KRİTİK ALTYAPI OPERATÖRLERİ İÇİN GÖRÜNTÜ İŞLEME TABANLI BİR YORGUNLUK TESPİT VE UYARI SİSTEMİ

Osman YEŞİL[1], Erdal IRMAK[2] ve Halil İbrahim BÜLBÜL[3]
[1]Hoca Ahmet Yesevi Üniversitesi, Türkiye Türkçesi ile Uzaktan Eğitim Programları (TÜRTEP), Siber Güvenlik Tezsiz Yüksek Lisans Programı
[2]Gazi Üniversitesi, Teknoloji Fakültesi, Elektrik Elektronik Mühendisliği Bölümü, Ankara
[3]Hoca Ahmet Yesevi Üniversitesi, Türkiye Türkçesi ile Uzaktan Eğitim Programları (TÜRTEP) Başkanlığı
osmanyesil@outlook.com; erdal@gazi.edu.tr; bhalil@yesevi.edu.tr

ÖZET

Haberleşme, enerji, ulaşım ve sağlık hizmetleri gibi merkezi altyapı sistemleri, ülkeler için kritik altyapı sayılmaktadır. Bu sistemlerin yönetimini sağlayan operatörlerin yorgunluk durumları son derece önem arz etmektedir. Özellikle merkezi SCADA sistem operatörlerinin kısa süreli bir dalgınlık veya dikkatsizlikleri, ciddi bir olayın veya alarmın gözden kaçmasına sebep olabilir. Oysa kritik altyapılarda bu tür olaylar çoğu zaman anlık müdahale gerektirmektedir. Bu nedenle çalışmada, operatörün uyku, yorgunluk ve dikkatsizlik gibi davranışlar göstermesi durumunda aktif olan bir alarm ve uyarı sistemi geliştirilmiştir. Bu amaçla, operatör karşısına yerleştirilen kameradan alınan gerçek zamanlı görüntüler, görüntü işleme teknikleri ile işlenmekte, belirlene özelliklere göre alarm üretilmekte, yazılım aracı olarak *OpenCV* kütüphanesinden faydalanılmış ve görüntü işleme uygulamalarında faydalanılmıştır. Sunulan tekniğin, insan kaynaklı operatör hatalarının en aza indirgenmesi için, basit ve etkili bir metodolojik yaklaşım olarak literatüre katkı sağlayacağı değerlendirilmektedir. Ayrıca, geliştirilen fikirsel yaklaşım ve yöntemin, birçok farklı alana ve altyapıya da kolayca uyarlanabileceği değerlendirilmektedir.

***Anahtar Kelimeler*—**Görüntü İşleme, Yüz Okuma, Yorgunluk Tespiti, Kritik Altyapı Operatörü

An Image Processing Based Tiredness Detection and Warning System for Critical Infrastructure Operators

ABSTRACT

Central infrastructure systems such as communication, energy, transportation and health services are considered critical infrastructures for countries. Tiredness of operators responsible for managing these systems is extremely important. Especially, an instant thoughtfulness or inattention of central SCADA system operators can cause a serious event or alarm to be overlooked. However, such events in critical infrastructures often require instant intervention. For this reason, an alarm and warning system that is activated when the operator shows some behaviors such as sleepiness, tiredness and carelessness has been developed in this study. For this purpose, real-time images captured from the camera mounted in front of the operator are processed with image processing techniques. OpenCV library, widely used in image processing applications, is preferred as software tool. It is considered that the presented technique will contribute to the literature as a simple and effective methodological approach to minimize human based mistakes. Furthermore, the idea of proposed approach and method can be easily adapted to many different areas and infrastructures.

Keywords— Image Processing, Face Detection, Tiredness Detection, Critical Infrastructure Operator

I. GİRİŞ (INTRODUCTION)

Kritik altyapı operatörleri mesai saatleri içerisinde rutin yaptığı işler ile meşgul olabilirler. Bu durum, operatörün dalgınlık veya yorgunluğuna da bağlı olarak merkezi sistemlerin yönetiminde oluşan uyarı veya hataların gözden kaçmasına sebep olabilir. Bir anlık dalgınlık bile son derece

olumsuz sonuçlar doğurabilir. Bu yönüyle konu, geçmiş literatür çalışmalarında da farklı boyutlarıyla ele alınmıştır. Örneğin Dönmez vd. (2018), insan kaynaklı kaza ve felaket durumlarının yorumlanabilmesi için veri odaklı araştırma stratejileri, kaza analiz araçları ile hataların seviyeleri üzerinde durmuş, çevresel faktörler arasında fizyolojik durumlardan bahsetmiş ve insan kontrolünde olan her sistem için aksaklık yaratabilecek sebepleri ortaya koymuştur [4]. Duyar vd. (2016), insan hataları dışında endüstriyel makinaların hatalarının veya arızalarının da önceden tahmin edilebilir olduğunu ifade etmektedir. Bu amaçla izlenecek makinaların matematiksel modellenmesine ihtiyaç duyulmaktadır [5]. Benzer şekilde Yılmaz (2018) tarafından sunulan çalışmada da yapay zekâ ve makine öğrenmesi ile otonom sistemlerin karşılayacağı ihtiyaçlardan ve endüstrilerin faydalanabileceği yararlardan bahsedilmiş, daha akıllı hale getirilerek daha gelişmiş otonom sistemlere sahip olunması durumunda çoğu insani işlemlerin robotlar tarafından yapılabileceği belirtilmiştir [21]. Öte yandan Köse (2018) ise çalışmasında, yapay zekâ çözümlerinin faydalı otonom sistemlerle birlikte endişeleri de doğurduğundan bahsetmektedir. Bazı işlemleri yapay zekâ sistemlerine bırakmanın yanında insan denetimini içermesi gerektiğini savunmuştur [11].

