Python ile Görüntü İşleme
(OpenCV)

OpenCV Nedir?

OpenCV, gerçek zamanlı bilgisayar vizyonunu amaçlayan bir programlama fonksiyon kütüphanesidir. Kütüphane çapraz platformdur ve açık kaynaklı BSD lisansı altında kullanım için ücretsizdir.Kütüphane aynı zamanda makine öğrenmesinide desteklemektedir. Cplusplus diliyle geliştirilmektedir.Kütüphane içerisinde 500'den fazla algoritma ve bu sayının yaklaşık 10 katı kadar da fonksiyon yer almaktadır.

TensorFlow, Torch/PyTorch , Caffe gibi derin öğrenme frameworklerini destekler.Linux,Windows,macOs,FreeBSD gibi masaüstü işletim sistemlerinde ve İOS , Android gibi mobil işletim sistemlerinde çalışabilmektedir.

OpenCV ile Neler Yapılabilmektedir?

- Sabit görselleri , videoları veya webcam görüntülerini okuyup kaydetme
- Yüzleri ve yüz özelliklerini saptama
- Görseller üzerindeki yazıları saptama (plaka,tabela vb.)
- Nesne saptama , sayma ve takibi
- Yüzlerdeki duyguları anlama
- Hareket takibi gibi işlemler yapılabilmektedir.

OpenCV Hangi Yazılım Dillerini Destekler?

- C++
- Java
- Python
- Android SDK

Resim Açma , Resim Okuma , Resim Yazma

Python dosyası oluşturduktan sonra içerisine openCV kütüphanesini dahil ediyoruz.

```
import cv2
```

Resmimizi okutmak için bir tane resim değişkeni oluşturuyoruz.

```
resim = cv2.imread("Resimler/seinfield.jpg")
```

Resmimizi bu şekilde okutursak resmimizi olduğu gibi yani renkli olarak açacaktır.Ancak ikinci parametre olarak “0” (sıfır) değerini verirsek resmimiz gri olacaktır.Gri olmasının nedeni tüm pikseller 0-255 arasında bir değer alacak ve tek bir renk kanalından oluşacaktır.Bu konuya ileride daha detaylı değinilecektir.

Resmimizi ekranda göstermek için :

```
cv2.imshow("Seinfield Resmi",resim)
```

Burada ilk parametre açılacak pencerenin başlığını ikinci parametre ise resmimizi okuttuğumuz resim değişkenimizi belirtir.

Resim açıldığında hemen kapanmaması için :

```
cv2.waitKey(0)
```

waitKey() fonksiyonu resim açıldığında ekranda ne kadar süre kalacağını belirten bir fonksiyondur.Parametreside delay yani bekleme süresidir.Ancak biz parametre olarak “0”(sıfır) değerini verirsek, biz herhangi bir tuşa basmadan resim penceremiz açık kalacaktır.

Açılan tüm pencerelerin kapatılması için:

```
cv2.destroyAllWindows()
```

komutu kullanılır.

Program Çıktısı :

OpenCV Resmimizi Nasıl Yorumlar?

OpenCV ‘de resimlerimizi bir numpy dizisi olarak tutar bunu görmek için:

```
print(type(resim))
```

kodunu çalıştırdığımızda bize

<class 'numpy.ndarray'>

değerini dönmektedir.Buradan resimlerimizin bir numpy array’i olarak saklandığını görebiliriz

Renk Sistemleri

Bilgisayarların anlamlandırdığı belirli renk sistemleri ve renk uzayları (color space)bulunmaktadır.Bunlardan bazıları RGB,BGR,HSV ve YCbCr’dır.

RGB(Red,Green,Blue)

Çok yaygın kullanılan bir renk sistemidir.İngilizce anlamı olarak red(kırmızı),gree(yeşil),blue(mavi) renklerinin baş harflerinden oluşturulmuştur.

Bu sistem, bilgisayar ve televizyon gibi elektronik cihazların algılama ve sunum mekanizmalarında;ayrıca fotoğrafçılık alanında yaygın şekilde kullanılmaktadır.Birbirine çok yakın biçimde konumlandırılan temel renk kaynakları birbirini etkileyerek insan gözünün algılayabildiği renk dizilimindeki bileşimleri oluşturur.

Renkleri belirtmek için tamsayılar kullanılır.8bitlik tam sayılar kullanıldığında aynı rengin 256 ,16bitlik sayılar kullanıldığında 65536 farklı tonunu elde etmek mümkündür.

OpenCV’de genellikle 8bitlik renkler kullanılır.

OpenCV ve BGR Mantığı

RGB renk sistemiyle temelde aynıdır.Sadece Kırmızı ve mavi renkler sıralamada birbirlerinin yerine geçmişlerdir.

OpenCV’nin ana renk sistemi BGR(blue,green,red)’dır.

OpenCV ‘de resmimizi print fonksiyonu ile yazdırdığımızda bize bir matris döner.Bu matris piksellerin BGR renk modelindeki renk skalasını döner.

```
print(resim)
```

```
Blue    Green      Red
[ 12        15         20]
[ 12        15         20]
[11         14         19]
 ...
```

Bize bu şekilde çok uzun bir matris değerleri dönmektedir.Her satır bir pikseli ifade etmektedir ve değerler 3 farklı kanalda gösterilir(R G B).Bu şekilde mavi , yeşil ve kırmızı kullanarak tüm renkler elde edilebilir.Her bir renk skalası için değerler 0 – 255 arasındadır.Yani her bir kanal için 256 farklı ton vardır.

Peki bir piksel için kaç farklı renk kombinasyonu vardır?

256 * 256 * 256 = 16.777.216 farklı renk kombinasyonu vardır.

Resimler Neden Gri Tona Çevirilir?

