SOSYAL MESAFE ANALİZİ

Ahmet Hazar Haspolat, Mustafa Berk Taşkın, Cüneyt Balcı, Ömer Sait Yorulmaz

Fenerbahçe Üniversitesi

Bilgisayar Mühendisliği

İstanbul, Türkiye

e-mail: {ahmet.haspolat, mustafa.taskin, cuneyt.balci, omer.yorulmaz}@stu.fbu.edu.tr,

*Özetçe — Görüntü işleme yöntemi ile kişi tespiti ve takibi işlemleri yapılıp; videoda tespit edilen kişiler arasındaki mesafenin hesaplanması sonucunda sosyal mesafe kurallarına uygun hareket edilip edilmediğinin belirlenmesi ve sosyal mesafe analizi yapılması.*

Anahtar Kelimeler — Görüntü İşleme, Python, Yapay Zeka, OpenCV

*Abstract — People detecting and people tracking operations are performed with image processing; determining whether the social distance rules are followed as a result of calculating the distance between the detected people in the video and analyzing the social distance.*

Keywords — Image Processing, Python, AI, OpenCV

# Giriş

Bu dokümanın kapsamında, görüntü işleme yöntemi ile canlı yayın veya video formatında bir görüntü verisinin işlenerek, görüntü içerisindeki insanların arasında bulunan mesafe üzerinden bu kişilerin arasında bir sosyal mesafe ihlalinin var olup olmadığının analiz edilmesi süreci anlatılacaktır.

Süreç akışında temel olarak;

* YOLO algoritması kullanılarak görüntü içerisindeki insanların tespit edilmesi
* Tespit edilen insanların arasındaki mesafenin hesaplanması
* Hesaplanan mesafe baz alınarak görüntü içerisindeki kişiler arasındaki sosyal mesafe ihlal analizi yapılması
* Elde edilen analiz kapsamında işlenmiş verilerin video formatında kaydedilmesi

adımları açıklanmıştır.

Bahsi geçen yazılımsal sürecin gerçekleştirilmesi için tasarlanmış sistem mimarisi, kullanılan geliştirme araçları ve geliştirilen proje kapsamında elde edilen sonuçlar da doküman içerisinde detaylandırılmıştır.

# Sistem Mimarisi

‘Sosyal Mesafe Analizi’ projesi geliştirilirken IDE olarak Visual Studio Code aracı kullanılmıştır. Proje geliştirilirken bilgisayar görüşüne yönelik programlama fonksiyonları kütüphanesi olan OpenCV ve bir nesne tespit algoritması olan YOLO/COCO kullanılmıştır. OpenCV gerçek-zamanlı bilgisayar görüsü uygulamalarında kullanılan açık kaynaklı bir kütüphanedir. YOLO (You Only Look Once) ise gerçek zamanlı bir nesne algılama sistemidir ve tek adımda nesne algılama stratejisini kullanır. Bu proje tasarlanırken COCO veri setiyle eğitilmiş versiyonu kullanılmıştır.

# Kullanılan Yazılım

Projenin işleyişi aşamasında baz alınan video verisinde; bir sokak içerisinde yürüyen insanları algılamak, algılanan insanların koordinatlarının belirlenmesiyle birlikte aralarındaki mesafeyi hesaplamak ve hesaplanan mesafe baz alınarak bir sosyal mesafe ihlalinin olup olmadığı sonucuna ulaşmak adına OpenCV kütüphanesi ve COCO veri setiyle eğitilmiş YOLOv3 algoritmaları kullanılmıştır.

Proje akışı aşağıdaki adımlar üzerinden yürütülmüştür:

