla tekrarlanmaktadır.

İstasyonlardaki halka açık/kapalı tüm alanlar, ileri teknoloji ürünü kameralar ile 24 saat ve gerçek zamanlı olarak izlenmektedir. Hem kumanda merkezi hem de istasyonlardan lokal olarak izleme yapılan kameralar bulunmaktadır.

1.2 DÜNYA GENELİNDE RAYLI SİSTEMLERİN KULLANILIŞI

Dünya genelinde raylı sistemlerin kullanımı ülkelerin gelişmişliği ve nüfus yoğunluğuyla doğru orantılıdır. Her açıdan faydalı olması hasebiyle ülkeler raylı sistemleri giderek daha fazla kullanmaktadır.

Dünyada demiryolu uzunluğu en uzun ülke Amerika'dır.2006 verilerine göre 226.706 km uzunluğa sahip demiryolu ağı bulunmaktadır. Türkiye'de bu uzunluk 2015 verilerine göre 8.699 km'dir.

1.3 Raylı Sistemlerde İntihar

İntihar vakalarınının gerçekleştirildiği alanların içinde çok büyük yüzdeye sahip olmasa da raylı sistemler de bulunmaktadır. Yaşamına son vermek isteyen birey gelen trenin önüne kendini atmaktadır. Geçici birkaç çözüm bulunsa da kalıcı çözüm hala bulunamamıştır.

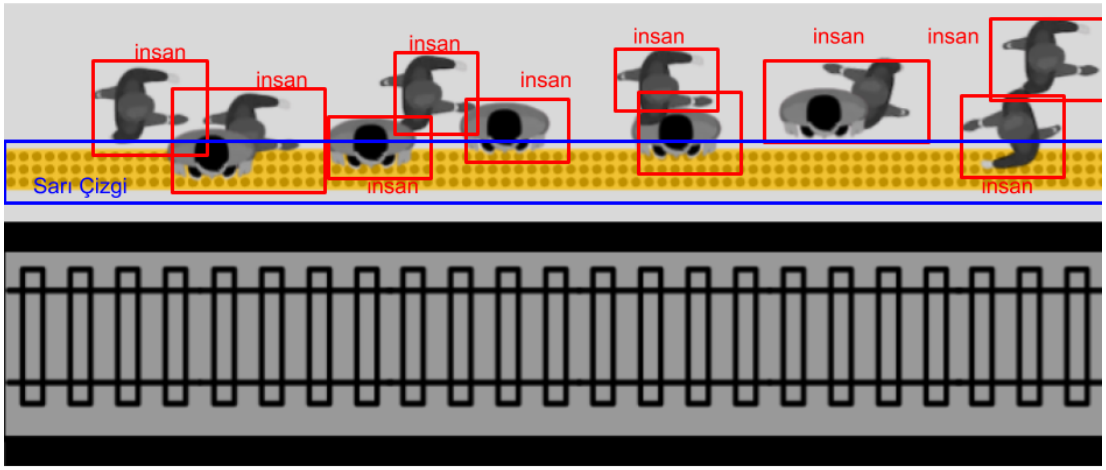
İntiharların yanı sıra raylarda herhangi başka canlının bulunması durumunda da kötü sonuçlar oluşabilmektedir. Örneğin durumdan habersiz bir köpek trenin altında kalabilmektedir. Bir canlının yaşamından daha önemli hiçbir şey bulunmayan dünyada bu tehlikelerin önlenmesi en öncelikli yapılması gereken iştir.

BÖLÜM 2. SİSTEMATİK YAKLAŞIM

2.1. Görüntü İşleme (Image Processing) Nedir?

Görüntü işlemenin nasıl gerçekleştiğini anlatmadan önce görüntü işlemenin tam olarak ne olduğunu ve büyük resimdeki rolünü bilmek önemlidir. Görüntü işlemenin sıklıkla kullanıldığı alan 'Bilgisayar Görmesi' dir. Bu iki teknoloji arasındaki benzerlikler ve farklar ise şunlardır, hem görüntü işleme hem de bilgisayar görmesi(Computer vision) algoritmaları girdi olarak bir görüntü alır; bununla birlikte görüntü işlemede çıktı aynı zamanda bir görüntüdür oysa bilgisayar görmesinde çıktı görüntü hakkında bazı özellikler/bilgiler olabilir.

Topladığımız veya ürettiğimiz veriler çoğunlukla ham verilerdir, yani bir dizi olası nedenden dolayı doğrudan uygulamalarda kullanılmaya uygun değildir. Bu nedenle önce onu analiz etmemiz, gerekli ön işlemeyi yapmamız ve sonra kullanmamız gerekiyor. Örneğin, bir kedi sınıflandırıcı oluşturmaya çalıştığımızı varsayalım. Programımız girdi olarak bir görüntüyü alır ve ardından görüntünün kedi içerip içermediğini söyler. Bu sınıflandırıcıyı oluşturmanın ilk adımı, yüzlerce kedi resmini toplamak olacaktır. Yaygın bir sorun, kazıdığımız tüm resimlerin aynı boyutta / boyutlarda olmamasıdır, bu nedenle onları eğitim için modele beslemeden önce hepsini standart bir boyuta yeniden boyutlandırmamız / önceden işlememiz gerekir. Bu, görüntü işlemenin herhangi bir bilgisayarla görme uygulaması için gerekli olmasının birçok nedeninden sadece biridir.