OpenCV'de görüntü işleme çalışılırken resimler üzerinde gri ton kullanılır.Bunun nedeni resimlerimizin 3 farklı kanalda değil de tek bir kanaldan üretilmesidir.Bu hem resmimizin boyutunu küçültür.Hem de üzerinde çalışmamız gereken pikselleri azaltır.Resim renkli iken üzerinde işlem yapacağımız 3 farklı kanal olacağından iş yükümüz artmaktadır.Bunu önlemek için tek bir kanal üzerinden çalışılır ve 0 – 255 değerleri arasında çalışma yapılır.Burada 0 -> siyah , 255 -> beyazdır.Resim pikselleri sadece bu iki siyah ve beyaz skalası arasında değer alacağı için gri olur.

Resim Verileri Hangi Türde Saklanıyor?

Bunu öğrenmek için :

```
print(resim.dtype)
```

komutunu çalıştırırız ve bize;

uint8(unsigned integer 8 bit) değerini döner bu ne demektir:

0 0 0 0 0 0 0 0 = 0(min)

1 1 1 1 1 1 1 1 = 255(max)

Resmimizin Boyutlarını Öğrenmek ?

Resmimizin yükseklik , genişlik ve renk kanalı değerlerini öğrenmek için :

```
print(resim.shape)
```

komutu çalıştırılır ve bize

(500, 390, 3) şeklinde değer döner.Burada birinci değer yükseklik ikinci değer genişlik üçüncü değer ise kaç tane renk kanalı kullanıldığıdır.Eğer resmimizi gri tonda açsaydık bize (500, 390) değerini dönecekti.

OpenCv'de Yazı Yazmak

OpenCV kütüphanesinin doğal olarak desteklediği font sayısı kısıtlıdır.Üstelik bu fontlar sadece ASCII karakterleri desteklemektedir.Yani Türkçe karakter kullanarak yazı yazmamız durumunda desteklenmeyen , Türkçe'ye özgü karakterlerin yerinde soru işareti bulunacaktır.

OpenCV de yazılar putText() fonksiyonu ile yazılır.putText () fonksiyonun özelliklerini inceleyelim:

```
img=cv.putText(img, text, org, fontFace, fontScale, color[, thickness[, lineType[, bottomLeftOrigin]]])
```

parametre tanımları bu şekildedir.Şimdi de bunların anlamlarına bakalım

Parametreler

img	Arkaplan görüntüsüdür.Hangi resmin üzerine yazı yazılacaksa o resimi ifade eder.
text	String türünde yazıyı ifade eder Türkçe karakter kullanılamaz.Kullanılırsa Türkçe karakterin yerine soru işareti gelecektir.
org	Resmin sol alt köşesini piksek cinsinden ifade eder.
fontFace	Sadece HERSHEYFONT ismindeki fontları tanır.Bu fontlar 8 adettir.
fontScale	Yazının büyüklüğünü ifade eder.Çarpan olarak ifade edilebilir.Normal ve varsayılan değeri 1'dir.
color	Yazının rengini belirtir.BGR renk uzayına göre olan renkleri içerir.
thickness	Harflerin çizgi kalınlığını ifade eder.
lineType	Yazı tipini ifade eder.
bottomLeftOrigin	Değer eğer False ise solt alt köşeye göre yazıyı düzgün bir biçimde yazar.True değerini verirsek yazıyı ters(tepetaklak) yazar.

HERSHEY Fontları Nelerdir ?	
FONT_HERSHEY_SIMPLEX Python: cv.FONT_HERSHEY_SIMPLEX	normal boyutta sans-serif yazı tipi
FONT_HERSHEY_PLAIN Python: cv.FONT_HERSHEY_PLAIN	küçük boyutta sans-serif yazı tipi
FONT_HERSHEY_DUPLEX Python: cv.FONT_HERSHEY_DUPLEX	normal boyutta sans-serif yazı tipi (FONT_HERSHEY_SIMPLEX'ten daha karmaşıktır.)
FONT_HERSHEY_COMPLEX Python: cv.FONT_HERSHEY_COMPLEX	normal boyutta sans-serif yazı tipi
FONT_HERSHEY_TRIPLEX Python: cv.FONT_HERSHEY_TRIPLEX	normal boyutta sans-serif yazı tipi (FONT_HERSHEY_COMPLEX'ten daha karmaşıktır.)
FONT_HERSHEY_COMPLEX_SMALL Python: cv.FONT_HERSHEY_COMPLEX_SMALL	FONT_HERSHEY_COMPLEX'in küçük versiyonudur.
FONT_HERSHEY_SCRIPT_SIMPLEX Python: cv.FONT_HERSHEY_SCRIPT_SIMPLEX	El-yazı stili yazı tipi
FONT_HERSHEY_SCRIPT_COMPLEX Python: cv.FONT_HERSHEY_SCRIPT_COMPLEX	FONT_HERSHEY_SCRIPT_SIMPLEX'in daha karmaşık bir şekli
FONT_ITALIC Python: cv.FONT_ITALIC	Fontların italik yazılıp yazılmayacağını ifade eder sayısal değeri 16'dır.

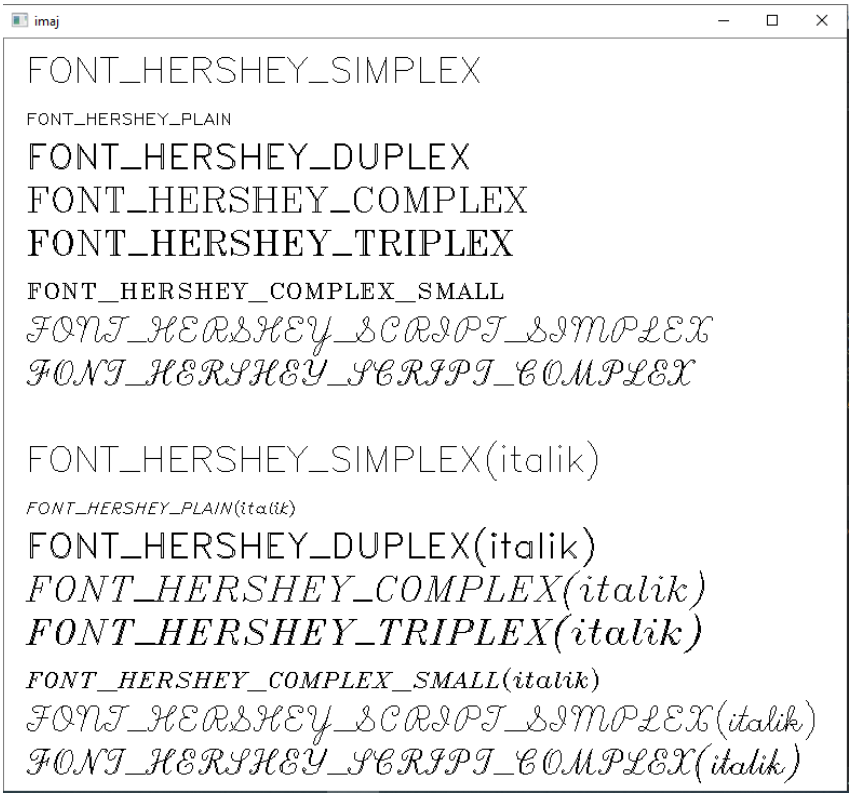
Yazı Tipleri Nelerdir ?(lineType)

Enumerator	
FILLED Python: cv.FILLED	
LINE_4 Python: cv.LINE_4	4-bağlı hat
LINE_8 Python: cv.LINE_8	8-bağlı hat
LINE_AA Python: cv.LINE_AA	Kenar yumuşatıcı hat

3-YazıYazmak.py

```
import cv2
import numpy as np

fontlar = [
   'FONT_HERSHEY_SIMPLEX', 'FONT_HERSHEY_PLAIN',
   'FONT_HERSHEY_DUPLEX','FONT_HERSHEY_COMPLEX',
   'FONT_HERSHEY_TRIPLEX','FONT_HERSHEY_COMPLEX_SMALL',
   'FONT_HERSHEY_SCRIPT_SIMPLEX',
   'FONT_HERSHEY_SCRIPT_COMPLEX'] #8 tane fontumuzla yazı yazmak için 8 tane yazı oluşturduk
imaj = np.ones((720,780,3),np.uint8)*255#beyaz imaj (resim)
for j in range(8):#font numaraları 8 tane olduğu için sırasıyla 0'dan başlayarak farklı fontlar oluşturacak.Bu değerler 0-7 arasında toplam 8 tanedir.
   cv2.putText(imaj,fontlar[j],(20,40+j*40),j,1.1,(0,0,0),1)
   '''
      putText fonksiyonun parametreleri
      birinicisi Hangi görsele ekleyeceğimizi
      ikincisi Ne yazacağımızı
      üçüncüsü Görselin hangi kısmına ekleyeceğimizi
      dördüncüsü Hangi formatta yazacağımızı
      beşincisi Skalamızı ekliyoruz (boyut)
      altıncısı Hangi renk ile yazacağımızı
      yedincisi de kalınlık belirtir
      biz burada opencv'nin bize sunduğu tüm fontlar ile yazı yazdık bunu for döngüsü içerisinde yaptık
   '''

italik = 16 #yazıyı italik yapmak için fontun değerine 16 ekliyoruz.Bazı font stilleri italik
            # yazıyı desteklememektedir.
for j in range(8):
   cv2.putText(imaj,fontlar[j]+'(italik)',(20,400+j*40),j+italik,1.1,(0,0,0),1)

cv2.imshow('imaj',imaj)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Program Çıktısı :



Türkçe Karakter Destekli Yazı Yazmak

OpenCV'nin desteklediği fontlar sadece ASCII karakterlerine izin vermekte ama Türkçe karakter yazdırabilmek için pillow kütüphanesi kullanarak bunu çözebiliyoruz.Pillow kütüphanesi açık kaynak kodlu bir grafik işleme kütüphanesidir.Aslında burada yaptığımız pillow kütüphanesiyle yazı şeklinde grafik çizmek bu yüzden Türkçe karakterleri oluşturabiliyoruz.

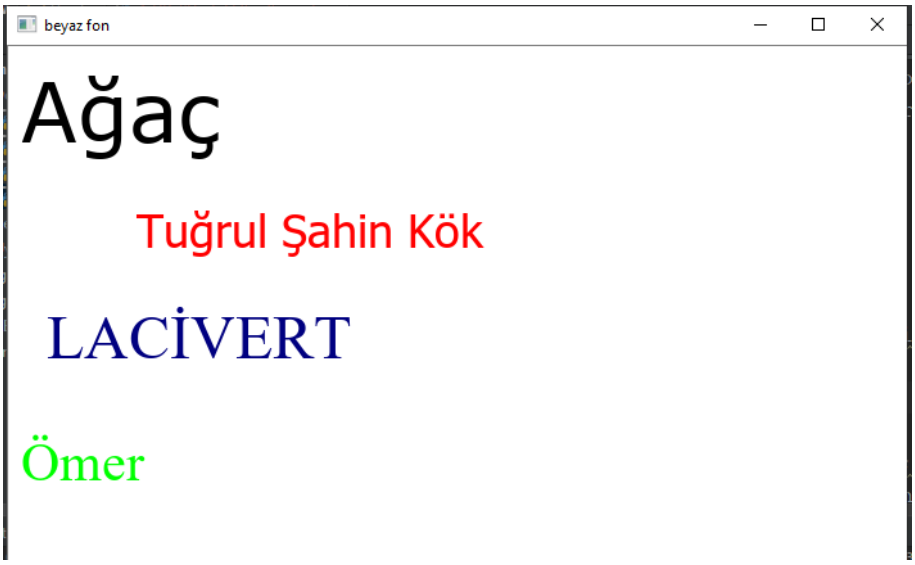
4-Utf8-DestekliYaziYazmak.py

