

# İSTANBUL TEKNİK ÜNİVERSİTESİ

## ÖĞRENCİ STAJ RAPORU STUDENT INTERN REPORT

İTÜ



FAKÜLTE FACULTY

BÖLÜM DEPARTMENT

ÖĞRENCİNİN STUDENT

Adı Soyadı *Name Surname*

Numarası *Student Number*

Staj Komisyonu Başkanı

*Head of Internship Commission*

STAJ BİLGİLERİ *INTERNSHIP INFORMATION*

Staj Türü *Internship Type*

Staj Yeri *Internship Location*

Staj Başlama Tarihi *Internship Start Date*

Staj Bitiş Tarihi *Internship End Date*

Staj Gün Sayısı *Internship Duration (days)*

## **RAPOR İÇERİĞİ**

Bu rapor, *Staj Projesi ve HAVELSAN Genç Yıldızlar Yarışması* başlığı altında açıklaması yapılan projenin geliştirilmesinde yer alan matematiksel ve yazılımsal arka planı parça parça inceler; her bir parçanın pratik gerçeklemesini algoritma ve/veya kod parçacıkları ile çözümler.

### **0- HAVELSAN ve ÇEVİRİM İÇİ STAJ HAKKINDA**

- a. Şirket Profili
- b. Organizasyon Şeması
- c. E-Öğrenme Platformu

### **1- STAJ PROJESİ ve HAVELSAN GENÇ YILDIZLAR YARIŞMASI**

- a. Staj Projesi
- b. HAVELSAN GYY

### **2- GÖRÜNTÜ İŞLEME TEMELLERİ**

- a. OpenCV Matris Düzeni
- b. Histogram ve Thresholding
- c. Kenar Algılama ve Kontür Çizimi
- d. Renk Ayırma (Ten Rengi Algılama)

### **3- DERİN ÖĞRENME: KONVOLÜSYONEL (EVİRİŞİMLİ) SİNİR AĞLARI**

- a. Derin Öğrenme İhtiyacı
- b. Veri Seti Araştırması ve Üretilmesi
- c. Veri Ön İşlemesi ve Veri Seti Boyutları
- d. Ön İşlenmiş Görüşeller ve Etiketlerine Bakış
- e. Kanal Boyutu Düzenlemesi
- f. Tensorflow Kütüphanesi ile Model Oluşturma
- g. ‘Overfitting’den Kaçınma Yöntemleri
- h. Modelin Derlenmesi ve Eğitimi
- i. Sonuçların İncelenmesi
- j. Daha Gelişmiş bir Model: *Transfer Learning*
- k. EKSTRA: *Mobile-Net-v2* Nesne Sınıflandırıcı

### **4- GERÇEK ZAMANLI UYGULAMANIN GELİŞTİRİLMESİ ve CANLI ÖĞRENME**

- a. Program Ana Döngüsü
- b. Kullanıcı Arayüzü
- c. Canlı Öğrenme

İçerikte bahsi geçen tüm proje dosyaları açık kaynak olarak İTÜ Kovan bulut hesabında paylaşılmaktadır. Ayrıca ilerleyen sayfalarda ayrıntılarına yer verilecek HAVELSAN Genç Yıldızlar Yarışması’nda üçüncülük elde eden projenin tanıtım ve demosunun yer aldığı video da Youtube üzerinde liste dışı olarak yayınlanmaktadır. İlgili linkler:

- [kovan.ITU.edu.tr/index.php/s/EYs38GfD43LHsrX](http://kovan.ITU.edu.tr/index.php/s/EYs38GfD43LHsrX)
- [youtu.be/K69YdKOAW74](https://youtu.be/K69YdKOAW74)

Yapılan Uygulama :	<b>HAVELSAN ve ÇEVİRİM İÇİ STAJ HAKKINDA</b>	Tarih: 06/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

#### a. Şirket Profili

HAVELSAN, Türk Silahlı Kuvvetlerini Güçlendirme Vakfı'nın bir şirketi olarak 1982 yılında kurulmuştur.

Türkiye'nin en büyük teknoloji firmalarından biri olarak kabul edilen HAVELSAN; deneyimi, uzman çalışanları, ileri teknolojiye dayalı yoğun özgün çözüm ve ürünlerile uluslararası pazarda lider bir markadır.

HAVELSAN; bünyesinde geliştirilen yüksek teknoloji ve yazılımların yanı sıra savunma, güvenlik ve bilişim sektörlerinde teknoloji üreten ve iş ekosisteminde yer alan firmaların çözüm ve ürünlerini de bir araya getirerek müşterilerine anahtar teslim çözümler sunan bir şirkettir.

Yazılım yoğun sistemler konusunda faaliyet gösteren HAVELSAN;

- Komuta Kontrol ve Savunma Teknolojileri
  - Eğitim ve Simülasyon Teknolojileri
  - Bilgi ve İletişim Teknolojileri
  - Ülke Güvenliği ve Siber Güvenlik Çözümleri

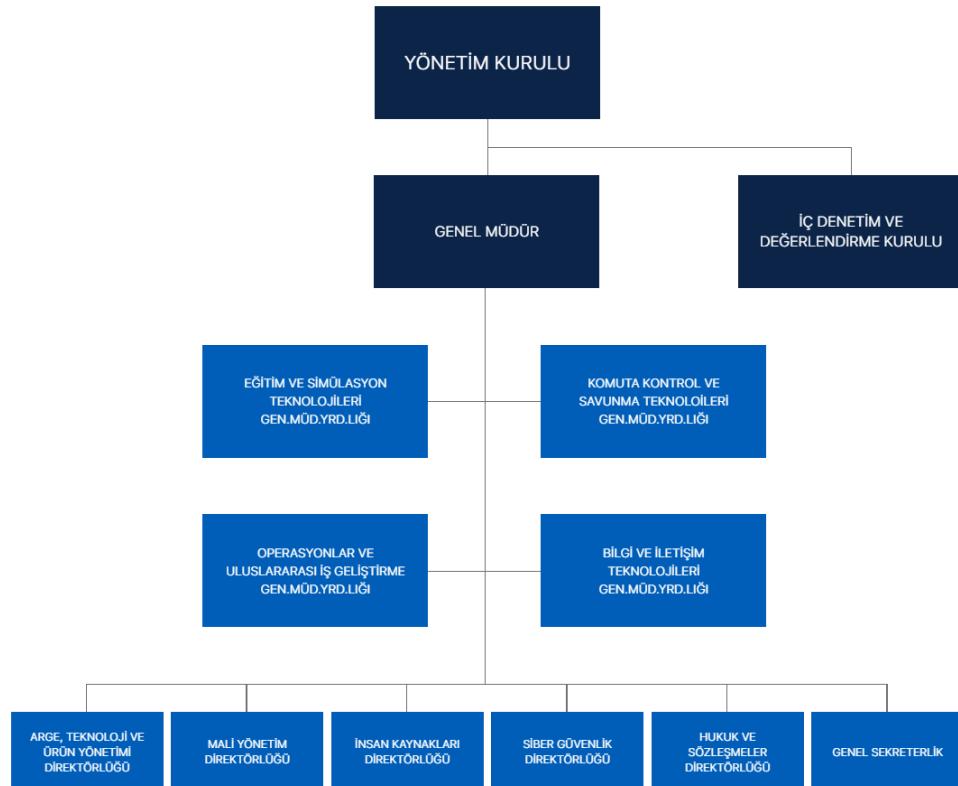
alanlarında özgün ürün ve sistemlerden oluşan çözümlerini Türk Silahlı Kuvvetlerinin (TSK), kamu kurum ve kuruluşlarının, özel sektörün ve uluslararası müşterilerinin hizmetine sunmaktadır.