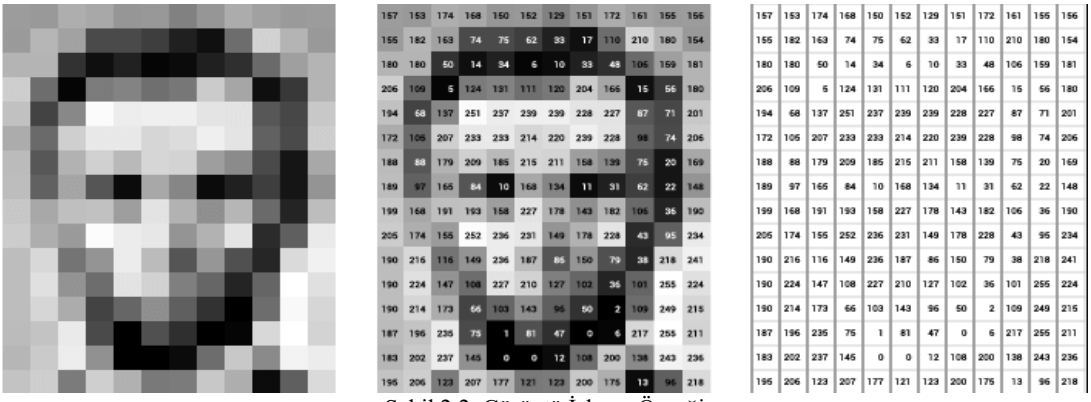


Şekil 2.1. Görüntü İşleme Örneği

Görüntü işleme, bir görüntüyü elde etmek ya da elimizde bulunan görüntüden yararlı bilgiler çıkarmak için çeşitli algoritmalar kullanarak görüntü üzerinde bazı işlemlerin gerçekleştirilme yöntemidir. Görüntü işlemenin temel aşamaları şunlardır:

1. Görüntünün çalışma ortamına aktarılması
2. Algılanan görüntünün işlenerek analiz edilmesi ve görüntülenmesi
3. Analiz sonucu görüntü raporunun ortaya çıkması

Görüntü işleme aslında matrisler üzerinde yapılan işlemler bütünü denilebilir. Genellikle resimlere baktığımızda çeşitli renklerin bir araya geldiği kareden ibaret olduğunu düşünülebilir. Halbuki resmi en küçük parçalara bölündüğünde çok boyutlu matrislerle karşılaşılmaktadır. Bu matrislerin her bir elemanına olan piksel adını verilmektedir. Ve görüntü işleme yöntemlerinde pikseli oluşturan matris hücrelerin üzerinden işlemler yapılır. Aşağıdaki görsel bu paragrafın görsel halidir.



Şekil 2.2. Görüntü İşleme Örneği

2.2. Görüntü İşlemenin Amacı Nedir?

Her teknolojinin olduğu gibi görüntü işlemenin amacı da insanların hayatını kolaylaştırmak, iyileştirmektir. Ürün sayma ve hata tespiti, uydu görüntüleri üzerinde nüfus yoğunluğu, çevre kirliliği tespiti, kalite ve yüzey analizi, askeri endüstri, robot kol yönetimi, eksik ve hatalı parça veya üretim kontrolü, robotik, trafik, uzaktan algılama yöntemleri, uydu görüntülerinden hava tahmini gibi konular görüntü işlemenin temel kullanım alanlarıdır. Bitirme tasarımında da görüntü işlemeden yararlanılmıştır.

2.2.1. Görselleştirme: görünmesi zor nesneleri gözlemleme

Özellikle tıp ve benzeri alanlarda insan gözünün görmesinin mümkün olmaması gibi durumlarda kullanılır.

2.2.2. Görüntü keskinleştirme ve restorasyon: gürültülü görüntüleri iyileştirme

2 farklı yöntem ile yapılabilir. İlk yöntem ‘Kenar görüntüsü kullanarak netleştirme’ dir. Bu algoritma orjinal görüntüden, görüntünün yumuşatılmış halini çıkararak belirgin kenarların görüntüsünü ortaya çıkarır.

Daha sonra orjinal görüntü ile belirginleşmiş kenarların görüntüsünü birleştirerek, kenarları keskinleşmiş görüntüyü (netleşmiş görüntü) elde eder. Görüntünün netleştirilmesi fotogrametri ve baskı endüstrisinde yaygın olarak kullanılır.

Şekil 2.3. Kenar Görüntüsü Kullanarak Netleştirme Örneği

İkinci kullanılabilecek olan algoritma ise ‘Konvolüsyon Yöntemi’(çekirdek matris ile netleştirme). Çekirdek matris kullanarak da netleştirme yapılabilir. Bu matriste temel mantık üzerinde işlem yapılan pikselin kenarlarındaki 4 tane piksele bakar (Köşelere bakmıyor, sıfırla çarpıyor, onları toplama katmıyor) bu piksellerin değerini aşağıya indirir, üzerinde bulunduğu pikselin değerini yukarı çıkarır. Eğer her tarafı aynı olan bir bölgede ise sonuç değişmez. Fakat bir sınır bölgesine gelirse üzerine bastığın pikselin değerini yükseltir. Böylece sınır olan yerler daha parlak gözükür. Bu anlatılanları kendiniz örnek piksel değerleri deneyin. Matris değerleri ile piksel değerleri çarpılıp toplandıktan sonra matris toplamına da bölmek gerekir (yani buradaki 1/3 sayısı bunu anlatıyor).

$$\begin{bmatrix} 0 & -2 & 0 \\ -2 & 11 & -2 \\ 0 & -2 & 0 \end{bmatrix} \times \frac{1}{3}$$

Şekil 2.4. Çekirdek Matris Örneği

Şekil 2.5. Kenar Görüntüsü Kullanarak Netleştirme Örneği

2.2.3. Desen tanıma: bir görüntüdeki çeşitli nesneleri tanıma

Son yıllarda nesne tanıma ve tespiti için R-CNN, Fast R-CNN, Faster R-CNN, Mask R-CNN ve SSD kütüphaneleri yaygın olarak kullanılmaktadır. R-CNN ilgili kütüphanenin bölge önerisi yaklaşımını kullanan en temel modelidir. Öte yandan, R-CNN kütüphanesinin Fast R-CNN, Faster R-CNN ve Mask R-CNN gibi geliştirilmiş

modelleri de bulunmaktadır. Bu modeller içerisinde genel başarımı itibariyle en iyi model Faster RCNN'dir. Farklı boyutlarda bölge önerisi yapan modelde ilgili pencereler konvansiyonel sinir ağlarından geçirilerek pencerelerin boyutları eşitlenmektedir. Sinir ağının sonucunda o bölgede sınıflandırma yapmak için Destek Vektör Makineleri (SVM) sınıflandırıcısı kullanılmaktadır. Bu gözetimli öğrenmede kullanılan bir makine öğretimi yöntemidir.