```
import cv2
import numpy as np
from PIL import Image, ImageDraw, ImageFont


# Türkçe karakterleri de yazabilmek icin pillow kutuphanesini kullaniyoruz
# pillow kutuphanesi bir goruntu isleme kutuphanesidir.Python da goruntu islemleri
yapilabilmesi icin olusturulmustur.
def print_utf8_text(image, text, fontName='DejaVuSerif',
        color=(0,0,0), yer=(10,10), boy=48):
    font = ImageFont.truetype(fontName, boy)  # font adi ve boyutu
    pilImg = Image.fromarray(image)  # imajı pillow moduna çevir
```

```
    #Image opencv de BGR renk uzayini kullaniyor ancak PIL RGB uzayini kullandigi icin
onun cevirimini yapiyoruz.
    draw = ImageDraw.Draw(pilImg)  # imajı hazırla
    #ImageDraw basit iki boyutlu grafik olusturulmasini saglar
    #burada yapilan pilimg'i cizmek

    draw.text((yer[0], yer[1]), text,
              fill=(color[0], color[1], color[2],0),font = font)


    image = np.array(pilImg)  # resmi tekrar opencv'nin kullanabilecegi moda (BGR)
ceviriyoruz
    return image
'''

    PIL.ImageDraw.Draw.text(xy, text, fill=None, font=None)
        Parameters:
            xy –metnin sol ust kosesi
            text – cizilecek metin
            fill – metin icin kullanilacak renk
            font – ImageFont ornegi
'''

if __name__ == "__main__":


    imaj = np.ones((400, 700, 3), dtype=np.uint8) * 255#beyaz bir zemin olusturak icin



    fontName='verdana.ttf'; renk = (0, 0, 0); yer = (10,10); boy = 64#sirayla deger atama
islemleri.
    #ayni satirda yazdigimiz icin aralara ; koyduk
    imaj = print_utf8_text(imaj, "Ağaç ", fontName,
                           renk, yer, boy)
    #fonksiyonumuzun icerisine parametrelerimizi gonderiyoruz
    # ve ayni islemleri farkli yazilar yazmak icin tekrarliyoruz

    fontName = 'tahoma.ttf'
    renk = (0, 0, 255); yer = (100,120); boy = 36
    imaj = print_utf8_text(imaj, "Tuğrul Şahin Kök ", fontName,
                           renk, yer, boy)

    fontName = 'times.ttf'; renk = (128, 0, 0)
    yer = (30,200); boy = 48
    imaj = print_utf8_text(imaj,
       "LACİVERT ", fontName, renk, yer, boy)

    fontName = 'times.ttf'; renk = (0, 255, 0)
    yer = (10,300); boy = 42
```

```
    imaj = print_utf8_text(imaj, "Ömer ", fontName, renk, yer, boy)

    cv2.imshow('beyaz fon', imaj)
    cv2.moveWindow('beyaz fon', 10, 10)# pencerenin x,y duzleminde nerede acilacagini
belirtir

    cv2.waitKey(0)
    cv2.destroyAllWindows()
```

Program Çıktısı:



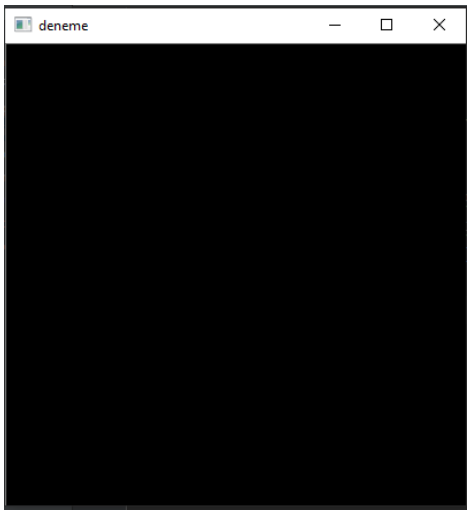
Görseller ve Matris İşlemleri

OpenCV'de resimler birer matris olarak görülür.Matris işlemleri numpy kütüphanesi ile yapılır.Oluşturacağımız görselleri matris ile oluşturur

5-bos_resim.py

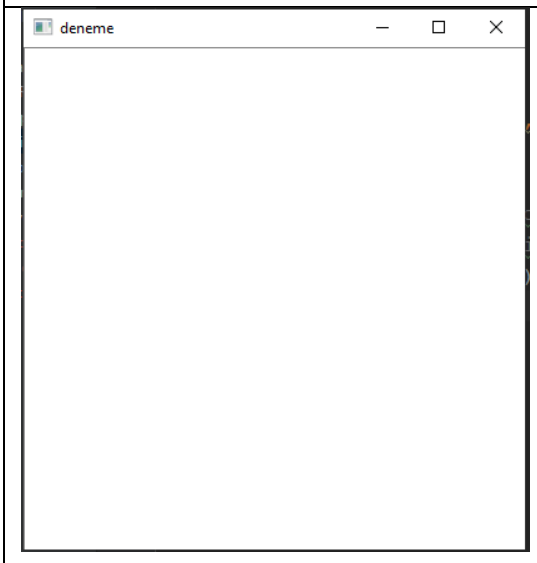
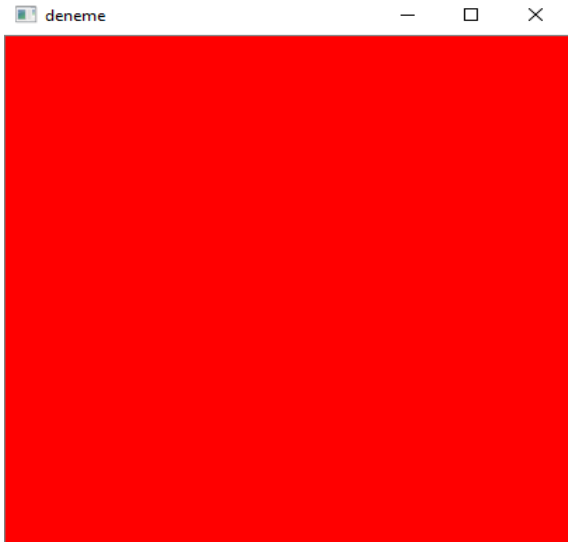
```
import numpy as np
import cv2