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
--	------	-------------------------

Yapılan Uygulama :	<b>HAVELSAN ve ÇEVİRİM İÇİ STAJ HAKKINDA</b>	Tarih: 07/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

### b. Organizasyon Şeması

Şirket organizasyon şeması şekildeki gibidir.



ÖĞRENCİ İmza <i>Cobri</i>	ONAY	FİRMA YETKİLİSİ İmza
---------------------------------	------	-------------------------

*Cobri*

Yapılan Uygulama :	<b>HAVELSAN ve ÇEVİRİM İÇİ STAJ HAKKINDA</b>	Tarih: 08/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

### c. E-Öğrenme Platformu

Şahsimin ve HAVELSAN A.Ş. şirketinin ilk defa deneyimlediği çevrim içi staj sürecinde, şirket tarafından stajyerlere eğitim vermek için E-Öğrenme Platformu hazırlanmıştı ve biz stajyerlerden verilen kullanıcı hesaplarıyla giriş yaparak atanan eğitimleri tamamlamamız bekleniyordu.

Eğitimler iki temel kısımdan oluşmaktadır:

- Yasal zorunluluk olan İş Güvenliği Eğitimi
- Stajyer farkındalık eğitimleri

Stajyer farkındalık eğitimlerini örneklendirecek olursak,

Sosyal ağların, mobil cihazların, webin güvenli kullanımı; veri, elektronik posta, parola güvenliği, siber güvenlik ve siber tehditler, veri bilimi ve analizi, otonom sistemler gibi teknik derinlik içermeyen birçok eğitim yer almaktaydı.

Bunun dışında Linux tabanlı yerli işletim sistemi Pardus kullanımını da teşvik edici eğitimler bulunuyordu.

Yukarıda belirtilen içerikler genel olarak staj süresinin tamamına yayılmış olsa da aynı zamanda staj projesinin belirleneceği ilk günlerde yani benim isinma dediğim dönemde daha yoğun bir şekilde işlendi. Bu sürecin biraz da sıkıcı geçtiğini söyleyebilirim.

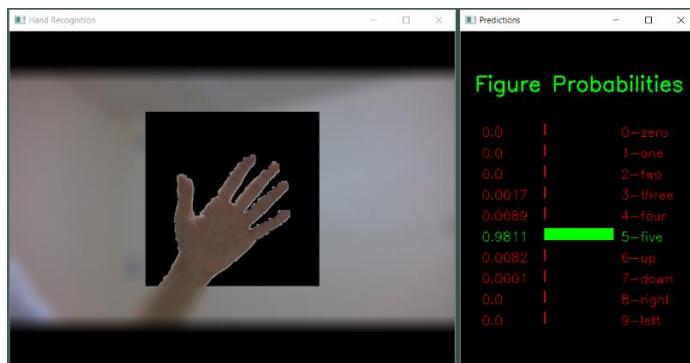
ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
--	------	-------------------------

Yapılan Uygulama :	<b>STAJ PROJESİ ve HAVELSAN GENÇ YILDIZLAR YARIŞMASI</b>	Tarih: 09/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

### a. Staj Projesi

Statik el hareketlerinin algılanması ve konvolüsyonel sinir ağları modeli eğitilerek sınıflandırılması

2019-2020 Bahar yarıyılında, Makine Öğrenmesinin temel teorisini edindiğim EHB420E Artificial Neural Networks dersinde, dönem projesi olarak herhangi bir Deep Learning kütüphanesinden gerçeklediğim Multi-Layer Perceptron modeliyle bilgisayara rakamları tanıtmak istemiştim. Staj döneminde ise Konvolüsyonel Sinir Ağları üzerine çalışmak istedim. Bu doğrultuda statik el hareketlerinin algılanması ve sınıflandırılmasının uygun bir proje olabileceğine kanaat getirdim.



### b. HAVELSAN GYY

Şirket, stajyerlerinden HAVELSAN ürünlerine eklenebilecek yenilikçi özellikler veya HAVELSAN'da geliştirilmesini hayal ettikleri yeni bir proje ile yarışmaya katılmalarını bekliyordu. Takım arkadaşım ile staj projemizi nasıl daha inovatif hale getirebiliriz sorusunu üzerine düşünerek, temas etmeden yalnızca kameralara gösterilen el hareketleri ile kontrol edilebilen bir otomat geliştirilmesi projesinde karar kıldı.

5 Ağustos'ta gerçekleşen final sunumuyla beraber yarışma sonuçlandı ve üçüncülük elde ettik.

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
---------------------	------	-------------------------

Yapılan Uygulama :	<b>STAJ PROJESİ ve HAVELSAN GENÇ YILDIZLAR YARIŞMASI</b>	Tarih: 10/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	



### Proje Açıklaması:

Son dönemde dünyayı etkisi altına alan Koronavirüs pandemisi sebebiyle insanlık, virus ile adeta savaşa girdi. Virüsün en büyük silahı ise temas. Öyleyse, bu savaşı kazanmak için temassız bir hayat sürdürmemiz gerekmekte.

Her gün onlarca noktaya temasta bulunuyoruz. Yiyecek otomatları da bunlardan biri. Teması sıfır indirmek için otomatları dokunmadan kontrol edebileceğimiz bir sistem gereklidir.

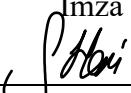
Yapay sinir ağları ile bilgisayarlar insanların el hareketlerini algılama yeteneği kazandılar. Yiyecek otomatları da bu yeteneğe sahip olursa insanlar sıfır temas ile istedikleri ürünü erişebilirler.

### Takım Üyeleri

- Selahaddin HONİ
- Yasin SOYLU

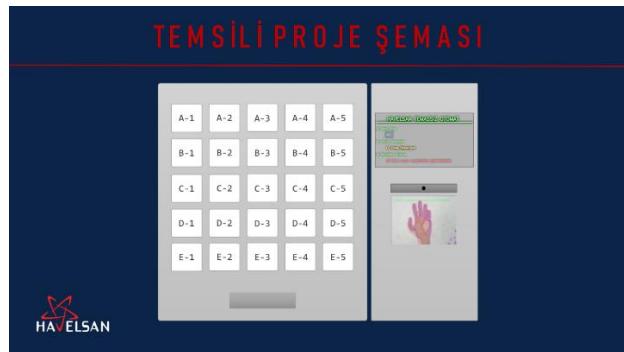
\*\*\*

Açıklaması verilen projenin gerçekleme adımları ikinci, üçüncü ve dördüncü ana başlıklar altında parka parça incelenecaktır. Bir sonraki sayfada final sunumunda gösterimi yapılan demo ve şema görselleri yer almaktadır.

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
--	------	-------------------------

Yapılan Uygulama :	<b>STAJ PROJESİ ve HAVELSAN GENÇ YILDIZLAR YARIŞMASI</b>	Tarih: 10/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

Final sunumunda gösterimi yapılan,  
Proje Şeması:



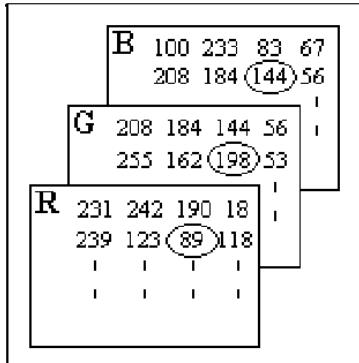
Final Demosu Ekran Görüntüsü:



ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
---------------------	------	-------------------------

Yapılan Uygulama :	<b>GÖRÜNTÜ İŞLEME TEMELLERİ</b>	Tarih: 13/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

### a. OpenCV Matris Düzeni



OpenCV cv2.imread(dosya\_yolu, {0 veya 1}) komutu ile belirtilen dosya yolundaki görsel dosyasını, GRAY-SCALE (0) veya RENKLİ (1) olarak Python ortamına almaktadır. İçeri alınan görsel Image Container adlı bir matris ile temsil edilmekte olup elemanları 0-255 arasında değişmektedir.