2.2.4. Görüntü tanıma: bir görüntüdeki nesneleri ayırt etme

YOLO (You Only Look Once), gerçek zamanlı nesne tespitini gerçekleştiren algoritmadır. YOLO algoritması hızlı bir algoritmadır bunun olmasının sebebi elindeki görüntüyü tek bir seferde nöral ağdan geçirerek resimdeki tüm nesnelerin sınıfını ve koordinatlarını tahmin edebilmesidir. Yani bu tahmin işleminin temeli, nesne tespitini tek bir regresyon problemi olarak ele almasında yatmaktadır. Algoritma ilk önce input görüntüsünü SxS'lik ızgaralara bölüyor. Bu ızgaralar 5x5, 9x9 veya 21x21'lik kutucuklara bölünmüş olabilir.Bu görüntüdeki her bir ızgara kendi içerisinde, bölgede nesnenin var olup olmadığını eğer varsa orta noktasının içinde olup olmadığını, orta noktası da içerisinde bulunuyor ise uzunluğunu, yüksekliğini ve hangi sınıftan olduğunu bulmakla sorumludur. YOLO, her bir oluşturduğu ızgara adına bir adet tahmin vektörü oluşturmaktadır.

2.3. Görüntü İşlemede Python

Python programlama dili basit bir kod yapısına sahip olduğu için öğrenilmesi oldukça kolaydır.Diğer programlama dillerinden farklı olan bir yapısı da girintili bir şekilde kod yazılabilmesi için kodların anlamdırılması oldukça kolay olmaktadır. Geniş bir açık kaynak kütüphanesine sahip olması ve hemen hemen tüm işletim sistemleri ile uyumlu çalışması görüntü işleme için tercih sebebidir. Ayrıca Python programlama dili önemli ticari veri tabanları ile entegre bir şekilde çalışabilmektedir.

2.4. Numpy Kütüphanesi

NumPy Python'da bilimsel hesaplamalar için kullanılan bir pakettir. Çok boyutlu diziler ve çeşitli türetilmiş nesneler (matrisler gibi) üzerinde hızlı işlemler yapabilmek için kullanılır. Ayrıca matematiksel, mantıksal, sıralama, seçme, giriş/çıkış dahil olmak üzere çeşitli yordamlar sağlayan Python kütüphanesidir.

2.5. OpenCV Kütüphanesi Kullanımı

OpenCV (Open Source Computer Vision) açık kaynak kodlu görüntü işleme kütüphanesidir. Computer Vision (Bilgisayarla Görme) kelimesinin baş harfleri kullanılarak isimlendirilen CV bileşeni, temel resim işleme fonksiyonları ve Bilgisayarla Görme için kullanılan yüksek seviyeli algoritmaları bünyesinde barındıran beş temel kütüphaneden biridir.

OpenCV kütüphanesinin görüntü işleme için çok kullanışlı ve çok gerekli olan video veya resimlerdeki nesneleri bulmak ve onları takip etme konusunda en yaygın kullanılan kütüphane olarak karşımıza çıkmaktadır.

En temel manada belirli bir algoritmaya göre bulunması istenen nesneler önce bilgisayara tanıtılır ve daha sonra ona benzer şekillerin bulunduğu resimler veya video kareleri taranarak o nesne bulunmaya çalışılır. Sistemin çalıştırılması için Opencv'nin yüklenmesi gerekmektedir. Bunun için öncelikle aşağıda verilen komut ile Opencv modülü Python üzerinde kurulur

```
pip install opencv-python
```

Kurulum yapıldıktan sonra aşağıda verilen komutlar ile modüller yazılıma eklenir.

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

Görüntü işlenecek çerçeveyi(frame) için öncelikle belli dizin üzerinde aşağıda verilen komut ile görüntü matrislerden oluşan bir değişkene atılır.

```
img=cv2.imread("dizin")
```

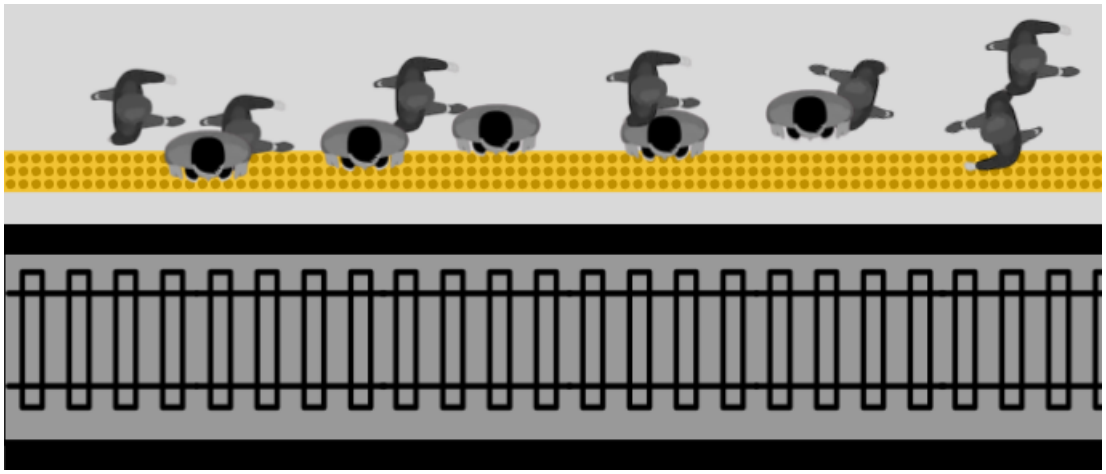
Görüntü yerine video üzerinde işlem yapmak için ise aşağıda verilen kod bloğu kullanılmaktadır. Burada ayrıca görüntü üzerinde bazı filtreleme işlemleri de yapılmıştır.

```
while True:
    ret, frame = cap.read()
    reSize = frame[350:800,300:1400]
    blur = cv2.GaussianBlur(reSize, (5,5), 0)
    hsv = cv2.cvtColor(blur, cv2.COLOR_BGR2HSV)
    low_yellow = np.array([18,94,140])
    up_yellow = np.array([48,255,255])
    low_brown = np.array([20,100,100])
    up_brown = np.array([30,255,255])
    mask_y = cv2.inRange(hsv, low_yellow, up_yellow)
    mask_r = cv2.inRange(hsv, low_brown, up_brown)
    mask = mask_r+mask_y
    edges = cv2.Canny(mask, 75, 158)
    lines = cv2.HoughLinesP(edges, 1, np.pi/180, 30, maxLineGap=50)
```