deneme = np.zeros((400,400,3),dtype=np.uint8)
#deneme[:] = (255,255,255) beyaz piksel degerleri
#deneme[:] = (0,0,255) kirmizi piksel degerleri
#400x400 piksel boyutlarinda 3 renk kanalina sahip bir siyah pencere olusturur.
#bu resmi bir numpy matrisiyle olusturup icerisini sifir ile doldurduk.(np.zeros)
cv2.imshow('deneme',deneme)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Program Çıktısı:



Bu 400x400 piksellik 3 renk kanalına sahip siyah bir penceredir.Bu görüntüyü bir numpy matrisiyle oluşturup içerisini 0(sıfır)'larla doldurduk.(np.zeros)

Bu siyah görüntüyü eğer beyaza çevirmek istersek o zaman görselimizin her bir pikselindeki renk bilgisini beyaza karşılık gelen (255,255,255) demetiyle doldurmamız gerekir.

Beyaz bir kare:	Kırmızı bir kare:
	

Renk Kanalları

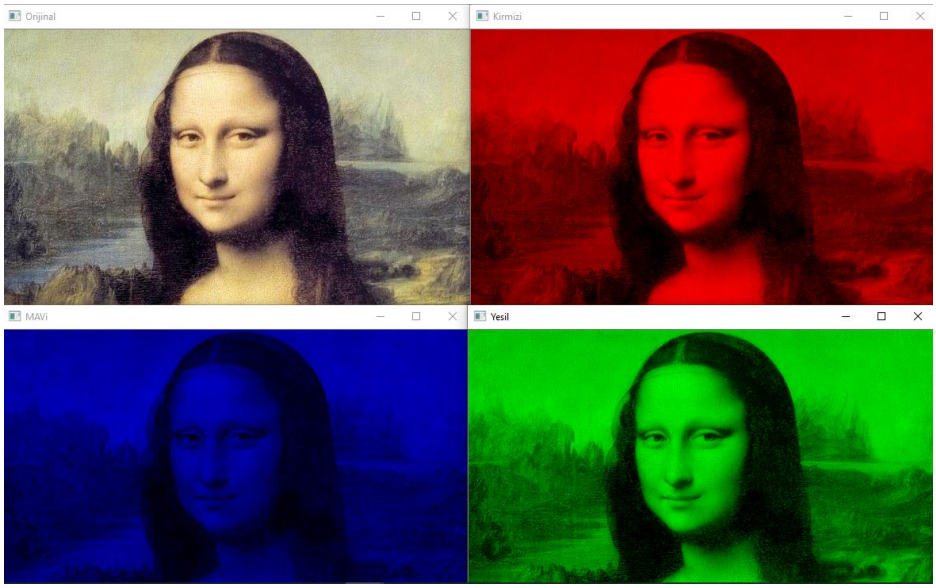
BGR renk uzayına sahip olan OpenCV'de görüntümüzün 3.boyutu renk kanallarını tutar.Herhangi bir resmi oluşturmak için bu renk kanallarının birlikte kullanılması gerekmektedir.Renk kanallarını birer filtre olarak düşünürsek 3 tane renk kanalımız(BGR) üst üste gelince resmimizi renkli biçimde görürüz.Bu renk kanallarını ayrı ayrı görüntülemekte mümkün.Yani resmimizi mavi , yeşil ve kırmızı filtreler ile gösterebilir.Bunu yapmak için bir renk kanalını göstermek için diğer renk kanallarının tüm değerlerini sıfırlamak olacaktır.