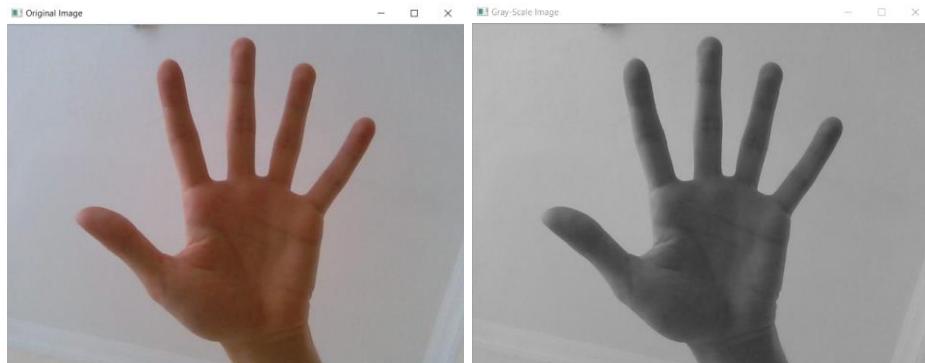
RENKLİ bir görsel için BLUE, GREEN ve RED kanalları olmak üzere 3 ayrı boyut; GRAY-SCALE bir görsel içinse yalnızca 1 boyut bulunmaktadır.

### b. Histogram ve Thresholding

OpenCV ile el fotoğrafını Python ortamına alalım ve GRAY-SCALE'e çevirelim.

```
import cv2
from matplotlib import pyplot as plt

# get the image into the environment
img1 = cv2.imread('el.jpg',1)
# convert to gray-scale
img0 = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)
# show images
cv2.imshow("Original Image", img1)
cv2.imshow("Gray-Scale Image", img0)
```



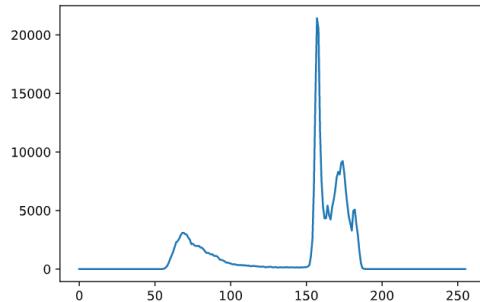
ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
---------------------	------	-------------------------

Yapılan Uygulama :	<b>GÖRÜNTÜ İŞLEME TEMELLERİ</b>	Tarih: 13/07/2020
Uygulama Birimi:	<b>EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ</b>	

Histogramda 0-255 arası renk değerlerinin çerçevedeki frekans dağılımını görmekteyiz.

```
# find frequency of pixels in range 0-255
histr = cv2.calchist([img0],[0],None,[256],[0,256])
# show the histogram
plt.plot(histr)
```

[<matplotlib.lines.Line2D at 0x21710a334c0>]



```
thresh150 = cv2.threshold(img0, 150, 255, cv2.THRESH_BINARY_INV)[1]
cv2.imshow("Thresh-150", thresh150)
plt.show()
```

Histogram incelendiğinde 150-200 renk değerleri arasında bir nesne olduğunu anlayabiliyoruz. Aynı şekilde 50-100 arasında da istenmeyen bir gürültü bulunmaktadır. Eşik değeri olarak 150 alındığında el maskesini elde edebilmekteyiz.



ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
---------------------	------	-------------------------

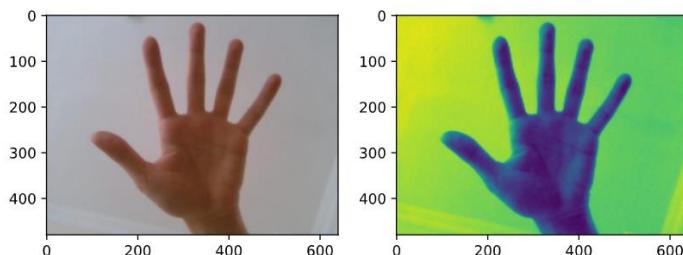
Yapılan Uygulama :	<b>GÖRÜNTÜ İŞLEME TEMELLERİ</b>	Tarih: 14/07/2020
Uygulama Birimi:	<b>EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ</b>	

### c. Kenar Algılama ve Kontür Çizimi

Aynı el resmini OpenCV ile okuyalım; ancak bu sefer görselleri Matplotlib ile yazdıralım. OpenCV BGR düzeni ile çalışırken Matplotlib RGB konteyner ile işlem yapmaktadır. Öyleyse, bu dönüşümü cv2.COLOR\_BGR2RGB parametresi ile gerçekleştirelim.

```
img1 = cv2.imread("el.jpg", 1)
img0 = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)
img1 = cv2.cvtColor(img1, cv2.COLOR_BGR2RGB)

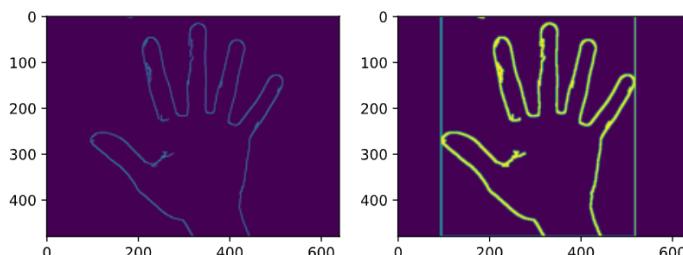
fig=plt.figure(figsize=(8, 8))
fig.add_subplot(1,2,1)
plt.imshow(img1)
fig.add_subplot(1,2,2)
plt.imshow(img0)
plt.show()
```



Canny filtresi ile GRAY-SCALE görsel üzerinden kenar algılaması yapalım; boundingRect() ile çevreleyen dörtgeni çizdirip dilate ile nesnenin kenarlarını belirginleştirelim.

```
img_edge = cv2.Canny(img0, 50, 100)
fig=plt.figure(figsize=(8, 8))
fig.add_subplot(1,2,1)
plt.imshow(img_edge)

img_edge = cv2.dilate(img_edge, None, iterations=2)
x,y,w,h = cv2.boundingRect(img_edge)
cv2.rectangle(img_edge, (x,y), (x+w, y+h), (240, 0, 159), 2)
fig.add_subplot(1,2,2)
plt.imshow(img_edge)
plt.show()
```



ÖĞRENCİ İmza	ONAY	FİRMA YETKİLİSİ İmza

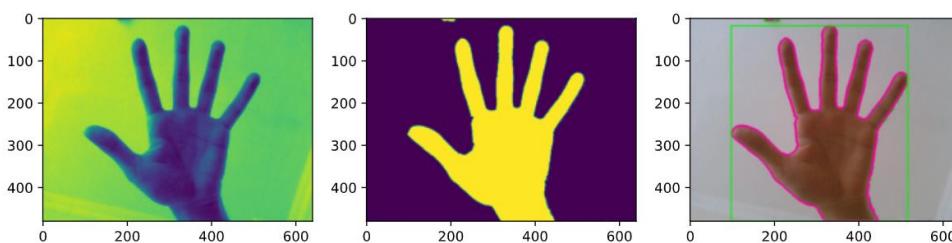
Yapılan Uygulama :	<b>GÖRÜNTÜ İŞLEME TEMELLERİ</b>	Tarih: 16/07/2020
Uygulama Birimi:	<b>EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ</b>	