BÖLÜM 3. PROJE SİSTEMATİĞİ VE YAPAY ZEKA TASARIMI

3.1. Sarı Çizgi Kontrolü

Yolcuların güvenliği için istasyon kenarındaki sarı çizgi kontrol edilmelidir. Sarı çizginin ihlali kontrol edilirse herhangi bir canlının bekleme esnasında raylara düşme ihtimali ortadan kalkacaktır.



Şekil 3.1. İstasyon Modeli

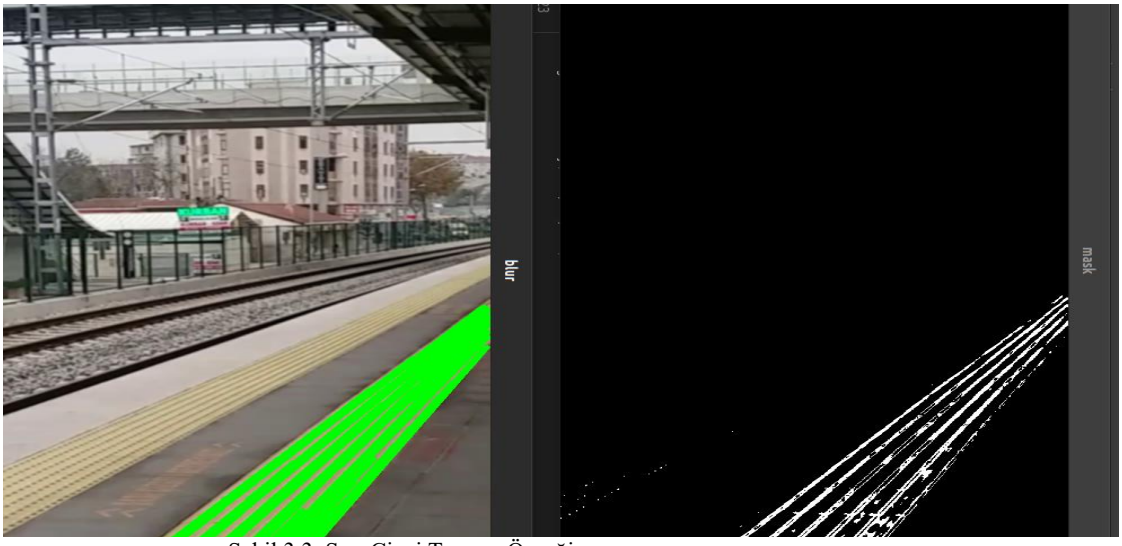
Projede sarı çizgi kontrolü için öncelikle istasyona yerleştirilen kameralardan veri alınması gerekir. 24 saat kayıt yapabilecek kameralara ihtiyaç vardır. Kameraların açısı hem sarı çizgiyi hem de rayları görebilecek şekilde olmalıdır. Bunun sebebi tren yokken sarı çizgi kontrolü yapılıp tren geldiği zaman da bu kontrolün ortadan kalkma durumudur.

Olası sarı çizgi ihlallerinde sistem otomatik olarak yetkili birimlere haber verecektir. Aynı zamanda istasyonda da sesli ve ışıklı uyarılar olacaktır. Hem istasyondaki kişileri uyarmak hem de yetkililere haber vermek tehlike ihtimallerini büyük oranda azaltacaktır. Sarı çizgiyi görüntüde algılamak için öncelikle cv2 ve numpy kütüphaneleri python koduna Şekil 3.2 ile verildiği eklenmelidir.

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

Şekil 3.2. Kütüphanenin Eklenmesi

Kütüphaneler eklendikten sonra görüntülerin kameradan alınacağı sisteme yazılmalıdır. Kamera görüntüleri alındıktan sonra sarı çizginin sistem tarafından algılanabilmesi için sisteme sarı çizginin en alt ve en üst HSV renk kodu girilmiştir. Bu sayede program çalıştırıldığında görüntüdeki bütün sarı rengi algılayabilecektir. Sarı rengi algılamak sistem için tamamen yeterli olmayacaktır bu yüzden sarı renginin çizgi şeklinde algılanması gerekir. Çizgi şeklinde algılandığında görüntüde sadece sarı çizgi algılaması yapılabilmektedir.



Şekil 3.3. Sarı Çizgi Tanıma Örneği

Yukarıda program çalıştığında sarı çizgi algılamanın nasıl gerçekleştiği gösterilmiştir. Trenin istasyonda durmadığı her an bu çizginin bütünlüğü bozulmayacak şekilde program tasarlanmıştır. Çizgi bütünlüğü bozulduğunda olağandışı bir durum olması dolayısıyla uyarı verilmektedir.

