KONYA TEKNİK ÜNİVERSİTESİ

DOĞA BİLİMLERİ VE MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

SAYISAL GÖRÜNTÜ İŞLEME

PROJE RAPORU

191220041

Kerim Kara

İçindekiler Tablosu

1- İşlem

```python
import cv2
import numpy as np
import argparse
def equalHistColor (img):
    img_yuv = cv2.cvtColor(img, cv2.COLOR_BGR2YUV)
    img_yuv[:,:,0] = cv2.equalizeHist(img_yuv[:,:,0])
    img_output = cv2.cvtColor(img_yuv, cv2.COLOR_YUV2BGR)
    return img_output

def click_and_crop(event, x, y, flags, param):
    global x_start, y_start, x_end, y_end, cropping, getROI
    if event == cv2.EVENT_LBUTTONDOWN:
        x_start, y_start, x_end, y_end = x, y, x, y
        cropping = True
    elif event == cv2.EVENT_MOUSEMOVE:
        if cropping == True:
            x_end, y_end = x, y
    elif event == cv2.EVENT_LBUTTONUP:
        x_end, y_end = x, y
        cropping = False
        getROI = True
x_start, y_start, x_end, y_end = 0, 0, 0, 0
cropping = False
getROI = False
refPt = []
image = cv2.imread('hsv9.jpg')
clone = image.copy()
cv2.namedWindow("image")
cv2.setMouseCallback("image", click_and_crop)
while True:
    i = image.copy()
    if not cropping and not getROI:
        cv2.imshow("image", image)
    elif cropping and not getROI:
        cv2.rectangle(i, (x_start, y_start), (x_end, y_end), (0, 255, 0), 2)
        cv2.imshow("image", i)
    elif not cropping and getROI:
        cv2.rectangle(image, (x_start, y_start), (x_end, y_end), (0, 255, 0), 2)
        cv2.imshow("image", image)
    key = cv2.waitKey(1) & 0xFF
    if key == ord("r"):
        image = clone.copy()
        getROI = False
        break
    elif key == ord("c"):
        break
refPt = [(x_start, y_start), (x_end, y_end)]
if len(refPt) == 2:
    roi = clone[refPt[0][1]:refPt[1][1], refPt[0][0]:refPt[1][0]]
    hsvRoi = cv2.cvtColor(roi, cv2.COLOR_BGR2HSV)
    print('min H = {}, min S = {}, min V = {}; max H = {}, max S = {}, max V = {}'.format(hsvRoi[:,:,0].min(), hsvRoi[:,:,1].min(),
                                                                                          hsvRoi[:,:,2].min(),

    hsvRoi[:,:,0].max(), hsvRoi[:,:,1].max(), hsvRoi[:,:,2].max()))
cv2.destroyAllWindows()
```

Python programlama dili ile verilen resimde seçilen alanın mim ve max hsv değerlerini getiren program yaptım.

2- İşlem

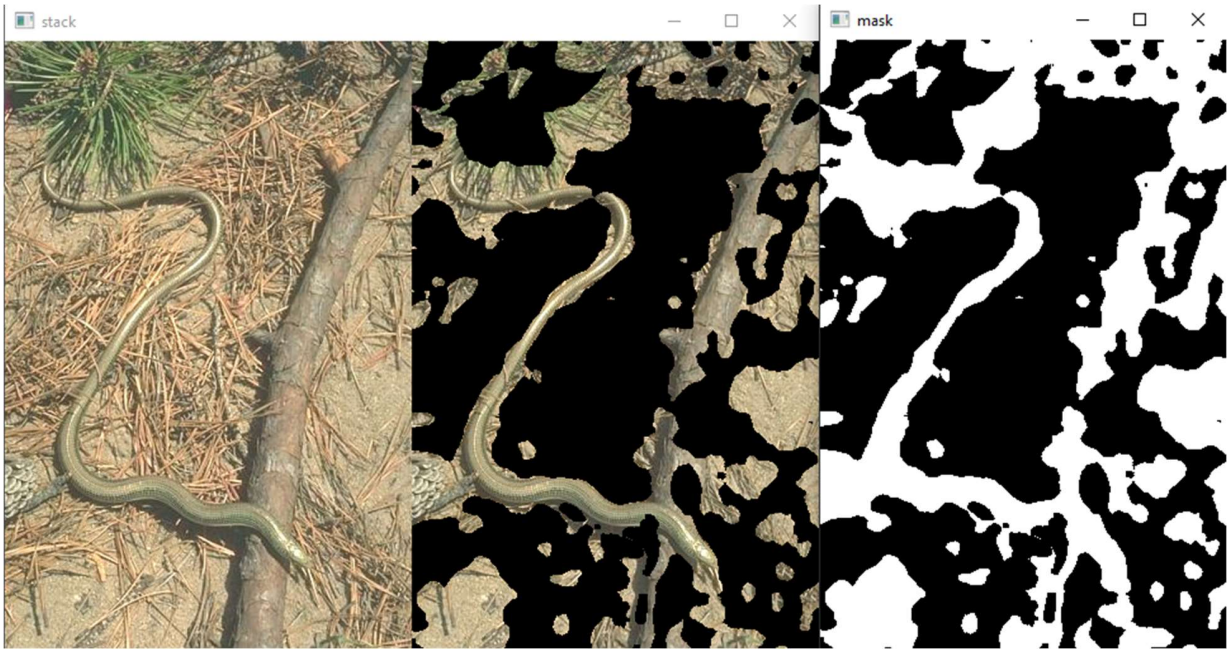
Orijinal resimde kontrast germe ve parlaklık arttırma yaptım.

```
from __future__ import print_function
from builtins import input
import cv2 as cv
import numpy as np
import argparse
image = cv.imread('sayisal.jpg')
new_image = np.zeros(image.shape, image.dtype)
alpha = 1.0 # Simple contrast control
beta = 50    # Simple brightness control
for y in range(image.shape[0]):
    for x in range(image.shape[1]):
        for c in range(image.shape[2]):
            new_image[y,x,c] = np.clip(alpha*image[y,x,c] + beta, 0, 255)
cv.imshow('Original Image', image)
cv.imwrite('kontrast.jpg', new_image)
cv.waitKey()
```

3- İşlem

2. İşlemin çıktısını 1. İşlemde bulunan program ile yılanın bir bölümünde alınan hsv değerleri ile maske oluşturdum. Orijinal resme 15x15 median filtresi uyguladım. Ardından resme, oluşturduğum maskeyi uyguladım.

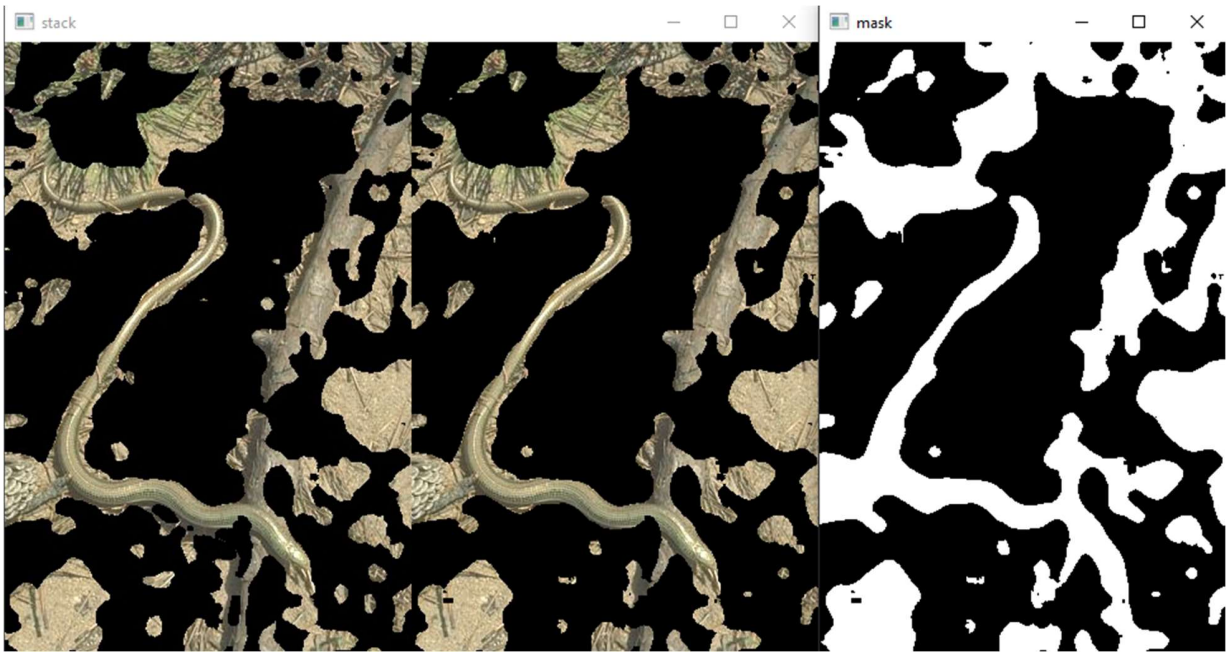
```
import cv2
import numpy as np

image = cv2.imread('kontrast.jpg')
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
blur = cv2.medianBlur(hsv, 15)
lower = np.array([20,42,84])
upper = np.array([34,109,255])

mask = cv2.inRange(blur, lower, upper)
res = cv2.bitwise_and(image,image, mask= mask)
cv2.imwrite('hsv.jpg',res)
cv2.imshow("mask ",mask)
cv2.imshow('stack', np.hstack([image, res]))
cv2.waitKey(0)
```



4- İşlem

3. İşlemin çıktısını 1. İşlemde bulunan program ile yılanın bir bölümünde alınan hsv değerleri ile maske oluşturdum. Orijinal resme 15x15 median filtresi uyguladım. Ardından resme, oluşturduğum maskeyi uyguladım.

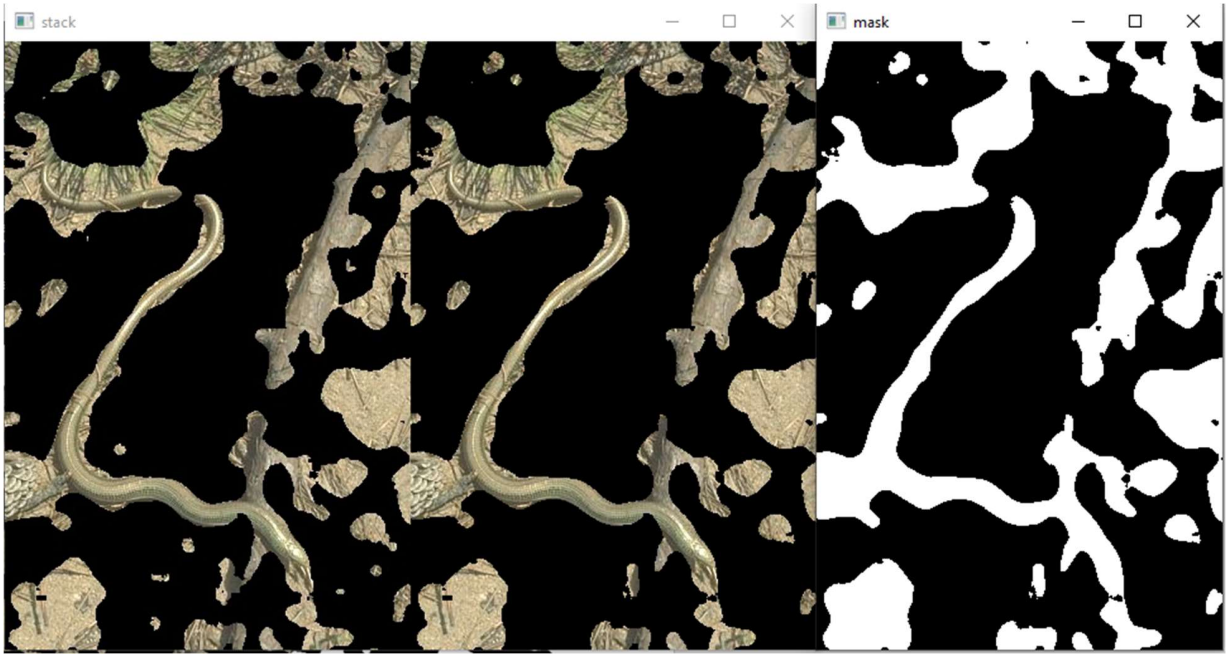
```
import cv2
import numpy as np

image = cv2.imread('hsv.jpg')
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
blur = cv2.medianBlur(hsv, 15)
lower = np.array([20,42,84])
upper = np.array([34,109,255])

mask = cv2.inRange(blur, lower, upper)
res = cv2.bitwise_and(image,image, mask= mask)
cv2.imwrite('hsv1.jpg',res)
cv2.imshow("mask ",mask)
cv2.imshow('stack', np.hstack([image, res]))
cv2.waitKey(0)
```