OpenCV ve Imutils kullanarak nesneleri çevreleyen kontürleri maske üzerinden almak mümkündür. (Maske eldesi için histogram incelenerek eşik değeri 150 seçilmiştir.)

```
thresh = cv2.threshold(img0, 150, 255, cv2.THRESH_BINARY_INV)[1]
cnts = cv2.findContours(thresh, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
cnts = imutils.grab_contours(cnts)

for c in cnts:
    cv2.drawContours(img1, [c], -1, (240, 0, 159), 3)
    x,y,w,h = cv2.boundingRect(c)
    cv2.rectangle(img1, (x,y), (x+w, y+h), (0, 240, 0), 2)

fig=plt.figure(figsize=(12, 12))
fig.add_subplot(1,3,1)
plt.imshow(img0)
fig.add_subplot(1,3,2)
plt.imshow(thresh)
fig.add_subplot(1,3,3)
plt.imshow(img1)
plt.show()
```



İlk sırada GRAY-SCALE değerlerin Matplotlib tarafından görselleştirilmesi; ikinci sırada eşik değeriyle maskelenmiş el görülmektedir. Son sırada ise elin çevresinde kontür çizimiyle birlikte, kontürün maksimum değerleriyle örülmüş bir dörtgen de bulunmaktadır.

#### d. Renk Ayırma (Ten Rengi Algılama)

Renkleri ayırmak için BGR yerine HSV boyutlarıyla çalışmak daha verimli sonuçlar vermektedir. (Elektromagnetikte farklı koordinat sistemleri kullanılması gibi.) Ten rengi için kabul edilecek minimum ve maksimum değerler SKIN\_LO ve SKIN\_HI sabitleriyle belirlenir.

```
import cv2
import matplotlib.pyplot as plt
import numpy as np

# Skin Detection Color Thresholds in HSV
SKIN_LO = np.array([0,48,80], dtype=np.uint8)
SKIN_HI = np.array([20,255,255], dtype=np.uint8)
```

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
---------------------	------	-------------------------

Yapılan Uygulama :	<b>GÖRÜNTÜ İŞLEME TEMELLERİ</b>	Tarih: 17/07/2020
Uygulama Birimi:	<b>EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ</b>	

```

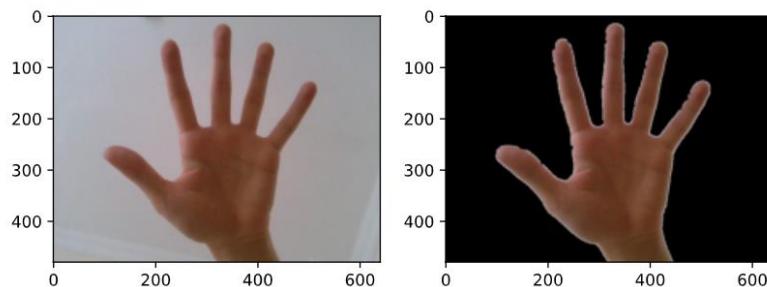
img_bgr = cv2.imread("el.jpg", 1)
img_rgb = cv2.cvtColor(img_bgr, cv2.COLOR_BGR2RGB)

fig = plt.figure(figsize=(8,8))
fig.add_subplot(1,2,1)
plt.imshow(img_rgb)

img_hsv = cv2.cvtColor(img_bgr, cv2.COLOR_BGR2HSV)
skinMask = cv2.inRange(img_hsv, SKIN_LO, SKIN_HI)
skinMask = cv2.dilate(skinMask, None, iterations=1)
skinMask = cv2.GaussianBlur(skinMask, (3, 3), 0)
skin     = cv2.bitwise_and(img_rgb, img_rgb, mask= skinMask)

fig.add_subplot(1,2,2)
plt.imshow(skin)
plt.show()

```



- Belirlenen eşik değerleri doğrultusunda maske oluşturulur.
- Gürültüyü azaltmak amaçlı *dilation* ve *GaussianBlur* uygulanır.
- *bitwise\_and()* işleminde üretilen maske kullanıldığında istenen renkteki nesnenin dışında kalan arka planın kaldırıldığını ikinci resimde görebilmekteyiz.

Görüntü İşleme Temelleri başlığı altında verilen kodlar, statik el hareketlerinin algılanması, temassız otomat hatta *Mobile-Net-v2* nesne sınıflandırıcı projelerinin yapıtaşlarını oluşturmaktadır. Konvolüsyonel sinir ağları açıklanırken aynı kodlar üzerine geri dönülmeyecektir.

ÖĞRENCİ İmza	ONAY	FİRMA YETKİLİSİ İmza

Yapılan Uygulama :	<b>DERİN ÖĞRENME: KONVOLÜSYONEL SİNİR AĞLARI</b>	Tarih: 20/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

### a. Derin Öğrenme İhtiyacı

Bir önceki ana başlık altında, OpenCV'nin görüntü işleme temel fonksiyonlarına değindim. Bu fonksiyonlar ile basit problemlere çözüm getirebilirken, problemin karmaşıklığı arttığında OpenCV yetersiz kalabilmektedir. Bu tür durumlarda makine öğrenmesine başvurulmaktadır.

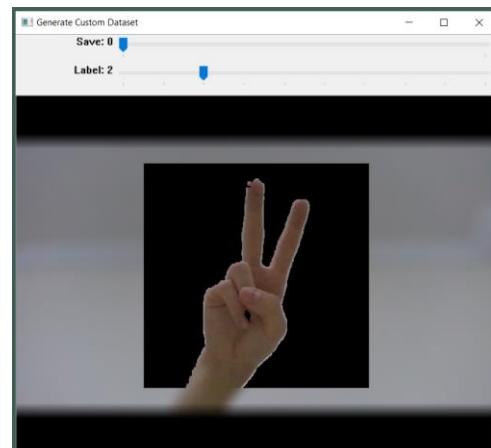
Bir mühendis olarak, karşılaşılan sorunlara cevap ararken, daima maliyeti de göz önünde bulundurmamalıyız. Örneğin, bilgisayarın şekilleri sınıflandırmamasını istiyorsak, bu görev için derin öğrenmenin kullanılması zaman ve donanım israfına mâl olacaktır. Eğitilen modelin koşturulacağı elektronik kartın saniyede yüzlerce konvolüsyon işlemi gerçekleştirecek olması kartın seçiminde daha pahalı modellere yöneltmemize sebep olabilmektedir. (Raspberry Pi 3 yerine Nvidia Jetson gibi)

El hareketlerinin sınıflandırılması derin öğrenme ihtiyacı duyulan bir problemdir. Bu sebeple konvolüsyonel sinir ağları modeli eğitererek çözüm arayacağız.

### b. Veri Seti Araştırması ve Üretilmesi

Statik el hareketlerinin algılanması üzerine başlayan proje, devamında otomatların temassız kontrolü inovasyonuyla güncellenince, istenilen el hareketlerini içeren veri setleri üzerinde internette araştırma yapıldı. Proje isterlerini karşılaşacak verimde etiketli görseller bulunamayınca, yeni bir veri setinin üretilmesine karar verildi.

Bilgisayar web kamerasından her 0.1 saniyede bir çerçeve yakalayıp eli algıladıktan sonra arka planı kaldırarak belirtilen etiket klasörüne kaydeden bir program yazıldı. Bu program ile 10 farklı el figürü için 200'er tane toplamda 2000 görsel üretildi.



Görüntü işleme temellerinde *El (Ten Rengi) Algılanması* algoritması paylaşıldığından bu programın ayrıntılı açıklamalarına girilmeyecektir.