3.2. Tren ve Ray Kontrolü

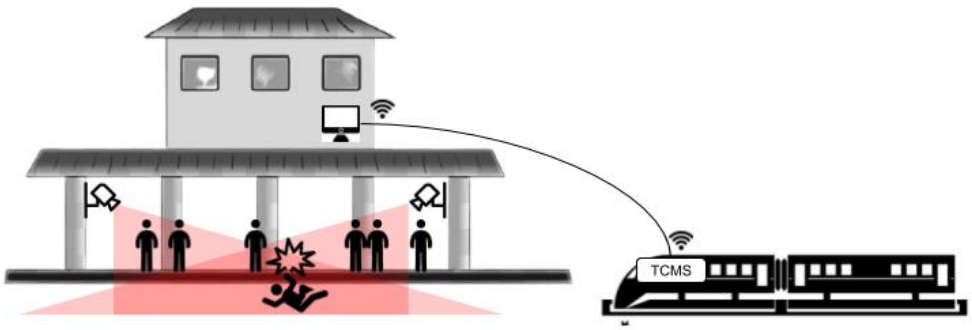
Sarı çizgi kontrolü sistemin çalışması için tek başına yeterli değildir. İstasyonda kameralar çift yönlü takılacağından dolayı bazı kameralarda raylar sarı çizginin

solunda olmasına rağmen bazı kameralarda sağında olacaktır. Bu durumu önlemenin en kolay yolu sarı çizgiden sonra rayları algılamaktır. Raylar algılandıktan sonra rayların olduğu kısım tehlikeli bölge olacak ve sarı çizgiden raylara doğru olan kısım geçilmesi yasak bölge olacaktır. Sistem bu şekilde çalıştığında olası kamera karmaşası önlenecektir.

Rayların algılanması sarı çizginin algılanmasıyla aynı mantıkta gerçekleşir. Önce rayların renk kodları programa girilmiştir. Renk kodlarının algılanması tek başına yeterli olmayacağından dolayı çizgi şeklinde algılanabilmesi için de çizgi kodları girilmiştir. Bu sayede çizgi şeklinde rayların bütünlüğü anlaşılabilmektedir.

Tren kontrolü de aynı şekilde sarı çizgi ihlali gibi olacaktır. Tren geldiğinde görüntüde rayların çizgi bütünlüğü bozulacaktır. Görüntüden rayların çizgisi tamamen çıktığında sarı çizgi ihlal kontrolü devre dışı olacak şekilde tasarlanmıştır. Sistemi daha da kuvvetlendirmek adına bir kontrol bloğu yerleştirilmiştir. Trenin gelişi harici durumlarda raylara istasyondan düşüldüğünde sistemi devre dışı bırakmamak adına rayların çizgi bütünlüğü bozulduğunda sistemi devre dışı bırakmadan önce sarı çizgi ihlali kontrol edilmektedir.

3.3. Raylara Herhangi Bir Canlı Düşmesi Durumu



Şekil 3.4. Raylarda Tehlike Algılanması

Tren istasyonda değilken raylara canlı düşmesi durumunda sistem öncelikle sarı çizgi ihlalinden, daha sonra da rayların çizgi bütünlüğünün bozulmasından durumu anlayabilmektedir. Bu iki çizgi bütünlüğünün herhangi birinin bozulması durumunda uyarı sinyalleri oluşmaktadır. Ray çizgi bütünlüğünün bozulması durumunda sarı çizgi uyarılarının yanı sıra yolda gelmekte olan trene wifi TCMS ile bilgi aktarımı

yapılmaktadır. Tren mesafesini ona göre ayarlayabilmekte ve olası tehlike minimuma indirilebilmektedir. Oluşturulan sistem sadece istasyonları kapsamaktadır. Raylı sistemlerde istasyon dışında tüm yola güvenlik kamerası konulabilmesi durumunda tüm yol izlenebilir. Özellikle yerin altında olmayan yollarda hayvan güvenliği için çok büyük öneme sahiptir. Rayların üstündeki canlılar tespit edilirse yolda olan trene haber verilir. Tren mesafesine göre hızını ayarlayabilir.

BÖLÜM 4. SİSTEMİN HABERLEŞMESİ

Herhangi tehlike anında sistemin en yakın trene ve istasyonun içine uyarı sinyali göndermesi gereklidir. Bu kapsamda MQTT protokolü ile internet üzerinden bu haberleşme sinyali gönderilmektedir. Burada MQTT protokolünde yalnızca sinyal gönderilmiş ve gönderilen sinyalin geldiği teyit edilmiştir. Sinyalin geldiğini gözlemlemek için bitirme kapsamında ayrıca bir mobil uygulama geliştirilecektir.

4.1 MQTT Haberleşmesi

MQTT yayıncı(publisher) ve abone(subscriber) ilişkisine dayanan bir haberleşme protokolüdür. Burada tüm mesajların yönetildiği bir aracı(broker) bulunmaktadır. Aracı yayıncılardan gelen mesajları abonelere iletmeyi sağlamaktadır ve böylece mesajlaşma trafiğini yönetmektedir. MQTT nesnelerin internetinin(IoT) ve makineden makineye(M2M) yaygınlaşmasıyla özellikle sunucu temelli uygulamalarda çok popüler olarak kullanılmaktadır. Burada yayıncı belli bir anahtar kelime üzerinden yayın yapmaktadır. Abone ise belli anahtar kelimeleri dinleyerek mesajları almaktadır. Böylece asenkron olarak kullanılan sistemlerde oldukça sık olarak kullanılmaktadır.

4.2 Tren İstasyon Haberleşmesinde MQTT Kullanımı

MQTT kullanımı ile haberleşme için öncelikle sisteme gerekli kütüphaneleri entegre etmek gereklidir. Python yazılım dilinde paho-mqtt adlı kütüphane MQTT ile ilgili sistemin ihtiyaçlarını karşılamaktadır. Geliştirilen sistemde MQTT haberleşmesi için ayrı bir sınıf oluşturulmuştur. Burada oluşturulan sınıf bir üst katman yazılımı olup sistemin daha efektif olarak kullanılmasını sağlamaktadır. MQTT kullanılarak oluşturulan sınıf tablo ile verilmiştir.