6-renk_kanallari.py

```
import cv2

deltax=0
deltay=0

img = cv2.imread('../Resimler/monalisa.jpg')
m = img.copy()
m[:,:,1]=0
m[:,:,2]=0
# mavi renk filtresini elde etmek icin yesil ve kirmizi renk kanallarini sifira esitliyoruz

y = img.copy()
y[:,:,0]=0
y[:,:,2]=0
# Yesil renk filtresini elde etmek icin mavi ve kirmizi renk kanallarini sifira esitliyoruz

k = img.copy()
k[:,:,0]=0
k[:,:,1]=0
# Kirmizi renk filtresini elde etmek icin yesil ve kirmizi renk kanallarini sifira esitliyoruz
print(img.shape)#resmin boyutlarini verir -> yukseklik, genislik, renk kanallari

#renk kanallarinin hepsinin ayni anda olmasi resmimizin orijinal halinin ortaya cikmasini
sagliyor
cv2.imshow("Orijinal",img);cv2.moveWindow('Orijinal',10,10)
'''
    Yükseklik, genişlik ve kanal sayısına img.shape
    ile erişebiliriz: Yükseklik indeks 0'da,
    Genişlik indeks 1'de; ve indeks 2'deki kanal sayısı.
'''

cv2.imshow('MAVi',m)
cv2.moveWindow('Mavi',10,img.shape[0]+deltay)
#moveWindow(x = 10,ya = yukselik + deltay)

cv2.imshow('Kirmizi',k)
cv2.moveWindow('Kirmizi',img.shape[1]+deltax,10)
#moveWindow(x = genislik + deltax , y = 10)
```

```
cv2.imshow('Yesil',y)
cv2.moveWindow('Yesil',img.shape[1]+deltax,img.shape[0]+deltay)
#moveWindow(x = genislik + deltax , yukseklik + deltay)
#moveWindow() ile pencerelerin acilma konumlarini ayarliyoruz
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Program Çıktısı:



Yeniden Boyutlandırmak

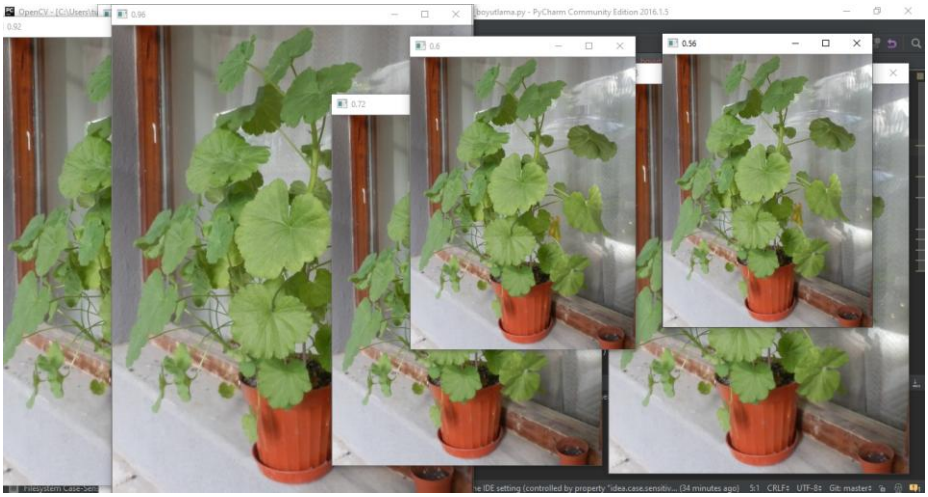
OpenCV kullanırken bazen resimleri ihtiyacımız çerçevesinde yeniden boyutlandırmamız gerekebilir.Bunu en-boy oranını koruyarak yapabileceğimiz gibi bu bir zorunluluk değildir.Resimlerimizi istediğimiz boyutlarda tekrar boyutlandırabiliriz.

7-yeniden_boyutlama.py

```
import cv2
import random

imaj = cv2.imread('../Resimler/sardunya2.jpg')
cv2.imshow('imaj',imaj)

oran = 0.8
imajlar=[]#resimleri bir liste icerisinde tutucaz
for j in range(10):
    oran = random.randint(1,25)/25 # 0 ile 1 arasinda bir sayi olusturur
    rx = int(imaj.shape[1]*oran) #Yeni imajin genisligi
    ry = int(imaj.shape[0]*oran) #yeni imajin yukseligi
    x = random.randint(100,1600)#100-1600 arasinda rastgele x koordinati
```

```
    y = random.randint(100,800)#100-800 arasinda rastgele y koordinati
    imajlar.append((str(oran),cv2.resize(imaj,(rx,ry))))
    #listeye ekleme fonksiyonu, cv2.resize(orjinal_imaj,(yeni_yukseklik,yeni_genislik)
    cv2.imshow(imajlar[j][0],imajlar[j][1])
    #pencere ismi olarak kullanilan orani ve yeni olusturan imajin kendisini kullaniyoruz
    cv2.moveWindow(imajlar[j][0],x,y)
    #bir ust satirda kullanidigimiz isimdeki pencerenin x ve y random komutlarini aliyoruz
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Program Çıktısı:



Maskeleme

Maskeleme resim birleştirme işlemlerinde kullanılan bir araçtır.Maske siyah ve beyazdan oluşan bir görüntüdür.Maske siyah ve beyazdan oluşan bir görüntüdür.Uygulama işlemi gerçekleştirildiğinde beyaz olan pikseller işleme sokulurken siyah olan pikseller işleme alınmazlar.