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
--	------	-------------------------

Yapılan Uygulama :	<b>DERİN ÖĞRENME: KONVOLÜSYONEL SİNİR AĞLARI</b>	Tarih: 21/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

### c. Veri Ön İşlemesi ve Veri Seti Boyutları

Ön işleme sırasında, görsellerin boyutunun düzenlenmesi –resize 224x224 piksel; görsellerin matris değerlerinin 0 ile 1 arasına oranlanması –normalize; etiketlerin kodlanması ve veri setinin eğitim ve test kümelerine ayrılması gibi adımlar bulunmaktadır.

Konvolüsyonel sinir ağı modelini eğitebilmemiz için öncelikle veri setinin Python ortamına alınması gerekmektedir. Bunu ve yukarıda belirtilen işlemleri gerçekleştiren Dataset() sınıfı yazıldı. Aşağıda bu sınıfın sadeleştirilmiş yapısı görülmektedir.

```
class Dataset():
    def __init__(self, path, bgr=0):
        self.imgs = list()
        self.labels = list()
        self.labels_one_hot = list()
        self.x_train = list()
        self.y_train = list()
        self.x_valid = list()
        self.y_valid = list()
    def load_dataset(self, path):
        # Loads the dataset from directory via given path
        # COLORFUL or GRAY-SCALE according to 'bgr' parameter
        # resizes the image (224x224) px
        pass
    def normalize(self):
        # divides the image values by 255
        pass
    def shuffle(self):
        # mixing
        pass
    def add_channel_dim(self):
        # adds channel dimension for GRAY-SCALE images
        pass
    def one_hot_encoding(self):
        pass
    def split_data(self, train_valid_ratio):
        # splitting dataset for validation for a given ratio
        # stores the seperated dataset in x-y_train etc. lists
        pass
```

Bir sonraki sayfada, yapısı verilen bu Dataset() sınıfıyla ön işleme süreci anlatılacaktır.

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
--	------	-------------------------

Yapılan Uygulama :	<b>DERİN ÖĞRENME: KONVOLÜSYONEL SİNİR AĞLARI</b>	Tarih: 21/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

Dataset() sınıfı nesnesi 'ds' nin metodlarıyla veriler içeri aldı ve ön işlemesi gerçekleştirildi. Veri setimizde 2000 farklı görsel ve etiketi bulunmaktadır. Her bir görselin boyutu 224x224 olup kanal boyutu değerinin olmadığına dikkat edelim. Kodlanmış etiketlerin boyutu ise görsellerin ayrıldığı kategori sayısına yani 10'a eşittir.

```
# Loading Dataset & Preprocess via custom Dataset() class
#=====
# Dataset Images are stored in the Dataset object as ds.imgs;
# Labels as ds.labels; One-hot-Encoding as ds.labels_one_hot
print("(!) DATA Preprocessing")
TR_start= timeit.default_timer()
path = "./dataset/custom_dataset/"
color = 0 # 0: Gray 1: BGR
print("Dataset is loading..")
ds = Dataset(path, color)
print("Rescaling images to 1..")
ds.normalize()
print("Shuffle process..")
ds.shuffle()
print("One-hot-encoding..")
ds.one_hot_encoding()
print("Shape of image-set:")
print(ds.imgs.shape)
print("Shape of one-input-img:")
print(ds.imgs[0].shape)
print("Shape of label-set:")
print(ds.labels.shape)
print("Shape of one-hot-label-set:")
print(ds.labels_one_hot.shape)
# End of Data Preprocessing
TR_stop = timeit.default_timer()
print(f"(~) SUCCESSFULL | Execution Time: {TR_stop-TR_start:.3f} sec")

(!) DATA Preprocessing
Dataset is loading..
Rescaling images to 1..
Shuffle process..
One-hot-encoding..
Shape of image-set:
(2000, 224, 224)
Shape of one-input-img:
(224, 224)
Shape of label-set:
(2000,)
Shape of one-hot-label-set:
(2000, 10)
(~) SUCCESSFULL | Execution Time: 5.556 sec
```

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
--	------	-------------------------

Yapılan Uygulama :	<b>DERİN ÖĞRENME: KONVOLÜSYONEL SİNİR AĞLARI</b>	Tarih: 22/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

Veri setini eğitim ve validation kümelerine ayıriyoruz. Modelin eğitimi sırasında validation kümesi görselleri dışında tutulacak; yalnızca modelin tahmin etme başarısının testi için kullanılacaktır.

```
# Splitting Dataset for Training
=====
print("Dataset splitting..")
ds.split_data(0.2)
print("Size of dataset:")
print(len(ds.imgs))
print("Size of training set:")
print(f"{len(ds.x_train)}")
print("Size of validation set:")
print(f"{len(ds.x_valid)}", end="\n\n")

assert len(ds.imgs) == len(ds.y_train)+len(ds.y_valid)

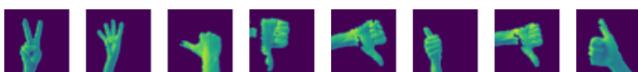
Dataset Splitting..
Size of dataset:
2000
Size of training set:
1600
Size of validation set:
400
```

#### d. Ön İşlenmiş Görüşeller ve Etiketlerine Bakış

```
# Visualize Training Images
=====
def plotImages(images_arr):
    fig, axes = plt.subplots(1, 8)
    axes = axes.flatten()
    for img, ax in zip( images_arr, axes):
        ax.imshow(img)
        ax.axis('off')
    plt.tight_layout()
    plt.show()

print(f"First 8 labels of dataset: {ds.labels[:8]}")
plotimages(ds.imgs[:8])
print(f"One-hot-encoded labels:")
for i in range(8):
    print(f"{ds.labels_one_hot[i]}")
```

First 8 labels of dataset: [2 4 9 7 7 6 7 6]



```
One-hot-encoded labels:
[0. 0. 1. 0. 0. 0. 0. 0.]
[0. 0. 0. 1. 0. 0. 0. 0.]
[0. 0. 0. 0. 0. 0. 0. 1.]
[0. 0. 0. 0. 0. 0. 1. 0.]
[0. 0. 0. 0. 0. 0. 1. 0.]
[0. 0. 0. 0. 0. 0. 1. 0.]
[0. 0. 0. 0. 0. 0. 1. 0.]
[0. 0. 0. 0. 0. 0. 1. 0.]
```

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
---------------------	------	-------------------------

Yapılan Uygulama :	<b>DERİN ÖĞRENME: KONVOLÜSYONEL SİNİR AĞLARI</b>	Tarih: 23/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

### e. Kanal Boyutu Düzenlemesi

GRAY-SCALE görseller için varsayılan olarak kanal boyutu bulunmaz; ancak bu durum, eğiteceğimiz modelin giriş şekliyle (input shape) uyum sağlamaz. Kanal boyutunun 1 olduğunu belirten ekleme yapılır.

```
# Reshaping the Image Set for the Input Layer of the Network
#=====
# Image shape is not convenient for the model input layer dimensions
# We need 1-more dimension representing channel length which is 1
# print(ds.imgs.shape)      # (-1, H, W)
if not color:
    ds.add_channel_dim()   # (-1, H, W, 1)
    print("Channel Dimension is added for GRAY-SCALE image")
    print(ds.imgs[0].shape, end="\n\n")
    IMG_H, IMG_W, CHANNEL = ds.imgs[0].shape

Channel Dimension is added for GRAY-SCALE image
(224, 224, 1)
```