Tablo 4.1. MQTT üst katman yazılımı

```
import paho.mqtt.client as paho_mqtt

class mqtt:
    __sub_topic = []
```

```
__client=paho_mqtt.Client()
__message=None

def __init__(self,broker="localhost",port=1883,keepalive=120):
    self.host=broker
    self.port=port
    self.keepalive=keepalive
    self.__client.on_connect=self.__on_connect
    self.__client.on_message=self.__on_message
    self.__client.on_publish=self.__on_publish
 def add_topic(self,topic_name,qos=0):
    self.__sub_topic.append((topic_name,qos))
 def print_subtopics(self):
    print(self.__sub_topic)
 def connect(self,username="",password=""):
    self.username=username
    self.password=password
    self.__client.username_pw_set(self.username, password)
    self.__client.connect(self.host, self.port, self.keepalive)
    self.__client.subscribe(self.__sub_topic)
    self.__client.loop_start()
 def __on_connect(self,client, userdata, layoutFlags, rc):
    if rc == 0:
        print("Connected to Broker")
    else:
        print("Broker Connection is failed ")
 def publish(self,topic,message):
    self.__client.publish(topic,message)
def __on_message(self,client, userdata, message):
    self.__message=message
def __on_publish(self,client, userdata, mid):
    print("Mesaj gönderildi")
def addsubscribefunc(self,function):
    self.__client.on_message=function
def get_message(self):
    if self.__message == None:
        return
    else:
        payload=self.__message
        self.__message=None
        return payload
```

Burada oluşturulan sınıf ile MQTT bağlantıların açılması ve mesajların iletilmesi çok daha hızlı koda entegre edilebilmektedir. Ana kod üzerinde bağlantı açmak için aracının adresi ve portu girilmesi gerekmektedir. Burada aracı olarak internet üzerinde

ücretsiz olarak kullanılabilen “broker.emqx.io” adresi ve 1883 portu kullanılarak bağlantı açılmıştır. Bu işlemler yapıldıktan sonra tren gelmesi durumunda ve hayati tehlikenin oluşması durumdan istasyon numarası JSON olarak broker üzerine gönderilmiştir. Böylece tren veya istasyon görevlileri bu bilgiyi kullanarak anlık müdahale yapabilir veya tren otomatik frenleme yaparak treni yavaşlatabilir.