Yorgunluk, uyku ve dikkatsizlik gibi bazı insana özgü durumların tespit edilmesinde şüphesiz ki en önemli odak noktası yüz ve bilhassa göz hareketlerinin takibidir. Eldem vd. (2017), kamera ve optik tarayıcı gibi araçlar kullanarak alınan görüntülerin anlamlandırılarak faydalı bilgiler elde edilebileceğini ortaya koymakla birlikte görüntü işleme için görüntünün alınması sonrasında bazı yazılım tabanlı işlemlerden geçirilmesi gerektiğini belirtmiştir. Bu amaçla ele aldıkları OpenCV, EmguCV, AForge.NET kütüphaneleri gibi popüler açık kaynak çözümler ile başarılı sonuçlar elde etmişlerdir [6]. Erişti (2010) tarafından sunulan çalışmada da görüntü işleme teknolojileri arasında en çok kullanılanlardan biri olan OpenCV kütüphanesinin avantajlarından ve çalışma mantığından bahsedimiştir. OpenCV kütüphanesinin işletim sistemlerinden bağımsız ve ücretsiz olarak kullanılabilmesinin kullanıcılar için geliştirme sürecini kolaylaştırdığı belirtilmektedir [7]. Omur vd. (2017) tarafından sunulan çalışmada ise, göz hareketlerini izleme işleminin tarihsel gelişiminden ve hareketleri incelerken kullanılan yöntemlerden bahsedilmiştir. Web kullanıcılarının odaklandıkları noktaları tespit ederek daha anlamlı içeriklerin sunulabilmesi için önemli bir çalışmadır [15]. Benzer olarak Baştuğ vd. (2019) tarafından sunulan çalışmada, göz hareketleri için göz takip cihazı kullanılmış ve cihazda tutulan kayıtlardan kişilerin okuma esnasında göz hareketleriyle ilgili analizler gerçekleştirilmiştir. Göz kırpma sayıları ve göz hareketleri ile sesli ve sessiz okuma arasındaki farklar anlamlandırılmaya çalışılmıştır [2].

Samtaş vd. (2011) tarafından hazırlanan çalışmada görüntü işleme teknolojisinin farklı uygulama alanları detaylıca değerlendirilmiştir [16]. Özellikle göz ve yüz takibi uygulamalarının sürücü hatalarının önüne geçmek için geliştirildiği çalışmalar dikkat çekmektedir. Örneğin Nennioğlu vd. (2018) tarafından sunulan çalışmada, araç şoförünün refleksleri algılama ve kontrol sistemleri arasında kurulan bağlantı sayesinde aksiyonun planlanması ve şoför müdahalesi olmadan aracın kullanımının sağlanması üzerinde durulmuştur [14]. Vural vd. (2018) ise çalışmalarında, sürücü hatası kaynaklı trafik kazalarını önlemek için gerçek zamanlı görüntü işleme kullanılarak yorgunluk tespit sistemi örneği ile erken uyarı sistemi geliştirmiş ve bu amaçla Python dili ile göz hareketleri ve yüz okuma işlemleri gerçekleştirmiştir [19]. Benzer olarak Liu vd. (2002) tarafından hazırlanan çalışmada, özellikle sürücü asistan sistemlerinde göz izleme ve okuma işlemlerine ihtiyaç olduğu açıkça belirtilmiştir. Göz izleme sayesinde sürücü ile ilgili yorgunluk seviyesi gibi bazı bilgilere ulaşılabileceği ve asistanın devreye girmesi sağlanarak bazı önlemlerin alınabileceği gösterilmiştir [13].

Güncel çalışmalarda artık görüntü işleme teknikleri ile birlikte yapay zekâ ve makine öğrenmesi gibi gelişmiş tekniklerin de kullanılarak daha kesin ve etkili çözümler oluşturulmaya çalışıldığı gözlenmektedir. Örneğin Yılmaz (2019) tarafından hazırlanan çalışmada, göz hareketleri ile yapay zekâ sayesinde anlamlı sonuçlar çıkarılması sonucunda arama motorlarının kullanılabilirliğinin artırılması amaçlanmıştır. Arama motorları kullanılırken göz hareketlerinin daha çok hangi alanlarda kümelendiği gözlemlenmiştir. Çalışma, göz hareketleri sayesinde gelecekte daha akıllı ve otonom çözümler ile etkileşim halinde olunacağını göstermektedir [19]. Kesici vd. (2018) tarafından hazırlanan çalışmada, insan kaynaklı hatalar

sonucunda güç sistemlerinde yaşanabilecek risklerden bahsedilmekte ve çözüm önerisi olarak makina öğrenmesi ile iki farklı model eğitilerek oluşturulan bir karar mekanizması sunulmaktadır [10].

Açıkça görüldüğü üzere, özellikle kritik altyapı sistemlerinde anlık bir aksaklık veya hata, kitlesel sorunlara yol açabilir. Buna sebep olabilecek risklerden en önemlisi ise sistem operatörlerinin çoğu zaman mesai saatlerinde yaşadıkları yüksek iş temposu veya bedenen yaşadıkları enerji düşüklükleri nedeniyle yorgunluk, dikkat dağınıklığı veya uyku gibi durumlar ile karşı karşıya kalabilmeleridir. Bu nedenle, sunulan bu çalışmada, operatörün uyku veya yorgunluk belirtisi gösterdiği anda bir erken uyarı sistemini devreye alarak hataların önüne geçilmesi amaçlanmıştır.

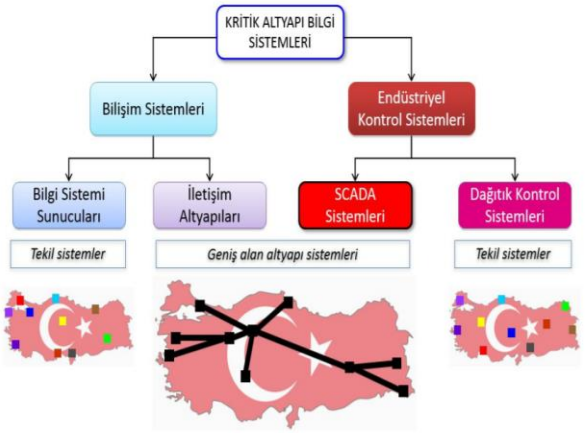
II. KRİTİK ALTYAPILARDA BİLGİ GÜVENLİĞİ (CYBER SECURITY IN CRITICAL INFRASTRUCTURES)

Kritik altyapıların güvenliğinin sağlanmasında temel olarak üç ana katman göze çarpmaktadır. Bunlar; kritik altyapının fiziksel varlıkları, görevli ve sorumlu personel ile diğer insan kaynakları ve bilişim sistemleridir. Bilişim sistemleri açısından kritik altyapıları, Şekil 1'de görüleceği üzere (Tübitak, 2020), dört farklı sınıfta değerlendirmek mümkündür [18]. Görüleceği üzere, SCADA sistemleri kritik altyapı endüstrisinde yaygın olarak kullanılan ve uzaktan denetleme ve kontrol sağlayan sistemlerdir ve temel görevleri aşağıdaki gibidir:

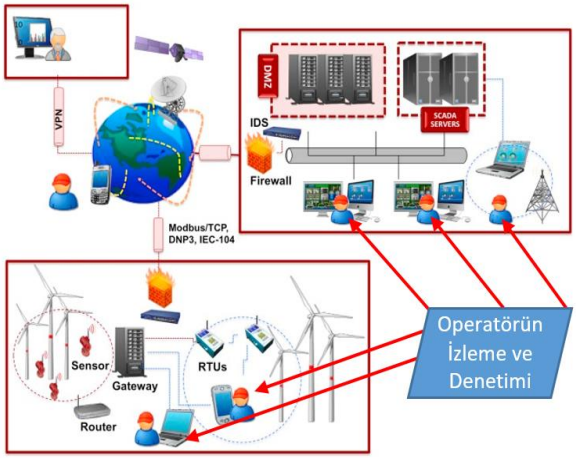
- Haberleşme ortamı sayesinde bütün SCADA bileşenlerinin haberleşmesini izlemek ve denetlemek.
- HMI yazılımı kullanarak SCADA haberleşmesiyle ilgili bilgi ve verileri grafiksel bir arayüz ile görüntülemek.
- Saha cihazlarına komut göndermek, saha cihazlarından gelen komutları almak ve haberleşme bağlantısını kontrol etmek.

Kritik altyapıların bilgi güvenliğinin sağlanmasında en önemli unsur olan SCADA sistemlerinde, operatöre ciddi bir sorumluluk düşmektedir. Zira yukarıda bahsedilen her üç işlevin de denetimi günümüz geleneksel SCADA sistemlerinde operatör tarafından yürütülmektedir. Şekil 2'de, örnek bir kritik altyapı olarak rüzgâr enerjisinden elektrik üreten bir sistemin işleyiş ve haberleşme altyapısı ile bu sistem üzerindeki temel siber güvenlik unsurları gösterilmiştir. En önemli kritik altyapı sektörü olan elektrik enerjisi üretim, iletim ve dağıtım sistemlerinin işleyişi temel olarak bu örnek ile büyük ölçüde benzerdir. Şekilden de açıkça görüleceği üzere, sürecin her aşamasında operatöre bağlı bir izleme ve denetim aşaması yer almaktadır. Dolayısıyla böyle bir kritik altyapılarda herhangi bir önemli alarmın yakalanamaması veya anlık bir operatör dalgınlığının; insan hayatını tehlikeye atacak sonuçlar, veri kaybı ve çevreye olan zararlar gibi fiziksel etkileri, tesis veya firmada ciddi maddi kayıplar gibi ekonomik etkileri, kamu güveni ve ulusal güvenliğin zarar görmesi gibi sosyal etkileri olabilir.

Kritik altyapılarda gerek yukarıda bahsedilen insan kaynaklı unsurları gerekse de bir bütün olarak güvenlik tehditlerini ele alan, Şekil 3'teki gibi bir bilgi güvenliği sürecinin işletilmesi önem arz etmektedir.



Şekil 1. Kritik altyapı bilgi sistemleri [18]



Şekil 2. Kritik bir altyapı sisteminin işleyişi ve siber güvenlik unsurları [1]

Şekil 3. Kritik altyapılarda bilgi güvenliği süreci [18]

III. YÖNTEM VE YAZILIM ALTYAPISI (METHODOLOGY AND SOFTWARE INFRASTRUCTURE)

Çalışmada, SCADA sistemi gibi merkezi bir ara yüz ekranı üzerinden izleme ve denetim işlemleri yürüten bir operatörün gözden kaçırabileceği anlık olayların önlenmesi üzerinde durulmuştur. Küçüköner'in (2005), göz organının görme ve algılama yetisini gerçekleştirir iken odak ve göz hareketlerini derinlemesine incelediği çalışmasında da belirttiği üzere, görmenin ve odaklanmanın rastlantısal durumu ile gözün açık bir durumda olması doğru orantılıdır [12]. Bu nedenle çalışmada, insan yüzü ve göz hareketlerinin sayısal modeli kullanılarak, görüntü işleme teknikleriyle yorgunluk, dalgınlık ve uyku durumları tespit edilmiştir. İzleyen alt bölümlerde, çalışmada kullanılan bazı yazılım araçları ve kütüphaneler hakkında kısa bilgiler verilmektedir.

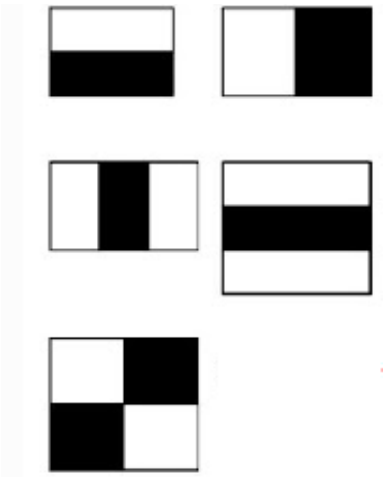
3.1. Python Dili (Python Language)

Görüntü işleme işlemleri için açık kaynak kodlu *Python* dili kullanılmıştır. 1991 yılında ilk sürümü piyasaya sürülen *Phyton*, geniş standart kütüphanesi ve dinamik yapısı ile nesne tabanlı programlamayı ve belirli bir oranda da fonksiyonel programlamayı destekleyen genel amaçlı bir programlama dilidir. *C* ve *C++* gibi dillerin aksine derlenmeye gerek olmadan çalıştırılabilmektedir. Bu durum *Python* ile program geliştirmeyi daha kolay hale getirmektedir. *Python* içerisinde bir programı yazarken ihtiyaç duyulan pek çok şey, veri yapıları, fonksiyonlar hazır olarak sunulmaktadır. Bu sayede diğer dillerde olduğu gibi bir problemi çözmek için en ince detaya kadar tasarım yapılmasına gerek kalmadan sunulan altyapı ile çok daha hızlı ve etkili bir şekilde program yazabilmek mümkündür.