### f. Tensorflow Kütüphanesi ile Model Oluşturma

Her konvolüsyon katmanında derinlik ve beraberinde parametre sayısı artmakta, *MaxPooling()* ile boyutlar daha anlamlı olan parametrelere indirgenmektedir. *Dense()* katmanları ise modelin, farklı el ifadelerinin görsel karakteristiklerini, öğrenmesini kolaylaştırmaktadır.

```
# Creating a Sequential CNN Model
#=====
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Conv2D, Flatten, Dropout, MaxPooling2D

model = Sequential([
    Conv2D(16, 5, padding='same', activation='relu', input_shape=(IMG_H, IMG_W, CHANNEL)),
    MaxPooling2D(),
    Conv2D(32, 5, padding='same', activation='relu'),
    MaxPooling2D(),
    Conv2D(32, 3, padding='same', activation='relu'),
    MaxPooling2D(),
    Conv2D(64, 3, padding='same', activation='relu'),
    MaxPooling2D(),
    Conv2D(64, 2, padding='same', activation='relu'),
    MaxPooling2D(),
    Conv2D(64, 2, padding='same', activation='relu'),
    MaxPooling2D(),
    Flatten(),
    Dropout(0.3),
    Dense(512, activation="relu"),
    Dropout(0.3),
    Dense(10)
])
model.summary()
```

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
--	------	-------------------------

Yapılan Uygulama :	<b>DERİN ÖĞRENME: KONVOLÜSYONEL SİNİR AĞLARI</b>	Tarih: 24/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

Model özetini incelediğimizde eğitimde 374,442 parametrenin kullanılacağını görüyoruz.

```
Model: "sequential"
=====
Layer (type)          Output Shape         Param #
conv2d (Conv2D)        (None, 224, 224, 16)    416
=====
max_pooling2d (MaxPooling2D) (None, 112, 112, 16)    0
conv2d_1 (Conv2D)        (None, 112, 112, 32)    12832
max_pooling2d_1 (MaxPooling2D) (None, 56, 56, 32)    0
conv2d_2 (Conv2D)        (None, 56, 56, 32)    9248
max_pooling2d_2 (MaxPooling2D) (None, 28, 28, 32)    0
conv2d_3 (Conv2D)        (None, 28, 28, 64)    18496
max_pooling2d_3 (MaxPooling2D) (None, 14, 14, 64)    0
conv2d_4 (Conv2D)        (None, 14, 14, 64)    16448
max_pooling2d_4 (MaxPooling2D) (None, 7, 7, 64)    0
conv2d_5 (Conv2D)        (None, 7, 7, 64)    16448
max_pooling2d_5 (MaxPooling2D) (None, 3, 3, 64)    0
flatten (Flatten)       (None, 576)           0
dropout (Dropout)       (None, 576)           0
dense (Dense)          (None, 512)           295424
dropout_1 (Dropout)     (None, 512)           0
dense_1 (Dense)         (None, 10)            5130
=====
Total params: 374,442
Trainable params: 374,442
Non-trainable params: 0
```

#### g. ‘Overfitting’den Kaçınma Yöntemleri

##### Dropout:

Modelin Sequential tasarımı sırasında bazı katmanlar arasına Dropout() yerleştirildi. Bu katmanların amacı ise modelin eğitim sırasında hep aynı ağırlıklara yakınsamasını, rastgele seçilen ağırllıkların unutulmasını sağlayarak, engellemektir. Bu sayede model, daha önce hiç görmediği figürlerin tahmin edilmesinde daha başarılı olacaktır.

##### Image Augmentation:

Eğitim setindeki görsellere kaydırma, yakınlaştırma, uzaklaştırma gibi rastgele efektler uygulanarak modelin eğitimi sırasında, aynı görseli ikinci kez görmemesi sağlanabilmektedir. Bunun için Keras’ın ImageDataGenerator fonksiyonu kullanılacaktır.

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
--	------	-------------------------

Yapılan Uygulama :	<b>DERİN ÖĞRENME: KONVOLÜSYONEL SİNİR AĞLARI</b>	Tarih: 24/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

## h. Modelin Derlenmesi ve Eğitimi

```
# Compile & Train the Model
#=====
epochs = 10
batch_size = 64

model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])

# Fitting with Image Augmentation
#-----
from tensorflow.keras.preprocessing.image import ImageDataGenerator

imgAug = ImageDataGenerator(width_shift_range=0.2,
                            height_shift_range=0.2,
                            zoom_range=0.2,
                            shear_range=0.1,
                            rotation_range=10)

history = model.fit(imgAug.flow(ds.x_train, ds.y_train, batch_size=batch_size),
                     steps_per_epoch=len(ds.x_train)//batch_size,
                     epochs=epochs,
                     validation_data=(ds.x_valid, ds.y_valid),
                     shuffle=1)
```

```
Epoch 1/10
25/25 [=====] - 705 3s/step - loss: 1.9847 - accuracy: 0.2887 - val_loss: 1.7926 - val_accuracy: 0.445
0
Epoch 2/10
25/25 [=====] - 715 3s/step - loss: 1.3560 - accuracy: 0.5244 - val_loss: 1.1582 - val_accuracy: 0.582
0
Epoch 3/10
25/25 [=====] - 745 3s/step - loss: 0.9136 - accuracy: 0.6700 - val_loss: 0.7512 - val_accuracy: 0.712
0
Epoch 4/10
25/25 [=====] - 785 3s/step - loss: 0.6644 - accuracy: 0.7600 - val_loss: 0.6600 - val_accuracy: 0.795
0
Epoch 5/10
25/25 [=====] - 855 3s/step - loss: 0.5724 - accuracy: 0.7844 - val_loss: 0.4454 - val_accuracy: 0.847
0
Epoch 6/10
25/25 [=====] - 935 4s/step - loss: 0.4337 - accuracy: 0.8350 - val_loss: 0.3357 - val_accuracy: 0.912
0
Epoch 7/10
25/25 [=====] - 1125 4s/step - loss: 0.3772 - accuracy: 0.8575 - val_loss: 0.3476 - val_accuracy: 0.875
0
Epoch 8/10
25/25 [=====] - 1215 5s/step - loss: 0.3922 - accuracy: 0.8481 - val_loss: 0.3062 - val_accuracy: 0.91
0
Epoch 9/10
25/25 [=====] - 1245 5s/step - loss: 0.3380 - accuracy: 0.8681 - val_loss: 0.2400 - val_accuracy: 0.94
0
Epoch 10/10
25/25 [=====] - 1215 5s/step - loss: 0.2879 - accuracy: 0.8944 - val_loss: 0.1903 - val_accuracy: 0.95
```

Optimizasyon algoritmaları modelin global minimuma en doğru ve en hızlı ulaşması hedefiyle geliştirilmiştir. Derleme sırasında optimizasyon algoritması olarak ‘adam’; Loss (Hata) fonksiyonu olarak *CategoricalCrossEntropy* kullanıldı. (!) Hata fonksiyonunun *sparse* özelliği ise etiketlerin *one-hot* kodlanması gerekliliğini ortadan kaldırmaktadır.

*Overfitting* engellenmesi amacıyla *ImageDataGenerator* yardımıyla eğitime başlandı.

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
--	------	-------------------------

Yapılan Uygulama :	<b>DERİN ÖĞRENME: KONVOLÜSYONEL SİNİR AĞLARI</b>	Tarih: 27/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

### i. Sonuçların İncelenmesi

Sonuçları görselleştirmek incelediğimizde *validation* başarısının *training* başarısının sürekli üzerinde kalmayı başardığını görebiliyoruz. Öyleyse, modelimiz de başarılıdır diyebiliriz. Peki daha başarılı olabilir mi?

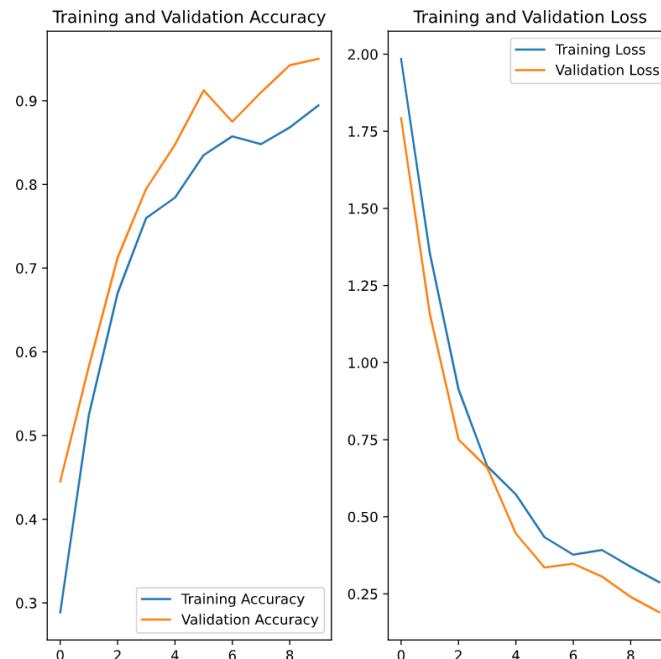
```
# Visualize training results
=====
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

loss=history.history['loss']
val_loss=history.history['val_loss']

epochs_range = range(epochs)

plt.figure(figsize=(8, 8))
plt.subplot(1, 2, 1)
plt.plot(epochs_range, acc, label='Training Accuracy')
plt.plot(epochs_range, val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')

plt.subplot(1, 2, 2)
plt.plot(epochs_range, loss, label='Training Loss')
plt.plot(epochs_range, val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.show()
```



ÖĞRENCİ İmza	ONAY	FİRMA YETKİLİSİ İmza
-----------------	------	-------------------------

Yapılan Uygulama :	<b>DERİN ÖĞRENME: KONVOLÜSYONEL SİNİR AĞLARI</b>	Tarih: 28/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

### j. Daha Gelişmiş bir Model: *Transfer Learning*

Mühendisler olarak tekerlegi yeniden icat etmekten kaçınmalıyız. İlerlemek için var olan teknolojinin üzerine koyabilmeliyiz.

Konvolüsyonel sinir ağları için de benzer bir durum söz konusudur. Mobile-Net-v2 otoritelerce kabul görmüş 1000 farklı nesneyi sınıflayabilen başarılı bir CNN modelidir. Bu modeli kendi proje isterlerimiz doğrultusunda yeniden inşa edebiliriz.

```
# Creating a Sequential CNN Model
#=====
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Conv2D, Flatten, Dropout, MaxPooling2D

# Transfer Learning via Mobile Net v2 Model
#=====
import tensorflow_hub as hub

feature_extractor_url = "https://tfhub.dev/google/tf2-preview/mobilenet_v2/feature_vector/2"

feature_extractor_layer = hub.KerasLayer(feature_extractor_url,
                                         input_shape=(224,224,3))

feature_extractor_layer.trainable = False

model = Sequential([
    feature_extractor_layer,
    Dropout(0.3),
    Dense(10)
])

model.summary()

Model: "sequential"
-----[REDACTED]-----
Layer (type)          Output Shape         Param #
keras_layer (KerasLayer)     (None, 1280)           2257984
dropout (Dropout)        (None, 1280)            0
dense (Dense)           (None, 10)             12810
-----[REDACTED]-----
Total params: 2,270,794
Trainable params: 12,810
Non-trainable params: 2,257,984
```

*Tensorflow Hub* üzerinden Mobile-Net-v2'nin son katmanı çıkarılmış halini indirip yerine kendi projemiz doğrultusunda 10 nöron yani sınıf içeren bir katman ekliyoruz.

Modelin son katman ağırlıkları dışındaki parametreleri yeniden eğitilmez bir başka deyişle dondurulur. 2,270,794 parametrenin yalnızca 12,810 tanesi el hareketlerinin sınıflandırılması amacıyla eğitilecektir.

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
--	------	-------------------------

Yapılan Uygulama :	<b>DERİN ÖĞRENME: KONVOLÜSYONEL SİNİR AĞLARI</b>	Tarih: 28/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

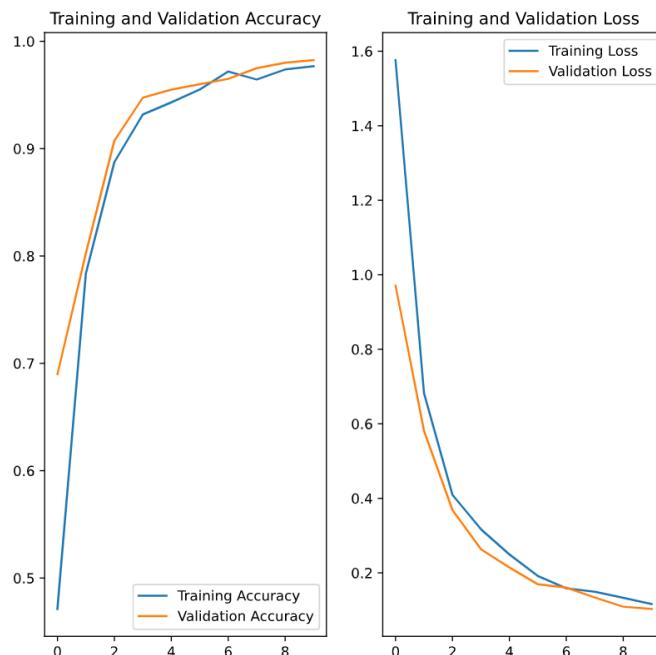
Mobile-Net-v2 modeli giriş şekli(*input shape*) olarak 3 kanallı renkli görselleri kabul etmektedir. Veri setimizi renkli bir şekilde içeri alıp ön işleme sürecinden geçirelim.

First 8 labels of dataset: [7 7 3 5 1 0 5 4]

```
(!) DATA Preprocessing
Dataset is loading..
Rescaling images to 1..
Shuffle process..
Shape of image-set:
(2000, 224, 224, 3)
Shape of one-input-img:
(224, 224, 3)
Shape of label-set:
(2000,)
```



Aynı eğitim koşullarını uyguladığımızda bu modelin daha yüksek başarımı daha hızlı bir şekilde ulaştığını görebiliyoruz.