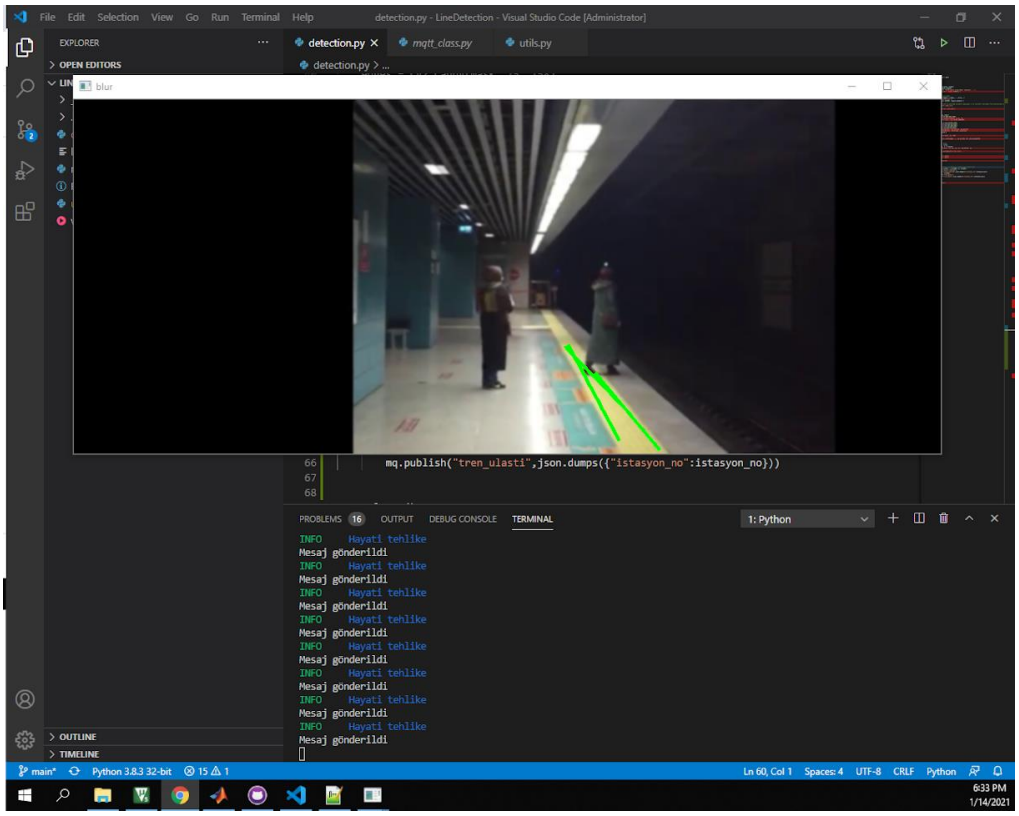
BÖLÜM 5. DENEYSEL SONUÇLAR

5.1. Yapay Zekada Hata Payı

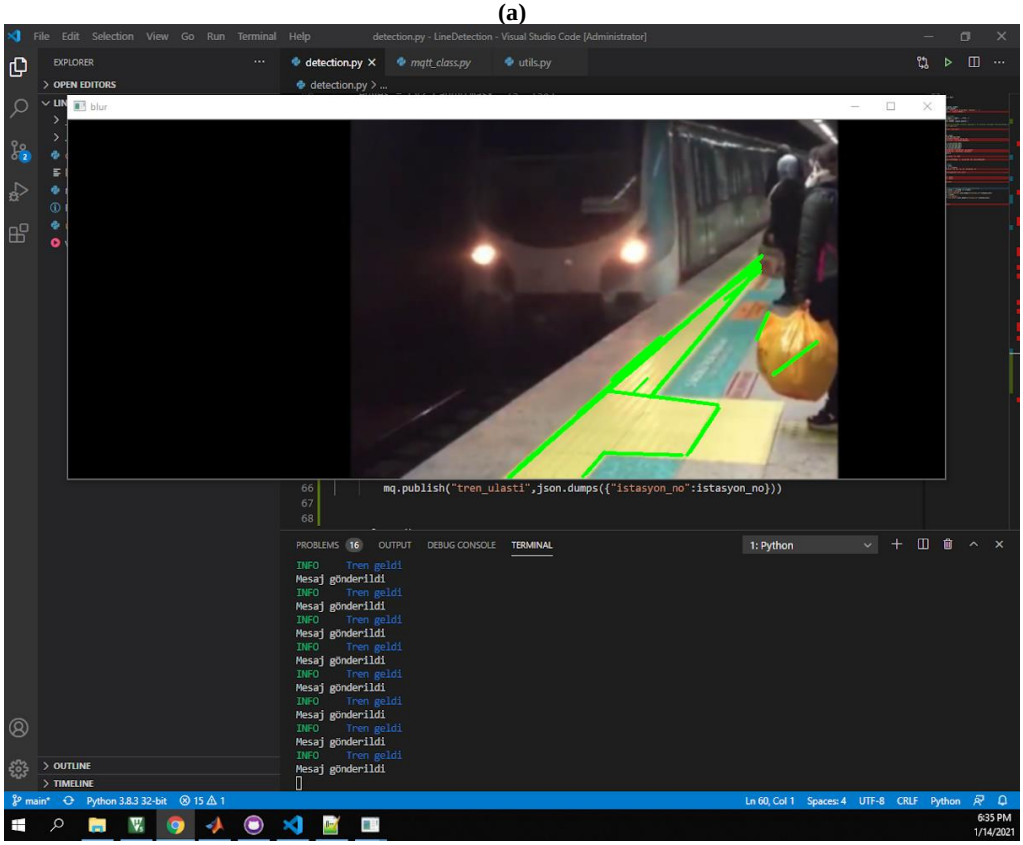
Yapay zeka uygulamalarında daha önce bulunan sonuçlardan bazı örnekler şu şekildedir. Mustafa Kısa, Karayollarında Seyreden Araçların Tanınması [7] adlı tez çalışmasında hata oranını tahminlerdeki bağıl hata %1.86-4.90 arasında değişmekte olup kabul edilebilir düzeyde bulunmuştur. Başka bir tez çalışması olan Ambalaj Yüzeyindeki Hataların Görüntü İşleme Tekniği ile Tespiti [8]'inde yapılan deney sonucu operatörlerin hata tespiti başarısı %78,58 bulunmuştur. Operatörler sırasıyla boyadan %85.88, ayarsızlıktan %77.89, bıçak çiziğinden %70, parlamadan %78 başarı sağlamaktadır. Bu çalışmada yapılan Görüntü İşleme yöntemiyle yapılan hata tespiti başarısı ise %90,58'dir. İnsansız Kara Araçları İçin Dinamik Nesnelerin Tanınması Amacıyla Görüntü İşleme Tabanlı Bir Sistem Geliştirilmesi [9] adlı doktora tezinde başarı oranı %85 bulunarak yaya ve otomobiller için en uzak tespit mesafeleri bulunmuştur. Stuart J. Russell 2018 yılında verdiği bir röportajında [10] yapay zekanın hata oranı hakkında şunları söylemiştir:"2010'da en iyi programlarını yüzde 30'luk hata payı oluyordu. 2015'te insanın yaptığı hata oranı yakalandı. 2017'de yüzde 2'ye düştü."

5.2. DENEYSEL SONUÇLAR

Sistemin, güvenilirliğini ve hata payını ölçmek için Gebze-Halkalı Marmaray hattı arasında farklı zamanlarda yapılan çekimlerle test edilmiştir. Test kapsamında video kaydı alınıp program üzerinde çerçevelere ayırarak hayati tehlike ve trenlerin algılanması saptanmıştır. Yapılan işlemler sonucunda alınan ekran görüntüleri Şkeil 5.1 ile verilmiştir. Burada görüldüğü üzere sarı çizginin geçilmesi ve trenin gelişi algılanıp uyarı sinyali ekranda gösterilmiş ve aracı üzerine de gönderilmiştir.



(a)



(b)

Şekil 5.1. Marmaray hattı (a) hayat tehlike (b) tren testleri

BÖLÜM 6. SONUÇLAR VE ÖNERİLER

Otonom sistemler, insanların hayatının kolaylaştırmasının yanında aynı zamanda güvenlik problemlerini de içinde barındırmaktadır. Bu nedenle özellikle raylı sistemlerde ray hatlarında güvenlik bariyerinin bulunmaması çocuk ve yaşlı kesim için tehdit oluşturmaktadır. Bu tehditin ortadan kaldırılması için güvenlik personelleri olmasına rağmen çok sık kazalar yaşanmaktadır. Bu nedenle, oluşturulan tasarım kapsamında ray hatları ve sarı çizgiler algılanarak hayati tehlike içeren durumlarda uyarı sinyallerinin tren ve istasyon görevlilerine gönderilmesi amaçlanmıştır. Bu amaç doğrultusunda görüntü işleme ve haberleşme teknolojileri kullanılarak bu durumlar belirli bir oranda algılanmıştır. Burada uyarı sinyalleri MQTT protokolü üzerinde bir aracıya gönderilerek asenkron olarak birden fazla istasyonda çalışabilecek gerçek zamanlı bir sistem tasarlanmıştır. Sistemin uyarı sinyallerini aracıya abone olarak alınabildiği için bu kısım ve sistemin daha efektif hale getirilmesi bitirme kapsamında tasarlanacaktır. Ayrıca sistemin daha efektif olması için trenin içinde bulunan TCMS'e bilgi gönderebilmesi gereklidir. Bunun için ise radyo frekans üzerinden de sinyal çıkışı alınması sistemin daha efektif bir şekilde kullanılabileceğini öngörmektedir.