3.2. *Haar Cascade* Algoritması (*Haar Cascade* Algorithm)

Yüz okuma işlemleri için eğitilmiş bir model üzerinden *Haar Cascade* algoritması kullanılmıştır. *Haar* özellikleri kullanarak nesne saptama yaklaşımı, *Haar* dalgacık dönüşümünden adapte olunarak geliştirilmiştir. Nesneler, alt parçacıklarının farklı renk dağılımı ve yoğunluk bilgisi kullanılarak farklı alt parçalara ayrılır. Parçalara ait özellikler farklı özellik setleri ile ifade edilerek nesnenin tamamı tanımlanır. Parametre olarak gelen resimdeki yüze ait alanların bulunması işlemi, tüm resim taranarak gerçekleştirilmektedir. Sınıflandırıcı, farklı boyutlardaki yüz özelliklerini bulabilir. Parametre olarak gelen resim içerisindeki değişik ölçeklerdeki yüz özelliklerini bulabilmek için aynı arama sürecini farklı ölçek parametreleri için tekrar eder [8].

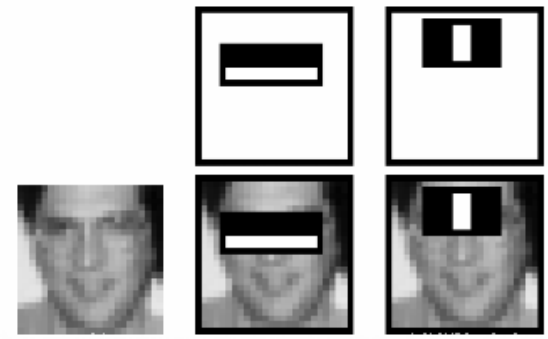
Şekil 4'te, *Haar Cascade* algoritmasının üç temel özelliği gösterilmektedir. Görüntü üzerinde belirli bir alan koyu renklerden ve belirli bir alan açık renklerden oluşuyor ise kenar özelliği olduğunu belirtmektedir. Görüntü üzerinde sırasıyla açık, kapalı, açık renkler oluşuyor ise çizgi özelliği vardır. Kare şeklinde koyu ve açık tonlar çapraz bir şekilde bulunuyor ise dört kare özelliğini belirtmektedir [3]. Verilen özellikleri kullanarak alınan görüntü üzerinde kenar, göz ve yüz gibi objelerin tespit edilmesi mümkündür.



Şekil 4. *Haar Cascade* özellikleri: kenar (üstte), çizgi (ortada), dört kare (altta)

Şekil 5'de, örnek bir yüz okuma görüntüsü verilmiştir. *Haar Cascade metodu*, insan yüzü için birçok kez eğitilmiştir ve yüz hatlarının nasıl bir yapıda olduğunu yüksek doğrulukta bilmektedir. Yüz arayan bir sistem öncelikle gözleri

aramaktadır. Eğer alınan görüntüde göz var ise burun var mı diye kontrol eder. Burun tespitinden sonra ise kaş var mı diye kontrol edilerek amaçlanan sonuçları veren yapıya sahip olması sağlanır.



Şekil 5. *Haar Cascade* ile örnek yüz okuma [3]

Haar Cascade kullanarak, alınan görüntü üzerinde nesne tanımlama işlemleri kolayca gerçekleştirilebilmektedir. Tablo 1'de, bu işlem için örnek bir *Phyton* kodu verilmiştir.

Tablo 1. Yüz okuma işlemi için *Phyton* kodu

```
import cv2
face_cascade =
cv2.CascadeClassifier('haarcascade_frontalfa
ce_default.xml')
cap = cv2.VideoCapture(0)
while 1:
   ret, img = cap.read()
   gray = cv2.cvtColor(img,
cv2.COLOR_BGR2GRAY)
   faces = face_cascade.detectMultiScale(gray,
1.3, 5)
   for (x, y, w, h) in faces:
      cv2.rectangle(img, (x, y), (x + w, y + h),
(255, 0, 0), 2)
      roi_gray = gray[y:y + h, x:x + w]
      roi_color = img[y:y + h, x:x + w]
   #show the frame window
   cv2.imshow('img', img)
   k = cv2.waitKey(30) & 0xff
cap.release()
cv2.destroyAllWindows()
```

3.3. Pyaudio

Bu makale çalışmasında, operatörün göz hareketleri algılandıktan sonra belirlenen bir sınır değer sonrası sesli alarm üretmek amacıyla Pyaudio komutu kullanılmıştır. Bu komut, Python için OSX, Windows ve Linux'ta platformlar arası bağımlılık gerektirmeyen ses çalma özelliği sağlar. Tablo 2'de, komutun kullanımına ait bir kod parçası verilmiştir.

Tablo 2. *Pyaudio* komutunun kullanımı

```
wf =
wave.open('C:\\Users\\o_yesil\\Desktop\\Döne
mProjesi\\Program\\alarmsound.wav', 'rb')
p = pyaudio.PyAudio()
```

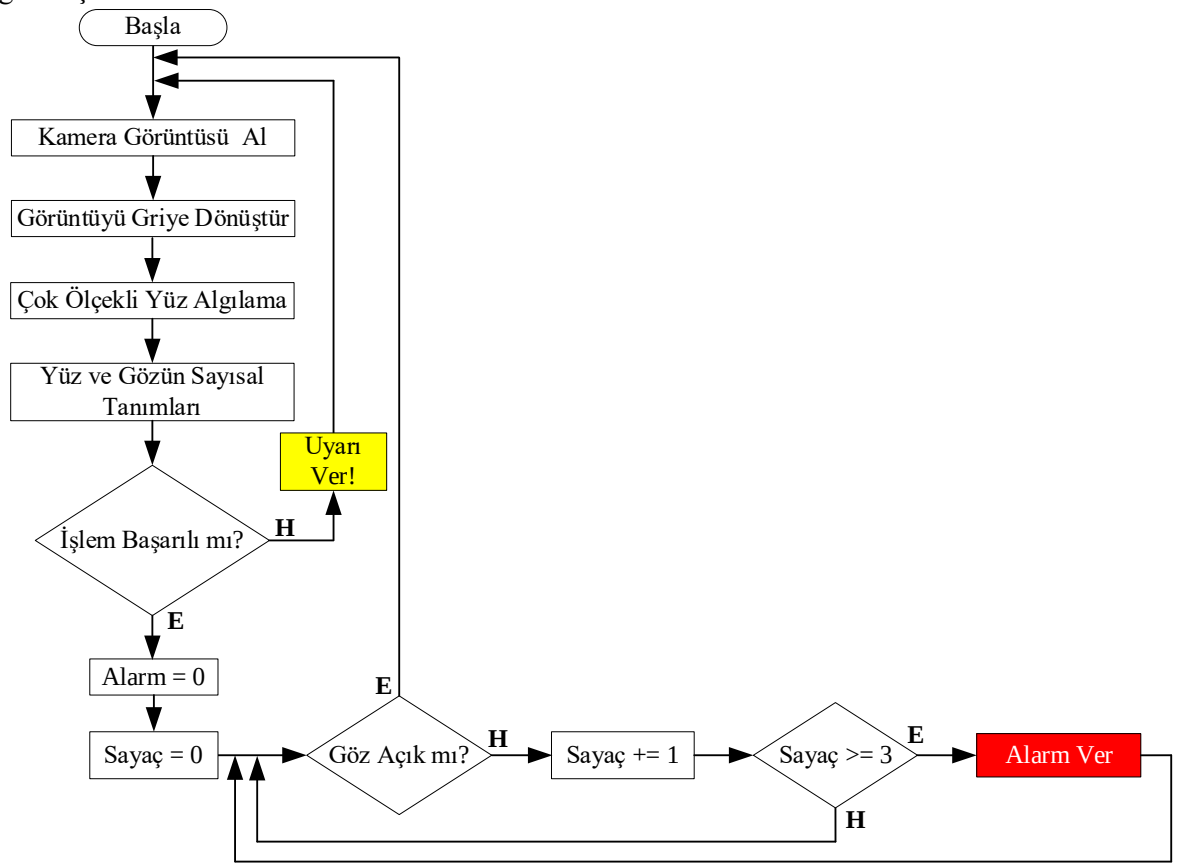
IV. TASARIM VE UYGULAMA (DESIGN AND IMPLEMENTATION)