* Başlangıç olarak YOLO klasörünün içerisindeki eğitilmiş verileri (yolov3.weights), labelları (coco.names) ve yolo.cfg dosyası görüntü işleme algoritmasının çalıştırılabilmesi için Visual Studio Code IDE’si üzerinde oluşturulan projeye entegre edildi.
* Uygulama geliştirme aşamasında baz alınan video, OpenCV kütüphanesi yardımıyla frame frame okunabilecek ve işlenebilecek şekilde dönüştürüldü.
* ImageProcess sınıfından her frame okunmasında standart değerlere dönecek ve YOLO konfigürasyonları ile beslenecek şekilde obje oluşturuldu.
* Yolo ile beslenmiş olan ImageProcess sınıfı içerisine verilecek her frame nesne tanıma için temelde 3 adet fonksiyon ile beslenecektir.
* Okunan frame ilk olarak ImageProcess içerisinde olan getLayerOutputs fonksiyonu tarafından işleme alındı. İlgili fonksiyon gönderilmiş frame’i binary objectlere çevirerek yolo algoritmasını besliyor.
* Yolo algoritması tek adımda nesne algılaması stratejisi kullanıyor. Yolo beslendiği frame’i işleyip dışarıya output olarak layerOutputs veriyor. İçerisinde saptamasını yaptığı nesnelerin hangi nesne olduğunu, nesnenin koordinatları ve saptama yapılmış nesnenin hangi güven oranıyla yapıldığının bilgilerinin dönüyor.
* Saptanan nesneler arasında “person” olanları ve %50 den daha fazla güven oranı varsa, bu nesnelerin bounding box bilgilerini ve her bir box için geçerli olan güven oranlarını saklıyoruz. Yalnız saptanan nesneler her bir nesne için birden fazla bounding box ve güven oranı döndüğünü unutmayalım.
* Saklanan bu bounding boxlar ve güven oranları non-max suppresion mekanizması içerisine gönderiliyor. Non-max suppresion matematiksel bir yaklaşım olarak, saptanmış nesneler için en uygun saptama için nesnelerin güven oranları içerisinde olan bounding boxları kullanıyor bu sayede görüntü içerisinde saptanan bir nesne için belki de var olan yüzlerce farklı saptama noktasını ve güven oranlarını en uygun seviyeye indirgeyip, her nesne için yalnızca bir adet bounding boxı bize veriyor.
* Bu işlemden sonra elimizdeki bounding boxlar bize o an frame içerisinde olan insanların bilgilerini veriyor. Bounding box’ın koordinatları görüntüdeki kişilerin koordinatlarına, enleri ve boyları kişilerin en ve boylarına gibi parametrelerle direkt olarak eşleşebiliyor.
* Bu NMS işleminden çıkan bounding boxlar birer person objesine çevriliyor. Çevirilen her bir person objesi gerekli hesaplamaların yapılaması ve hesaplamalar sonucunda oluşturulacak olan videoda işlenilmek üzere utils.py içerisinde olan hesaplama fonksiyonlarına gönderiliyor.
* Kişilerin koordinatları baz alınarak hesaplanan kişiler arasındaki mesafe ölçümünde, kişinin en genişliğin iki katı, bir başka kişi ile arasındaki mesafeden daha az ise sosyal mesafe ihlali olarak değerlendiriliyor ve, bu kişilerin etrafı kırmızı bir kutu ile gösterilip, ihlali paylaştığı kişi ya da kişiler arasında “Sosyal Mesafe İhlali” yazdırılması için ayarlanıyor.
* Kordinatları saptanmış, mesafe ölçümleri yapılmış ve sosyal mesafe ihlal durumu belirlenmiş kişiler, yeni video işaretler ve yazılar yardımıyla gösterilmesi için tekrardan ImageProcess objesi içerisindeki drawRectangle ve putText fonksiyonları içerisine gönderiliyor. Kişilerin durumlarına göre çizimleri yapan fonksiyon nihai frame’i de oluşturmuş oluyor.
* Nihai frame ise son olarak videoWriter objesindeki write fonksiyonuna gönderilerek, framelerin output-video klasörü içerisinde bir video haline getiriliyor.

# IV Sonuçlar

Bu projenin tasarımı ve geliştirilmesi aşamasında öncelikle açık kaynaklı bir kütüphane ile çalışma gereksinimi keşfedildi ve sonrasında bu gereksinim üzerinden araştırma, deneme-yanılma, uygulama konusunda ilerleme kaydedildi. İkincil olarak projenin temel konusu olan ‘Görüntü İşleme’nin yazılımsal temelleri, hareketli ve sabit görüntülerin nasıl manipüle edilebileceği ve elde edilen veriler üzerinden görüntünün nasıl işleneceği hususunda kazanım sağlandı. Bu bilgiler ışığında yazılım alanında farklı bir dala dair vizyon kazanıldı.

##### Proje Ekibi

AHMET HAZAR HASPOLAT – Çamlıca Doğa Koleji’nde lise eğitimimi tamamlamış olup, lisans eğitimime Fenerbahçe Üniversitesi - Bilgisayar Mühendisliği bölümünde devam etmekteyim. Akademik eğitimime görüntü işleme ve yapay zeka alanlarında devam etmeyi hedefliyorum.

MUSTAFA BERK TAŞKIN – Bursa Doğa Koleji Anadolu Lisesi mezuniyetimin ardından, lisans eğitimime Fenerbahçe Üniversitesi - Bilgisayar Mühendisliği bölümünde devam etmekteyim. Python ve Javascipt dillerinde kendimi geliştirmeye devam ediyorum.

CÜNEYT BALCI – 28.08.2000 yılında İstanbul'da doğdum. 2018 yılında Final Temel Lisesi'nden mezun oldum. Şu anda Fenerbahçe Üniversitesi - Bilgisayar Mühendisliği bölümünde lisans eğitimi almaktayım.

ÖMER SAİT YORULMAZ – Fenerbahçe Üniversitesi - Bilgisayar Mühendisliği öğrencisiyim. Akademik eğitimimle eşzamanlı olarak, 2018 yılından beri özel bir şirkette, Software Developer pozisyonunda görev almaktayım.

##### Referans Dosyalar

Github link: <https://github.com/fbuni/sosyal-mesafe-analizi>

YouTube link: <https://youtu.be/kWJ96iv2-o8>

##### Kaynaklar

<https://pjreddie.com/darknet/yolo/>

<https://opencv.org/>

<https://docs.python.org/3.8/>