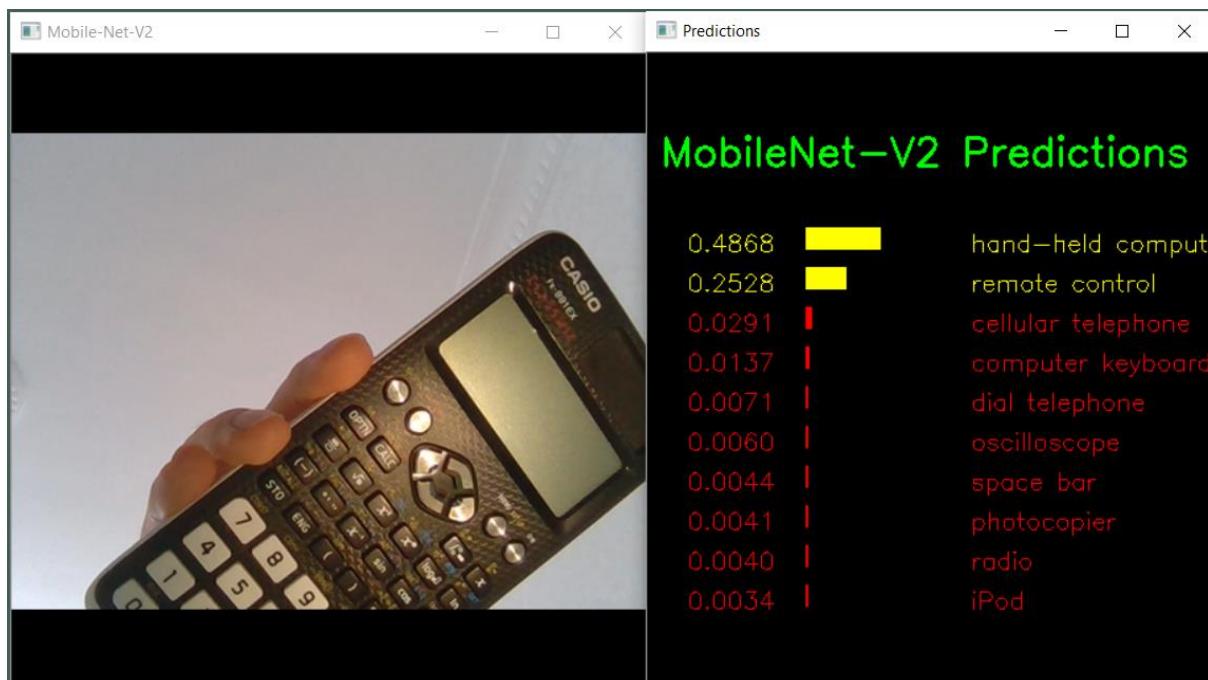


ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
---------------------	------	-------------------------

Yapılan Uygulama :	<b>DERİN ÖĞRENME: KONVOLÜSYONEL SİNİR AĞLARI</b>	Tarih: 29/07/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

#### k. EKSTRA: *Mobile-Net-v2* Nesne Sınıflandırıcı

Transfer Learning sırasında Mobile-Net-v2 modelinin tüm ağırlıklarına erişim sağlayabildiğimiz keşfedilince, elimizdeki gerçek zamanlı verilen girişlere karşın tahmin üreten ve tahmin olasılıklarını arayüze taşıyabilen program, 1000 farklı sınıflandırma görevini yerine getirmesi üzerine Mobile-Net-v2 için özelleştirildi.



Kameraya gösterilen hesap makinesi için Mobile-Net-v2'nin yaptığı tahminler sağ tarafta görülmektedir.

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
--	------	-------------------------

Yapılan Uygulama :	<b>GERÇEK ZAMANLI UYGULAMANIN GELİŞTİRİLMESİ ve CANLI ÖĞRENME</b>	Tarih: 04/08/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

### a. Program Ana Döngüsü

Program akışı soft-real-time gerçekleşecektir. Kameradan okunan her çerçeve(frame) içindeki el algılanmalı, arka planı kaldırılarak ön işlemeye gönderilmeli; ardından tahmin ve tahmin olasılıklarının üretilmesi için önceden eğitilmiş modele giriş olarak verilmelidir. Çıkan sonuçlar değerlendirilerek kullanıcıya sunulmalı yani Graphical User Interface oluşturulmalıdır. Aşağıda bunun için oluşturulmuş oldukça sadeleştirilmiş bir psödo-kod bulunmaktadır.

```
while not ESCAPE:
    PARAMS = get_parameters_from(TRACKBAR)
    FRAME = get_frame_via(WEBCAM)
    INPUT_0 = detect_skin_in(FRAME)
    INPUT_1 = preprocessing_the(INPUT_0)
    OUTPUT = model.predict(INPUT_1)
    show_GUI()
```

Tüm bunlar gerçekleşirken program kullanıcının yönetimindeki Kontrol Panelindeki (Trackbar) parametre değişikliklerine karşı duyarlı kalmalıdır.



### b. Kullanıcı Arayüzü

Arayüz tasarımda numpy yardımıyla sıfırlardan oluşan bir matris ile siyah bir arka plan oluşturuldu. Eğitilmiş konvolüsyonel sinir ağı modelinden gelen tahmin olasılıkları OpenCV kütüphanesinin sağladığı imkanlar doğrultusunda görselleştirildi.

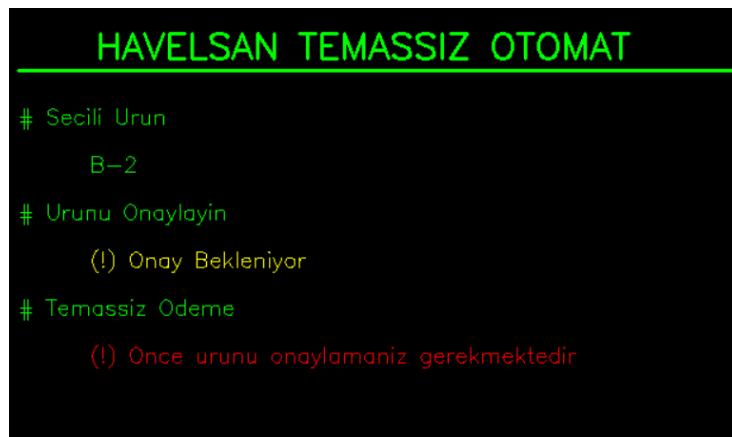
Her bir figür için ölçülen olasılık OpenCV'nin rectangle() fonksiyonunda dikdörtgenin genişliğiyle ilişkilendirildi.

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
---------------------	------	-------------------------

Yapılan Uygulama :	<b>GERÇEK ZAMANLI UYGULAMANIN GELİŞTİRİLMESİ ve CANLI ÖĞRENME</b>	Tarih: 05/08/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

Otomat arayüzü içinse, modelden tahminler ile seçili ürün imlecinin konumu ve ürün onay sekmesi bağıdaştırıldı.

EHB öğrencisi olarak arayüz kodlarının ayrıntısını paylaşma gereği görememekteyim. Giriş sayfasında belirtilen link üzerinden tüm kodlara ulaşabilirsiniz.



### c. Canlı Öğrenme

Elimizde uygun bir veri seti var olduğu sürece Tensorflow ile yapay sinir ağı inşa etmek ve eğitmek oldukça kolaylaştı. Öyleyse en zayıf halka veri seti gibi gözükmektedir. Peki, veri setini ve öğrenmeyi canlı hale getirebilir miyiz?

Gerçek zamanlı öğrenmeyi mümkün kılabilmek için yine Tensorflow'un metotları kullanıldı.

```
# Load Base Model & Freeze Weights
#=====
model_ID = 'hand-recog-mobile-net-v2-tf-1'
model = tf.keras.models.load_model(f'./models/{model_ID}')
for layer in model.layers[:-1]:
    layer.trainable = False
```

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
--	------	-------------------------

Yapılan Uygulama :	<b>GERÇEK ZAMANLI UYGULAMANIN GELİŞTİRİLMESİ ve CANLI ÖĞRENME</b>	Tarih: 06/08/2020
Uygulama Birimi:	EĞİTİM ve SİMÜLASYON TEKNOLOJİLERİ	

Modelin son katmanı hariç diğer parametreleri dondurulur. Ana döngü esnasında Canlı Öğrenme aktifleştirildiğinde sonraki 16 çerçeve ön belleğe alınır. Ardından modelin train\_on\_batch() metoduyla hafızaya alınan 16 çerçevelik batch eğitimi, kontrol panelinden belirlenen etiket doğrultusunda yapılır.

Aşağıda ilk durumda elin algılanmadığı durum için modelin %46.39 olasılık ile 9 etiketini tahmin ettiğini görüyoruz.



İkinci durumda, elin algılanmadığı siyah ekran, 3 etiketyle modele canlı olarak yeniden öğretilmektedir.



Modelin tahmin olasılıklarının canlı öğrenme sonrasında değiştiğini gözlemlayabiliriz.

ÖĞRENCİ İmza 	ONAY	FİRMA YETKİLİSİ İmza
---------------------	------	-------------------------