Önceki bölümde metodolojisi anlatılan çalışmada, kullanılan donanım ve yazılımlar sayesinde operatörün herhangi bir dalgınlık veya uyku durumunun algılanarak uyarı sisteminin devreye girmesi ile olası hataların önüne geçilmesi amaçlanmıştır. Şekil 6'da, bu amaçla geliştirilen çalışma yönteminin temel aşamalarını gösteren bir akış diyagramı verilmektedir.

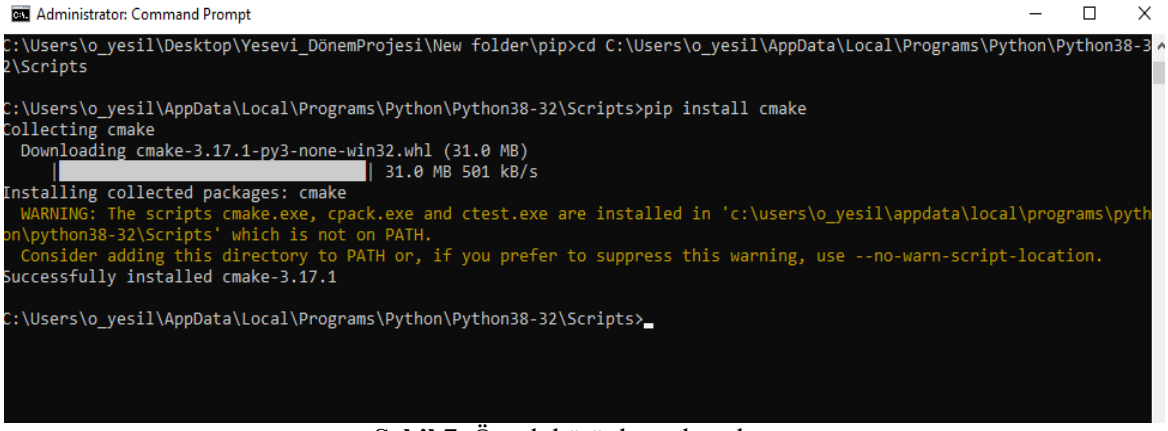
Sistemde öncelikle yüz algılaması için eğitilmiş "haarcascade_frontalface_default" modeli ile kameradan alınan görüntü üzerinde, OpenCV kütüphanesinde yer alan COLOR_BGR2GRAY kullanılarak renk dönüşümü yapılır. Yüz algılandıktan sonra sağ ve sol göz koordinatları (x, y) belirlenir. Göz kapatma durumunu tespit etmek için, gözün modelde hangi koordinatlara eşit olduğu belirlenir ve bu koordinatlara göre uyku durumu algılanır.

Uyku durumu tespit edildiğinde, bunun geçici bir göz kapaması olup olmadığının araştırılması için aynı olayın 3 kez tekrarlanması şeklinde bir esneklik oluşturulmuştur. Buradaki tekrar sayısı test amaçlı olarak deneysel çalışmanın doğrulanması için 3 olarak seçilmiştir. Uygulamanın gerçekleştirileceği kritik altyapı sektörünün gereklerine göre yazılım üzerinden farklı bir bekleme süresi de ayarlanabilir. Belirlenen süre boyunca gözün kapalı kalması durumunda, hem ekran üzerinde yazılı hem de dâhili ses sistemi kullanılarak sesli alarm sistemi devreye girmektedir. Tasarımın yazılım geliştirmesinde Python 3.6 kullanılmıştır. Bu yazılım paketi kurulduktan sonra tasarımda kullanılan kütüphane kurulumları için Python betik yükleme aracı olan pip betiğinden faydalanılmıştır. Bu amaçla curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py şeklinde bir komutun, komut satırı üzerinden girilmesi yeterlidir. Bu yöntemle, geliştirilen tasarımda kullanılan üç kütüphane olan imutils, cmake ve opencv-python kurulmuştur. Şekil 7'de

örnek bir kütüphane kurulum sürecine ait ekran görüntüsü verilmektedir. Görüntü işleme işlemi için kamera aracılığı ile alınan gerçek zamanlı görüntünün sayısal hale dönüştürülmesi gerekmektedir. Bu amaçla daha önceden defalarca eğitilmiş bir model olan haarcascade_frontalface_default.xml dosyası kullanılmaktadır. Bu işleme ilişkin kod tasarımı Tablo 3'te verilmektedir.