KAYNAKLAR

[1] İrfan Şenlik, KENT İÇİ RAYLI ULAŞIM SİSTEMLERİTSOURIS, Elektrik Mühendisliği Dergisi

[2] Mehmet ÖZDEMİR,Görüntü Keskinleştirme Yöntemlerinin Nesne-Yönelimli Sınıflandırma Açısından Değerlendirilmesi, Harita Dergisi Temmuz 2017 Sayı 158

[3] UYDU GÖRÜNTÜLERİ VE SAYISAL UZAKTAN ALGILAMA, https://www.ktu.edu.tr/dosyalar/ormanamenajmani_0d954.pdf

[4] Resul DAŞ, Berna POLAT, Gürkan TUNA, Derin Öğrenme ile Resim ve Videolarda Nesnelerin Tanınması ve Takibi, Fırat Üniversitesi Müh. Bil. Dergisi, 31(2), 571-581, 2019

[5] Mehmet KARAKOÇ, GÖRÜNTÜ ÐSLEME TEKNÐKLERÐ VE YAPAY ZEKÂ YÖNTEMLERI KULLANARAK GÖRÜNTÜ ÐÇÐNDE GÖRÜNTÜ ARAMA, PAMUKKALE ÜNÐVERSÐTESÐ FEN BÐLÐMLERÐ ENSTÐTÜSÜ

[6] MESUT PİŞKİN, OPENCV İLE GÖRÜNTÜ İŞLEME

[7] Mustafa Kısa, KARAYOLUNDA SEYREDEN ARAÇLARIN TANINMASI, SELÇUK ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ KARAYOLUNDA

[8] https://dergipark.org.tr/tr/download/article-file/841561

[9] Güray SONUGÜR, İNSANSIZ KARA ARAÇLARI İÇİN DİNAMİK NESNELERİN TANINMASI AMACIYLA GÖRÜNTÜ İŞLEME TABANLI BİR SİSTEM GELİŞTİRİLMESİ, AFYON KOCATEPE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

[10] https://www.turkiyegazetesi.com.tr/teknoloji/554344.aspx

ÖZGEÇMİŞ

Nisanur Karatepe 26.07.1999'da İstanbul Bahçelievler'de doğdu. İlk, orta ve lise eğitimini Küçükçekmece'de tamamladı. 2017 yılında Fahreddin Kerim Gökay Anadolu Lisesi'nden mezun oldu. Aynı yıl Sakarya Üniversitesi Bilgisayar Mühendisliği Bölümünü kazandı.

Zeynep Sena Nur Tekin 04.12.1998'de Bolu'da doğdu. İlk ve orta eğitimini Almanya'da tamamladı. 2017 yılında Düzce Anadolu İmam-Hatip Lisesinden mezun oldu. Aynı sene Sakarya Üniversitesi Bilgisayar Mühendisliği Bölümünü kazandı.

BSM 498 BİTİRME ÇALIŞMASI DEĞERLENDİRME VE SÖZLÜ SINAV TUTANAĞI

KONU :
ÖĞRENCİLER (Öğrenci No/AD/SOYAD):

Değerlendirme Konusu	İstenenler	Not Aralığı	Not
Yazılı Çalışma			
Çalışma klavuza uygun olarak hazırlanmış mı?	x	0-5	
Teknik Yönden			
Problemin tanımı yapılmış mı?	x	0-5	
Geliştirilecek yazılımın/donanımın mimarisini içeren blok şeması (yazılımlar için veri akış şeması (dfd) da olabilir) çizilerek açıklanmış mı?			
Blok şemadaki birimler arasındaki bilgi akışına ait model/gösterim var mı?			
Yazılımın gereksinim listesi oluşturulmuş mu?			
Kullanılan/kullanılması düşünülen araçlar/teknolojiler anlatılmış mı?			
Donanımların programlanması/konfigürasyonu için yazılım gereksinimleri belirtilmiş mi?			
UML ile modelleme yapılmış mı?			
Veritabanları kullanılmış ise kavramsal model çıkarılmış mı? (Varlık ilişki modeli, noSQL kavramsal modelleri v.b.)			
Projeye yönelik iş-zaman çizelgesi çıkarılarak maliyet analizi yapılmış mı?			
Donanım bileşenlerinin maliyet analizi (prototip-adetli seri üretim vb.) çıkarılmış mı?			
Donanım için gerekli enerji analizi (minimum-uyku-aktif-maksimum) yapılmış mı?			
Grup çalışmalarında grup üyelerinin görev tanımları verilmiş mi (iş-zaman çizelgesinde belirtilebilir)?			
Sürüm denetim sistemi (Version Control System; Git, Subversion v.s.) kullanılmış mı?			
Sistemin genel testi için uygulanan metotlar ve iyileştirme süreçlerinin dökümü verilmiş mi?			
Yazılımın sızma testi yapılmış mı?			
Performans testi yapılmış mı?			

Tasarımın uygulamasında ortaya çıkan uyumsuzluklar ve aksaklıklar belirtilerek çözüm yöntemleri tartışılmış mı?			
Yapılan işlerin zorluk derecesi?	x	0-25	
Sözlü Sınav			
Yapılan sunum başarılı mı?	x	0-5	
Soruları yanıtlama yetkinliği?	x	0-20	
Devam Durumu			
Öğrenci dönem içerisindeki raporlarını düzenli olarak hazırladı mı?	x	0-5	
Diğer Maddeler			
Toplam			

DANIŞMAN (JÜRI ADINA):
DANIŞMAN IMZASI: