

**GERÇEK ZAMANLI SOSYAL MESAFE ANALİZİ**

Arda ALHAN, Doğa TURAN, Evrim Arda KALAFAT

Fenerbahçe Ünversitesi

Bilgisayar Mühendisliği ve Endüstri Mühendisliği

İstanbul, Türkiye

*e-mail*: {arda.alhan , doga.turan, evrim.kalafat }@stu.fbu.edu.tr

***Özet***

Sosyal mesafe kurallarının denetimini yapmak için Python yazılım dili kullanılarak görüntü işleme temelli bir uygulama geliştirilmiştir. Uygulamada insan takibi ve sayma işlemlerinin üstüne aralarındaki mesafe hesaplanarak sosyal mesafe kuralına uygunluğun analizi yapılmıştır.

***Anahtar Kelimeler***

Python, OpenCV, Sosyal Mesafe

***Abstract***

This is an application to monitoring social distance in Python programming language. In practice, we measured the distance between people in the test video and detected those who do not comply with social distance.

***Key Words***

Python, OpenCV, Social Distance

**GİRİŞ**

Bu proje, bir sokak görüntüsündeki insanların sosyal mesafe kuralına uygun hareket edip etmediğini göstermek için tasarlanmıştır. Görüntüye giren her kişiyi seçerek, diğer kişilerle arasındaki mesafeyi ölçen bu yazılım, aradaki mesafe belli bir seviyenin altına indiğinde uyarı vermektedir. Görüntüdeki kişilerin ve sosyal mesafeye uymayanların sayısını ayrı ayrı gösteren bu uygulama toplumun bu konudaki bilinç seviyesini ölçmek için bir araç olarak kullanılabilir.

**YAZILIM MİMARİSİ**

1. KULLANILAN KÜTÜPHANELER

Bu projede, Numpy, OpenCV ve imutils kütüphaneleri kullanılmıştır.

openCV: Gerçek zamanlı olarak görüntüyü okumak ve işlemek için kullanılan açık kaynaklı bir kütüphanedir.

Numpy: büyük çok boyutlu diziler ve matrisler için destek eklerken bu dizilerle çalışmak için yüksek düzeyli matematiksel fonksiyonların geniş bir koleksiyonudur. Bu projede de koordinatları tutan dizilerde ve koordinatlar arasındaki mesaje ölçüm işleminde kullanılmaktadır.

Imutils: görüntü işlemede kolaylık sağlayan bir dizi fonksiyonu kapsar. Bu projede bu kütüphanenin sağladığı “non\_max\_suppression” fonksiyonundan yararlanılarak görüntüye giren insanların seçimini en iyi hale getirilmesi sağlanmıştır.

1. ÇALIŞMA MANTIĞI

Referans koddaki HOG yaklaşımı kullanılarak insan tespiti yapılmakta. detectByPathVideo fonksiyonu ile bir kare okunup detect fonksiyonuna beslenmektedir. Detect fonksiyonunda ise detectMultiScale fonksiyonu tespiti yapıp, fonksiyonun döndürdüğü değerler ile insanların etrafına kareler çizilmektedir. Tespit edilen insanların koordinatları kullanılarak orta noktaları tespit edilmekledir. Orta noktalar arsındaki mesafe tespit edilen insanların ortlama enlerinin 2 katı ile kıyaslanıp sosyal mesafe kuralının ihlal edilip edilmediğinin analizi yapılmaktadır. Sosyal mesafe kuralı ihlalinde, ihlal eden kişileri içine alan kareler kırmızıya döner ve ihlal eden kişiler sayaçta sayılır.

**YAZILIM**

import numpy as np  
import cv2, imutils, argparse  
from imutils.object\_detection import non\_max\_suppression  
  
  
def average(lst):  
 if len(lst) == 0:  
 return 0  
 else:  
 return sum(lst) / len(lst)  
  
  
def detect(frame):  
 (bounding\_box\_cordinates, weights) = HOGCV.detectMultiScale(frame, winStride=(4, 4), padding=(8, 8), scale=1.03)  
  
 # non-maxima suppression  
 bounding\_box\_cordinates = np.array([[x, y, x + w, y + h] for (x, y, w, h) in bounding\_box\_cordinates])  
 nms\_box = non\_max\_suppression(bounding\_box\_cordinates, probs=None, overlapThresh=0.65)  
  
 person = 1  
 # calculate center of each box and draw the bounding boxes for people  
 centers = []  
 widths = []  
 for (xA, yA, xB, yB) in nms\_box:  
 widths.append(xB - xA)  
 xC = xA + int((xB - xA) / 2)  
 yC = yA + int((yB - yA) / 2)  
 centers.append((xC, yC))  
 cv2.rectangle(frame, (xA, yA), (xB, yB), (0, 255, 0), 2)  
 cv2.circle(frame, (xC, yC), 15, (120, 0, 0), 2)  
 cv2.putText(frame, f'person {person}', (xA, yA - 5), cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, (255, 255, 255), 2)  
 person += 1  
  
 cv2.putText(frame, 'Status : Detecting', (5, 25), cv2.FONT\_HERSHEY\_DUPLEX, 0.6, (0, 0, 0), 3)  
 cv2.putText(frame, 'Status : Detecting', (5, 25), cv2.FONT\_HERSHEY\_DUPLEX, 0.6, (255, 255, 255), 1)  
  
 cv2.putText(frame, f'Total Persons : {person - 1}', (5, 55), cv2.FONT\_HERSHEY\_DUPLEX, 0.6, (0, 0, 0), 3)  
 cv2.putText(frame, f'Total Persons : {person - 1}', (5, 55), cv2.FONT\_HERSHEY\_DUPLEX, 0.6, (0, 200, 0), 1)  
  
 violations = []  
 averageWidths = average(widths)  
 distance = averageWidths \* 2  
 print("Distance: ", distance)  
 violation\_counter = 0  
 for i, p1 in enumerate(centers):  
 for j, p2 in enumerate(centers[i + 1:]):  
 if (np.linalg.norm(np.array(p2) - np.array(p1))) < distance:  
 if abs(p1[1] - p2[1]) < distance // 2:  
 cv2.line(frame, p1, p2, (0, 0, 255), thickness=3, lineType=3)  
 violations.append((i, j + i + 1))  
  
 # detect violations and draw red bounding boxes  
 for (i, j) in violations:  
 (xA, yA, xB, yB) = nms\_box[i]  
 cv2.rectangle(frame, (xA, yA), (xB, yB), (0, 0, 255), 2)  
 (xA, yA, xB, yB) = nms\_box[j]  
 cv2.rectangle(frame, (xA, yA), (xB, yB), (0, 0, 255), 2)  
 violation\_counter += 1  
  
 cv2.putText(frame, f'Social distance violations : {violation\_counter}', (5, 85), cv2.FONT\_HERSHEY\_DUPLEX, 0.6,  
 (0, 0, 0), 3)  
 cv2.putText(frame, f'Social distance violations : {violation\_counter}', (5, 85), cv2.FONT\_HERSHEY\_DUPLEX, 0.6,  
 (0, 0, 255), 1)  
  
 cv2.imshow('output', frame)  
 return frame  
  
  
def detectByPathVideo(path):  
 video = cv2.VideoCapture(path)  
 check, frame = video.read()  
 if check == False:  
 print('Video Not Found. Please Enter a Valid Path (Full path of Video Should be Provided).')  
 return  
  
 print('Detecting people...')  
 while video.isOpened():  
 check, frame = video.read()  
  
 if check:  
 frame = imutils.resize(frame, width=min(800, frame.shape[1]))  
 frame = detect(frame)  
  
 key = cv2.waitKey(1)  
 if key == ord('q'):  
 break  
 else:  
 break  
 video.release()  
 cv2.destroyAllWindows()  
  
  
path = "test.mp4"  
HOGCV = cv2.HOGDescriptor()  
HOGCV.setSVMDetector(cv2.HOGDescriptor\_getDefaultPeopleDetector())  
detectByPathVideo(path)

**SONUÇLAR**

Bu proje, bir video kaydındaki insanların sosyal mesafe kuralına uyma eğilimlerini açıkça ortaya sermektedir. Bu yazılım, sokak kameralarına entegre edilip Covid-19 salgınıyla savaşma sürecinde halkın bu konudaki duyarlılığı ölçülerek bu konuda çalışmalar yapılabilir. Bu da salgınla savaş sürecini olumlu etkiler.

Bu proje takım çalışması, araştırma ve nesne tabanlı programlama yaklaşımı ile yazılım geliştirme becerilerimizi geliştirmiştir.

**PROJE EKİBİ**

Evrim Arda KALAFAT, 25.09.2001 yılında istanbulda doğdu. 2019 yılında Kadıköy Final Temel Lisesi’nden mezun oldu. Şu anda Fenerbahçe Üniversitesi Bilgisayar Mühendisliği bölümünde lisans eğitimi almakta. Bilgisayar Mühendisliği Bölüm Temsilciği yapmaktadır. C, C++ ve Pyhton dillerinde bilgili. Programlama, yapay zeka ve siber güvenlik ile ilgileniyor.

Arda ALHAN, 18.05.2001 yılında doğdu. 2019 yılında Eyüp Anadolu Lisesi’nden mezun oldu. Şu an da Fenerbahçe Üniversitesi Bilgisayar Mühendisliği bölümünde Lisans eğitimi almakta. C, C#, Python dilleriyle ve Unity oyun motoruyla ilgileniyor.

Doğa TURAN, 18.09.2001 yılında Gölcük’te dünyaya geldi. 2019 yılında Şehit Özcan Kan Fen Lisesi’nden mezun oldu. Fenerbahçe Üniversitesi’nde Endüstri Mühendisliği anadal ve Bilgisayar Mühendisliği ikinci ana dal olmak üzere lisans eğitimi almaktadır. C ve Pyhton dilleri hakkında donanımlıdır.

**REFERANS DOSYALAR**

<https://github.com/rhgod/Social-Distance-Tracker>

<https://www.youtube.com/watch?v=zXZrWz8jATM&t=12s>

**KAYNAKLAR**

<https://docs.opencv.org/master/d6/d00/tutorial_py_root.html>

<https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c>

<https://github.com/jrosebr1/imutils/blob/master/imutils/object_detection.py>