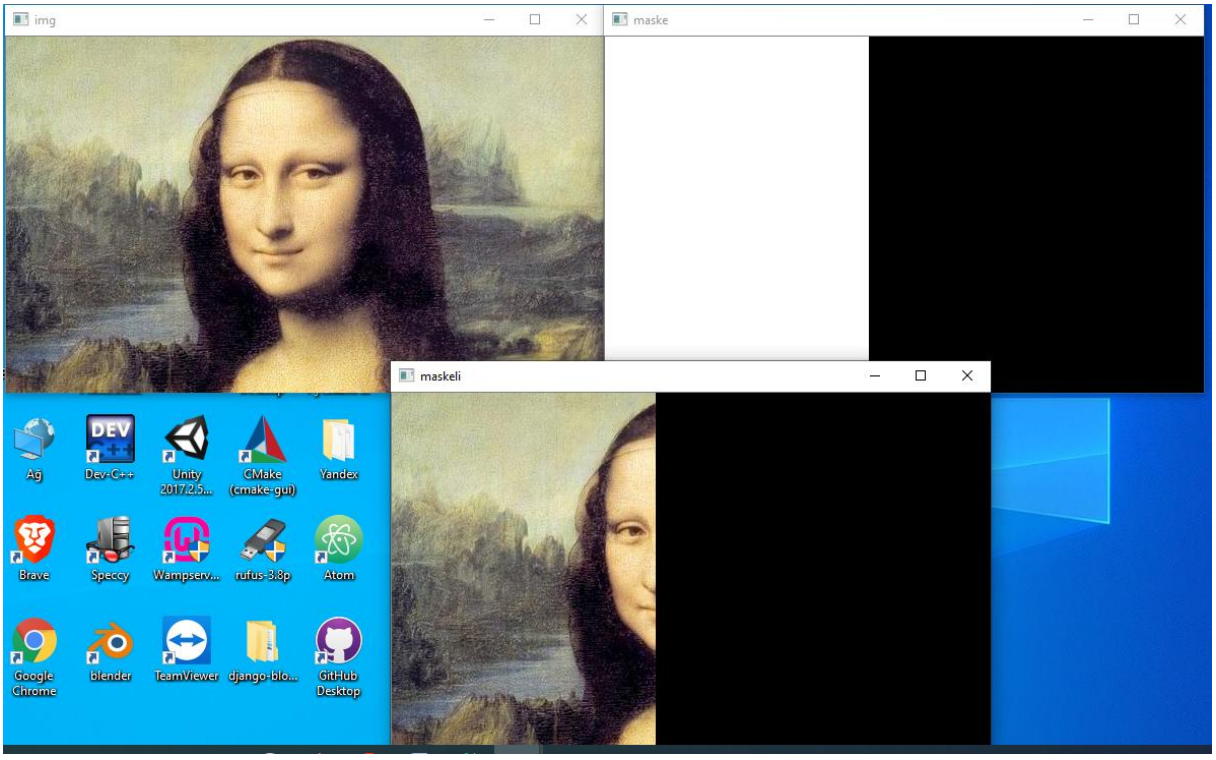
Örneğimiz de monalisa.jpg dosyamızın maskesini yarısı beyaz diğer yarısı siyah olacak şekilde oluşturduk.Resmimizi ve maskemizi bitwise_and() komutu ile işleme soktuğumuzda resmimizin yarısı monalisa diğer yarısı ise siyah piksellerden oluşturduk.

8-maskeleme.py

```
import cv2
import numpy as np


deltax = 0
deltay = 0

img = cv2.imread("../Resimler/monalisa.jpg")
```

```
maske = np.ones(img.shape,dtype="uint8")*255 #beyaz bir resim olusturduk
   #img.shape resmin boyutu kadar bir pencere olusturmamizi sagliyor
maske[:,260:] = [0,0,0]#resmin ortasina kadar siyah alan olusturmak


maskeli = cv2.bitwise_and(img,maske)#maskenin siyah bolgeleri siyah kalir beyaz kisimlari diger resim ile doldurulcak
      #cv2.bitwise_and(ilk girdi dizisi,ikinci girdi dizisi)
      #ilk parametre okuttugumuz resmimiz ikincisi ise olusturdugumuz siyah maske
      #bitwise_and iki resmin bitsel birlesimini saglar
cv2.imshow('img',img)#resim gosterme islemi
cv2.moveWindow('img',10,10)#resmin nerede cikagi
cv2.imshow('maske',maske)#maske gosterimi
cv2.moveWindow('maske',img.shape[1]+deltax,10)#resmin nerede cikacagi
cv2.imshow('maskeli',maskeli)#maske gosterimi
cv2.moveWindow('maskeli',380,img.shape[0]+deltay)#maskeli resmin nerede cikacagi

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Program Çıktısı:



Resim Ekleme

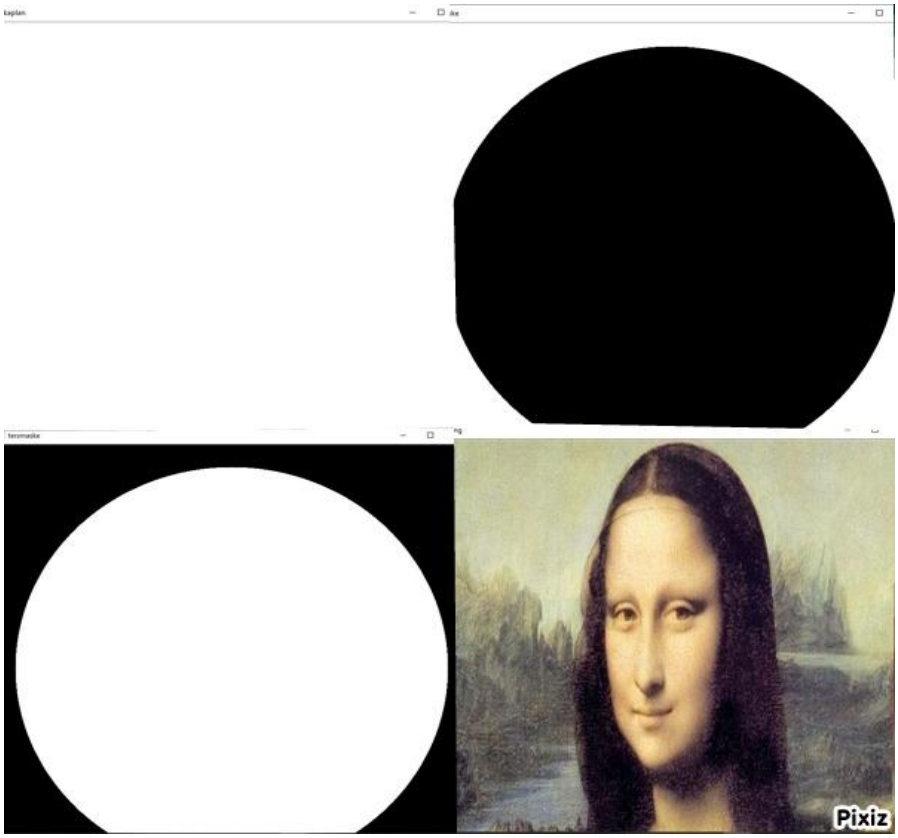
OpenCV'de herhangi bir resim üzerine başka bir resim ekleme işlemi yapabiliyoruz.Uygulama olarak yapmak istediğimiz maske ve ters maske oluşturarak arkplan resmimizin ortasına monalisa.jpg dosyamızı oturtmak.Kodlarımızı inceledikten sonra bunu adım adım gözlemleyelim.

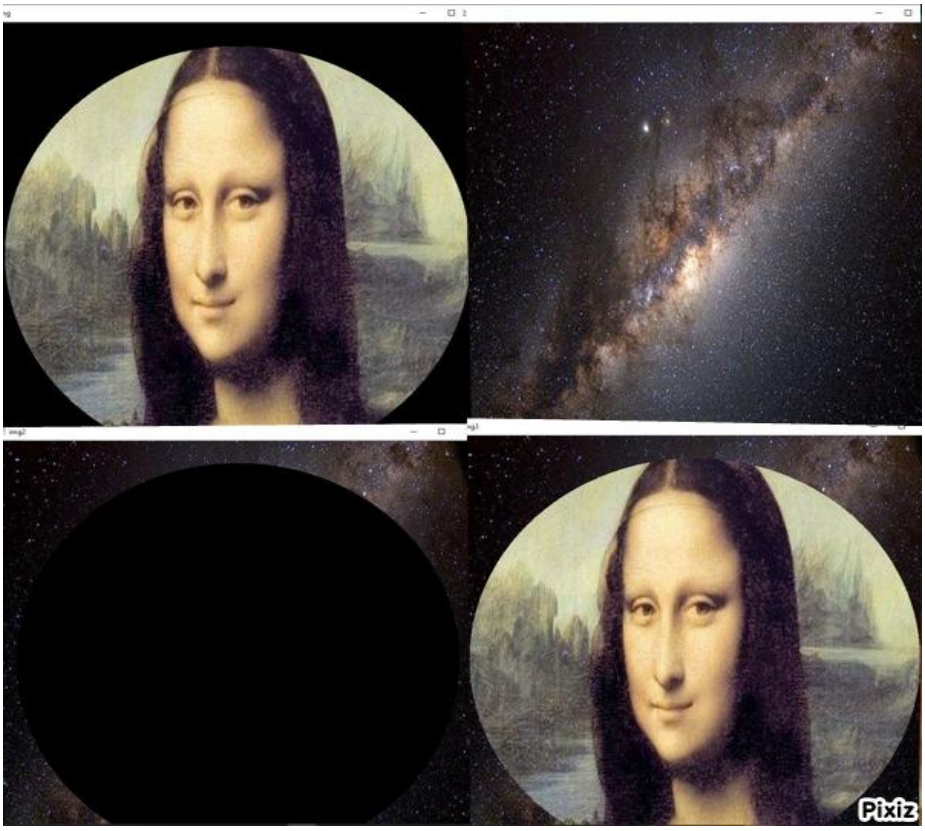
9-resim_ekleme.py