Şekil 6. Tasarlanan algılama ve uyarı sisteminin akış ve çalışma diyagramı



Şekil 7. Örnek kütüphane kurulumu

Tablo 3. Kameradan alınan görüntünün sayısal formata dönüştürülmesi

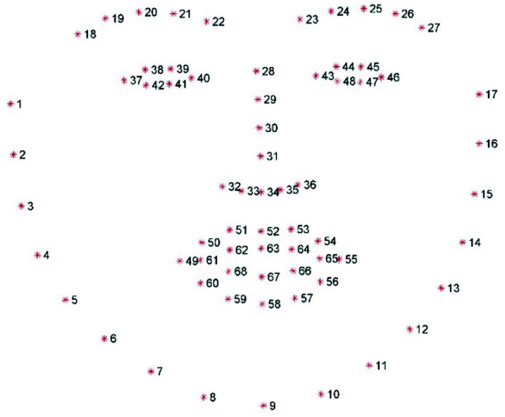
```
face_cascade=
cv2.CascadeClassifier('haarcascade_frontalfa
ce_default.xml')
eye_cascade =
cv2.CascadeClassifier('haarcascade_eye.xml')
cap = cv2.VideoCapture(0)
```

Alınan görüntü sayısal formata dönüştürüldükten sonra, yüz ve göz hatlarının sayısal haritasının çıkarılması için *OpenCv* aracılığıyla eğitilmiş model dosyaları kullanılmaktadır. Bu amaçla kullanılan *detectMultiScale* metodu, yüz algılama

ve sayısal harita dosyası olan *haarcascade* içinde yer alan objeleri tanımak için kullanılan genel bir fonksiyondur. İlk parametresi gri tonlamaya dönüştürülen resim değişkenidir. İkinci parametre olan *scaleFactor*, resimde yer alan bir yüzün başka bir yüzden büyük olması durumunda ortaya çıkan bozulmayı engeller. *minNeighbors* isimli üçüncü parametre, yanındaki nesnelerle ilişkili olarak nesne algılamak için kullanılan bir algoritmadır. Bütün bu işlemlerin gerçekleştirilmesini gösteren kod tasarımı Tablo 4'te ve çıkarımı yapılan yüz hatlarına ait sayısal harita görünümü ise Şekil 8'de verilmiştir.

Tablo 4. Görüntü işleme ve yüz algılama adımlarına ilişkin kod tasarımları

```
while 1:
    ret, img = cap.read()
    gray = cv2.cvtColor(img,
cv2.COLOR_BGR2GRAY)     faces =
face_cascade.detectMultiScale(gray, 1.3, 5)
```



Şekil 8. Görüntü üzerinden yüz hatlarına ait sayısal haritanın çıkarımı

Tablo 5'te verilen kod tasarımından da görüleceği üzere, kameradan alınan görüntü içinde başarıyla yakalanan yüz bölümleri çerçeve içine alınmakta ve sonrasında algılanan yüz bölümlerinden göz hareketlerinin ayrılması ve okunması işlemlerinin başarı oranını artırmak için *roi_color* isimli değişkene aktarılmaktadır.

Tablo 5. Yüzün çerçeve içine alınması ve göz algılama

```
for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x + w, y + h),
(255, 0, 0), 2)
        roi_gray = gray[y:y + h, x:x + w]
        roi_color = img[y:y + h, x:x + w]
        # cv2.imshow("x",roi_gray)
        # cv2.imshow("y", roi_color)
        eyes =
eye_cascade.detectMultiScale(roi_gray)
```

Tablo 5'te kullanılan yöntem sonucu eğer bir göz algılandı ise öncelikle algılanan gözün çevresi, sayısal harita aracılığı ile dikdörtgen içine alınmaktadır ve gözün açık durumda olduğu bilgisi kayıtlanmaktadır. Eğer gözün açık olması durumu yakalanamadı ise öncelikle bu durum *closed* adlı sayaç içine alınmaktadır. Eğer 3 defa gözün açık olması durumu algılanmadı ise kullanıcının uyku halinde olduğuna karar verilmekte ve operatörü uyaracak alarm sisteminin devreye girmesi sağlanmaktadır.

Daha önce de belirtildiği üzere, tasarımın bu bölümünde deneysel doğrulama için sadece 3 adımlı bir döngü süresi öngörülmüştür. Uygulamada, sistem ihtiyaç ve gereksinimleri doğrultusunda bu değer farklı bir şekilde de ayarlanabilir. Gözün çerçeve içine alınması ve açık durumda olduğunun kayıtlanmasına ilişkin kod tasarımı, Tablo 6'da verilmektedir. Tablo 7'de, gözün 3 defa açık durumda algılanmaması ve operatörün uyku durumunda olması sonucu alarm sisteminin aktivasyonuna ilişkin kod tasarımı yer almaktadır. Alarmın çalma ve durdurma adımlarına ilişkin kod tasarımları ise Tablo 8'deki gibidir.

Tablo 6. Gözün çerçeve içine alınması ve açık durumda olduğunun algılanması

```
def openeye():
print("Göz Açık")
        if eyes is not ():
          for (ex, ey, ew, eh) in eyes:
              cv2.rectangle(roi_color, (ex, ey), (ex
+ ew, ey + eh), (0, 255, 0), 2)
              openeye()
```

Tablo 7. Gözün açık/kapalı durumuna göre uyku tespiti ve alarm sisteminin aktivasyonu

```
@counter
def closed():
print("Göz kapalı")
    if eyes is not ():
        for (ex, ey, ew, eh) in eyes:
            cv2.rectangle(roi_color, (ex, ey), (ex
+ ew, ey + eh), (0, 255, 0), 2)
            openeye()
    else:
```

```
        closed()
        if closed.count == 3:
        print("Operatör uyumaktadır.")
           now = datetime.datetime.now()
           CurrentTime = now.strftime("%Y-
%m-%d %H:%M:%S")
  cv2.putText(img, "Yorgunluk Tespiti: " +
              str(CurrentTime), (10,20),
              cv2.FONT_HERSHEY_SIMPLEX
              , 0.5, (0xFF, 0xFF, 0xFF0), 2)
stream.start_stream
sound()
```

Tablo 8. Alarm çalma ve durdurma işlemleri

```
wf =
wave.open('C:\\Users\\o_yesil\\Desktop\\Döne
mProjesi\\Program\\alarmsound.wav', 'rb')
p = pyaudio.PyAudio()
def callback(in_data, frame_count, time_info,
status):
   data = wf.readframes(frame_count)
   return (data, pyaudio.paContinue)
stream =
p.open(format=p.get_format_from_width(wf.g
etsampwidth()),
              channels=wf.getnchannels(),
              rate=wf.getframerate(),
              output=True,
              stream_callback=callback)
stream.stop_stream()
```

Tasarlanan kodlar ve uygulama sayesinde, operatörün gözünün açık veya kapalı olma durumları üzerinden başarılı bir şekilde uyku ya da dalgınlık kaynaklı dikkat dağılımlarının önlenmesi sağlanmakta ve bu durumların algılanması halinde uyarı sistemi devreye girmektedir.

Şekil 9'da, açık durumda olan gözlerin algılanması gösterilmiştir. Şekil 10'da ise, 3 defa kapalı durumda algılanan gözler sonucunda, ekranın sol üst köşesinde uyarı metninin belirdiği görülebilir. Ayrıca bu durumda sesli alarm sistemi de uyarı vermektedir.

Örnek bir olay çalışması (case study) olan bu makalede sadece ekran uyarısı ve ses uyarısı üzerinde durulmuş olsa da basit bazı güncellemelerle mail atılması gibi farklı uyarı mekanizmalarının da devreye alınması mümkündür.

V. SONUÇ VE DEĞERLENDİRMELER (CONCLUSION and EVALUATIONS)

Bu makalede, özellikle kritik altyapılar gibi önemli sistemlere ait SCADA veya benzeri denetim ara yüzlerinin izlenmesi ve denetiminden sorumlu operatörlerin, uyku veya benzer dikkat dağılımlarının tespit edilmesi ve buna ilişkin alarm nitelikli uyarı mekanizmalarının geliştirilmesi için bir yöntem sunulmuş, çözüm algoritmaları verilmiş, bunun için ön bir çalışma yapılarak, bundan sonra yapılacak olan çalışmalara bir altyapı oluşturulmuştur. Elde edilen bulgular, bu çalışmanın mevcut kütüphaneler ve uygulamalar kullanılarak hızlıca yapılabileceğini gösterse de yüksek seviyede bir sistem güvenliği sağlanması için bu ve buna benzer çalışmalara ihtiyaç duyulabilecektir.

Yöntemin uygulanabilirliğini test etmek için operatörlerin yüz ifadelerine dayalı olarak göz hareketleri hakkında bilgi elde eden prototip bir yazılım geliştirilmiştir. Tasarım aşamasında *Python* aracılığı ile sayısal görüntü işleme yöntemleri kullanılmıştır.

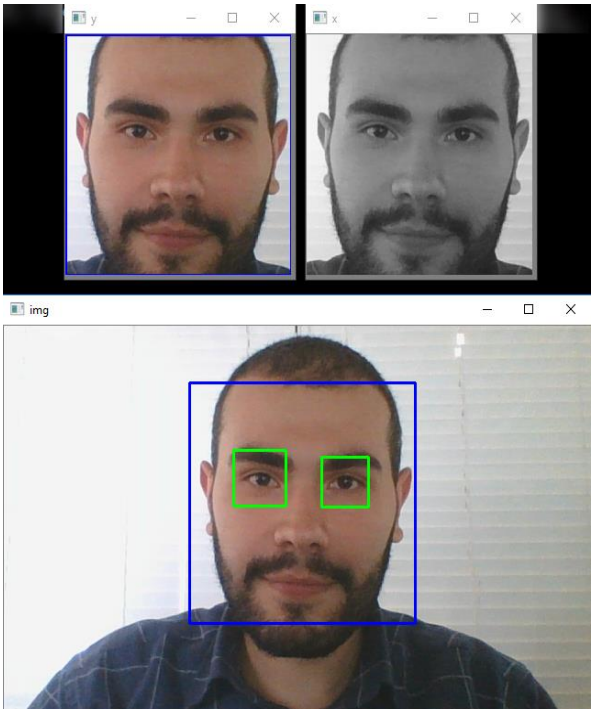
Operatör yorgunluk tespiti için ise *Haar Cascade* sınıflandırma algoritmaları uygulanmıştır. Böylece, operatör karşısına yerleştirilen video kameradan alınan görüntüyü kullanarak, gerçek zamanlı yüz okuma ve yüz işaretçileri ile uyku veya yorgunluk durumları tespit edilmektedir. Özellikle insan yüzü için birçok kez eğitilmiş olan *Haar Cascade* algoritması, yüz hatlarının nasıl bir yapıda olduğunu yüksek doğrulukta bilmektedir ve sadece yüzyüze değil farklı açılardan bakıldığında da yüzü ve gözü başarıyla yakalayabilmektedir. Böylece yüzün ve gözün algılanamadığı farklı durumlar için de geçerli bir yöntem ortaya çıkmaktadır. Örneğin bayılma, sağa sola yukarı aşağı yönlü belirli bir açıdan daha fazla uzun süreli bakarak gözün algılanamaması gibi durumlarda da alarm üretilmesi mümkün olmaktadır.

Bilimsel katkı olarak ele alındığında;

- sunulan çalışmada bir ara yüz geliştirilmiş ve belirlenen problemin çözümünde kullanılmıştır.
- yazılım ve kod tasarımlarının optimizasyonu yapılmıştır.
- yeni bir kontrol tekniğinden ziyade metodolojik bir yaklaşım ve uygulama önerisini literatüre sunulmuştur.

Sunulan bu çalışmanın iyileştirilmesi için aşağıdaki çalışmalar gelecek çalışmalarda tamamlanacaktır. Bunlar;

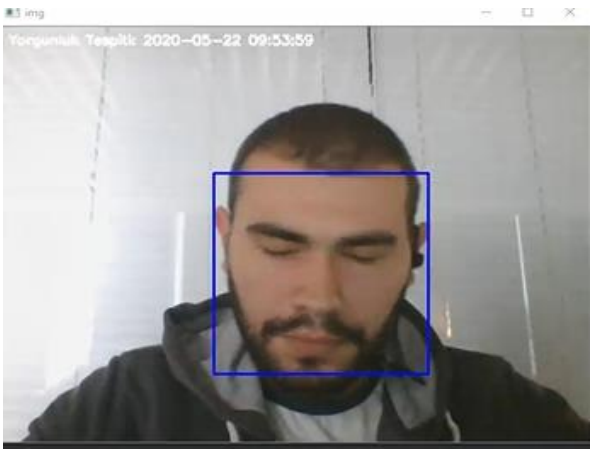
- Sunulan bu çalışmada ilgili kodların geliştirilmesi, bunun farklı senaryolar için geliştirilmesi, sadece bir PC karşısında değil oda içi veya ortam içi bir kameradan alınacak gerçek görüntülerle uyuma durumu tespit edilecektir.
- Geliştirilen yöntemin başarısı gerçek ortamlar ve senaryolar (uyuma, uzanma, farklı kişileri içeriye alma, vb) için alarm üretilmesi durumu test edilecektir.
- Metodolojik fikrin öne çıkarılması amaçlı bu çalışmanın sonraki aşamalarında, kodlarda yapılacak iyileştirmeler, farklı görüntü işleme tekniklerinin ve sınıflandırma algoritmalarının denenmesi, yorgunluk ve uyku algılaması sonucu ileri düzey alarm üretilmesi, farklı denetim sistemlerinin devreye alınması üzerinde durulabilecektir.



Şekil 9. Açık durumda olan gözlerin algılanması

Sonuç olarak; geliştirilen bu ve buna benzer çalışmaların ve çözümlerin;

- Uzaktan eğitim sistemlerinde yapılan sınavlarda öğrencinin hem dikkatini hem de yapılacak olan testlerde tamamen olmasa da kısmen kullanılabileceği,
- Öğrenci/okuyucu/kullanıcıların öğrenme davranışını belirleme kullanılabileceği,
- Özellikle video konferans sistemleri ile ders verme ve sınav yapma gibi hususların popüler olduğu günümüzde olası çözüm geliştirmeye katkı sağlayacağı değerlendirilmektedir.



Şekil 10. Gözün kapalı olması durumunun algılanması ve uyarı verilmesi

KAYNAKLAR (REFERENCES)

[1]. Alcaraz, C., Zeadally, S., *Critical infrastructure protection: Requirements and challenges for the 21st century*, International Journal of Critical Infrastructure Protection, 2015, 8: 53-66

[2]. Baştuğ, M., Keskin, K., Şimşek, İ., *Sesli ve Sessiz Okumada GözHareketleri: Bir Göz İzleme (Eye Tracking) Çlışması.* Eskişehir Osmangazi Üniversitesi Sosyal Bilimler Dergisi, 2019, 20: 327-337.

[3]. Deniz, E., *OpenCv Haar Cascade ile Yüz Tanıma*, İnternet Kaynağı: https://ertugruldeniz.com/goruntu-isleme-haar-cascade-nedir-opencvhaar-cascade-ile-yuz-tanima-142, Son Erişim Tarihi: 22.05.2020

[4]. Dönmez, K., Suat, U. *İnsan Faktörleri Analiz ve Sınıflandırma Sistemi'nin (HFACS) Literatürde Yaygın Kullanımının Değerlendirilmesi*. Journal of Aviation, 2018, 2.2: 156-176

[5]. Duyar, A., Önel, İ.Y., İzzet, Y., Özdemir, H. *İnovatif Model Bazlı Arıza Erken Uyarı Yazılımıyla Beklenmedik Duruşlara Son Verme*, Mühendis ve Makina, 2016, 57.672: 44-49.

[6]. Eldem, A. H., Palalı, A., *Görüntü işleme teknikleriyle yüz algılama sistemi geliştirme*. Bitlis Eren Üniversitesi Fen Bilimleri Dergisi, 2017, 6.2: 44-48.

[7]. Erişti, E., *Görüntü İşlemede Yeni Bir Soluk, OPENCV*, Akademik Bilişim'10- XII. Akademik Bilişim Konferansı Bildirileri, 10-12 Şubat 2010, Muğla.

[8]. Kaplan, A., *Gerçek ve Yarı Gerçek Zamanlı Yüz Tespit Etme*. Yüksek Lisans Tezi, Fırat Üniversitesi Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, 2018.

[9]. Kara, P. Ö., Günay, E. C., *Çernobil Kazası ve Etkileri*. Mersin Üniversitesi Tıp Fakültesi Lokman Hekim Tıp Tarihi ve Folklorik Tıp Dergisi, 2013, 3.2.

[10]. Kesici, M., *Güç Sistemlerinde Geçici Hal Kararsızlığının Ağaç Tabanlı Makine Öğrenmesi Yöntemleri ile Erken Tespiti*, Güç Sistemleri Dergisi, 2018.

[11]. Köse, U., *Güvenli Yapay Zekâ Sistemleri İçin İnsan Denetimli Bir Model Geliştirilmesi*. Mühendislik Bilimleri ve Tasarım Dergisi, 2018, 6.1: 93-107.

[12]. Küçüköner, M., *Görme Üzerine Bir İnceleme*. Sanat Dergisi, 2005, 8: 31-34.

[13]. Liu, X., Xu, F. L., Fujimura, K., *Real-time eye detection and tracking for driver observation under various light conditions.* Intelligent Vehicle Symposium, 2002. IEEE. IEEE, 2002. p. 344-351.

[14]. Nennioglu, A. K., Koroglu, T. *Otonom araçlarda hareket planlaması*. Artıbilim: Adana Bilim ve Teknoloji Üniversitesi Fen Bilimleri Dergisi, 1 (2), 2018, 20-29.

[15]. Omur, S., Aydoğdu, A. G., *Eye tracking researches and new trends in the field of communication*. International Journal of Social Sciences and Education Research, 2017, 3.4: 1296-1307.

[16]. Samtaş, G., Gülesin, M., *Sayısal görüntü işleme ve farklı alanlardaki uygulamaları*. Electronic Journal of Vocational Colleges, 2011, 2.1: 85-97.

[17]. Tarık, A. K. *İç Güvenlik Yönetimi Açısından Kritik Altyapılarını Korunması*. ASSAM Uluslararası Hakemli Dergi, 2019, 42-51.

[18]. TÜBİTAK, *Kritik Bilgi Sistem Altyapıları için Asgari Güvenlik Önlemleri Dokümanı*, İnternet Kaynağı: https://www.uab.gov.tr/uploads/pages/siber-guvenlik/kritik.pdf, Son Erişim Tarihi: 04.07.2020

[19]. Vural, R., A., Sert, M. Y., Karaköse, B. *Gerçek Zamanlı Sürücü Yorgunluk Tespit Sistemi*. Marmara Fen Bilimleri Dergisi, 2018, 30.3: 249-259.

[20]. Yılmaz, İ. G., et al. *Göz Hareketlerini İzleme Yöntemiyle Arama Motorlarının Otomatik Tamamlama Özelliğinin Kullanılabilirlik Açısından İncelenmesi*. Tasarım Enformatiği, 2019, 1.1: 48-58.

[21]. Yılmaz, F. *Robotlar Hayatımızda*. FSM İlmî Araştırmalar İnsan ve Toplum Bilimleri Dergisi, 2018, 12: 109-120.