```
import cv2
import numpy as np

bekle = True
#tusa bastigimizda false 'a doner
arkaplan = np.ones((800,800,3),dtype=np.uint8)*255#800x800 boyutlarinda beyaz arkaplan
if bekle:
    cv2.imshow('arkaplan',arkaplan)#arkaplani goster
    cv2.waitKey(0)

#Arkaplanin ortasina 360piksel yaricapli siyah bir daire cizimi
maske = cv2.circle(arkaplan,(400,400),360,(0,0,0),-1)
#beyaz arkaplanin uzerine 400x400 piksel ve capi 360px olan bir siyah cember cizdiriyoruz


if bekle:
    cv2.imshow('maske',maske)
    cv2.waitKey(0)

#ters maske islemi
tersmaske = (255-maske)#bu islem maskedeki 0'lari 255'e , 255'leri 0'a cevirir yani beyazlar
siyah,siyahlar beyaz olur
if bekle:
    cv2.imshow('tersmaske',tersmaske)
    cv2.waitKey(0)
#resim yukleme
img = cv2.imread('../Resimler/monalisa(800x800).jpg')
if bekle:
    cv2.imshow('img',img)
    cv2.waitKey(0)
#ters maskeyi img ye uygula

img = cv2.bitwise_and(img,tersmaske)#resim uzerine tersmaskeyi uyguluyoruz
#resmin orta kisminda monalisa (maskenin beyaz yerleri) dis kisminda siyah alan olusacak
if bekle:
    cv2.imshow('img',img)
    cv2.waitKey(0)

#diger resmi yukle ve maske uygula
```

```
img2 = cv2.imread('../Resimler/uzay(800x800).jpg')
if bekle:
    cv2.imshow('img2',img2)
    cv2.waitKey(0)
img2 = cv2.bitwise_and(img2,maske)#uzay resmimize maskeyi uyguluyoruz
#resmin ortasi siyah kenarlari beyaz oluyor.Beyaz(dis) kisma uzay resmimiz gelmis oluyor.
if bekle:
    cv2.imshow('img2',img2)
    cv2.waitKey(0)
#iki resim birlestirme islemi
img3 = img + img2#maske ve tersmaske uygulanmis resimleri birlestiriyoruz
cv2.imshow('img3',img3);cv2.waitKey(0)
cv2.destroyAllWindows()

'''
İşlem Asamalari:
    1-arkaplan olusacak
    2-maske olusacak
    3-tersmaske olusacak
    4-monalisa resmini yukledik
    5-monalisaya ters maske uyguladik
    6-uzay resmini yukledik
    7-uzay resmine maske uyguladik
    8-son olarak iki maske uygulanmis goruntuyu birlestirme islemini gerceklestirdik
'''
```

Program Çıktısı:




Pixiz