5- İşlem

4. İşlemin çıktısını 1. İşlemde bulunan program ile yılanın bir bölümünde alınan hsv değerleri ile maske oluşturdum. Orijinal resme 15x15 median filtresi uyguladım. Ardından resme, oluşturduğum maskeyi uyguladım.

```
import cv2
import numpy as np

image = cv2.imread('hsv1.jpg')
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
blur = cv2.medianBlur(hsv, 15)
lower = np.array([20,42,84])
upper = np.array([34,109,255])

mask = cv2.inRange(blur, lower, upper)
res = cv2.bitwise_and(image,image, mask= mask)
cv2.imwrite('hsv2.jpg',res)
cv2.imshow("mask ",mask)
cv2.imshow('stack', np.hstack([image, res]))
cv2.waitKey(0)
```



6- İşlem

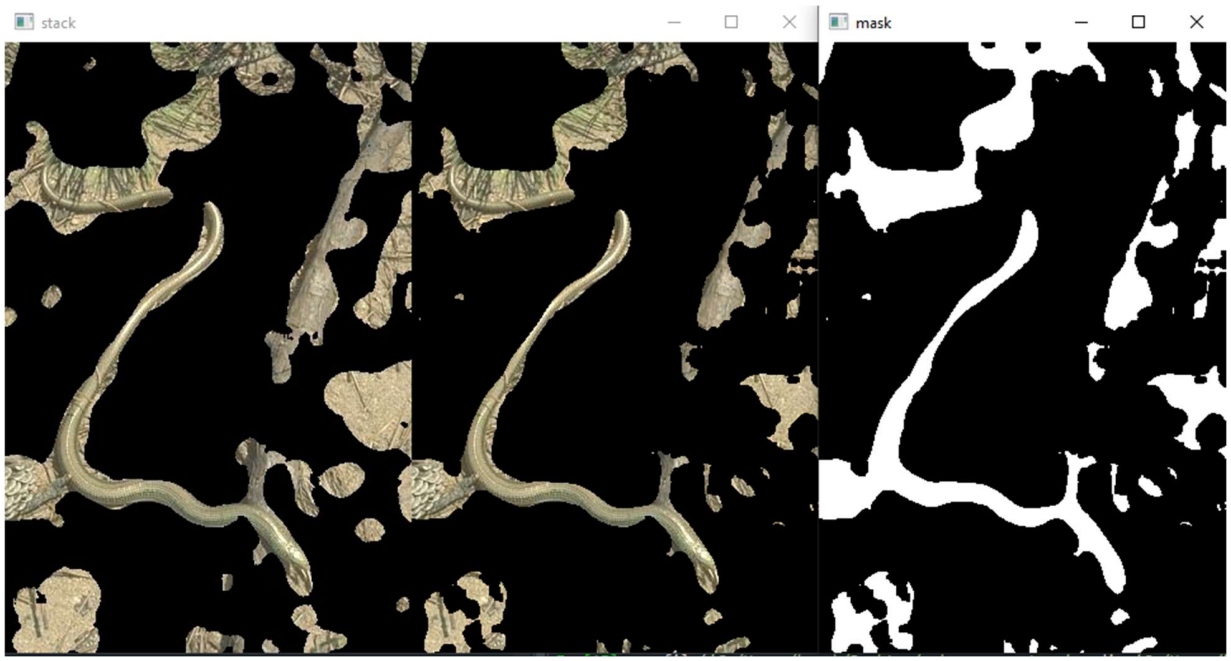
5. İşlemin çıktısını 1. İşlemde bulunan program ile yılanın bir bölümünde alınan hsv değerleri ile maske oluşturdum. Orijinal resme 15x15 median filtresi uyguladım. Ardından resme, oluşturduğum maskeyi uyguladım.

```
import cv2
import numpy as np

image = cv2.imread('hsv2.jpg')
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
blur = cv2.medianBlur(hsv, 15)
lower = np.array([20,42,84])
upper = np.array([34,109,255])

mask = cv2.inRange(blur, lower, upper)
res = cv2.bitwise_and(image,image, mask= mask)
cv2.imwrite('hsv3.jpg',res)
cv2.imshow("mask ",mask)
cv2.imshow('stack', np.hstack([image, res]))
cv2.waitKey(0)
```

7- İşlem

6. işlemin sonucunda oluşan resim için hsv değerlerini değiştirerek adımları uyguladım. Ek olarak 2x2 boyutlu kare yapı ile 2 iterasyonlu erozyon uyguladım.

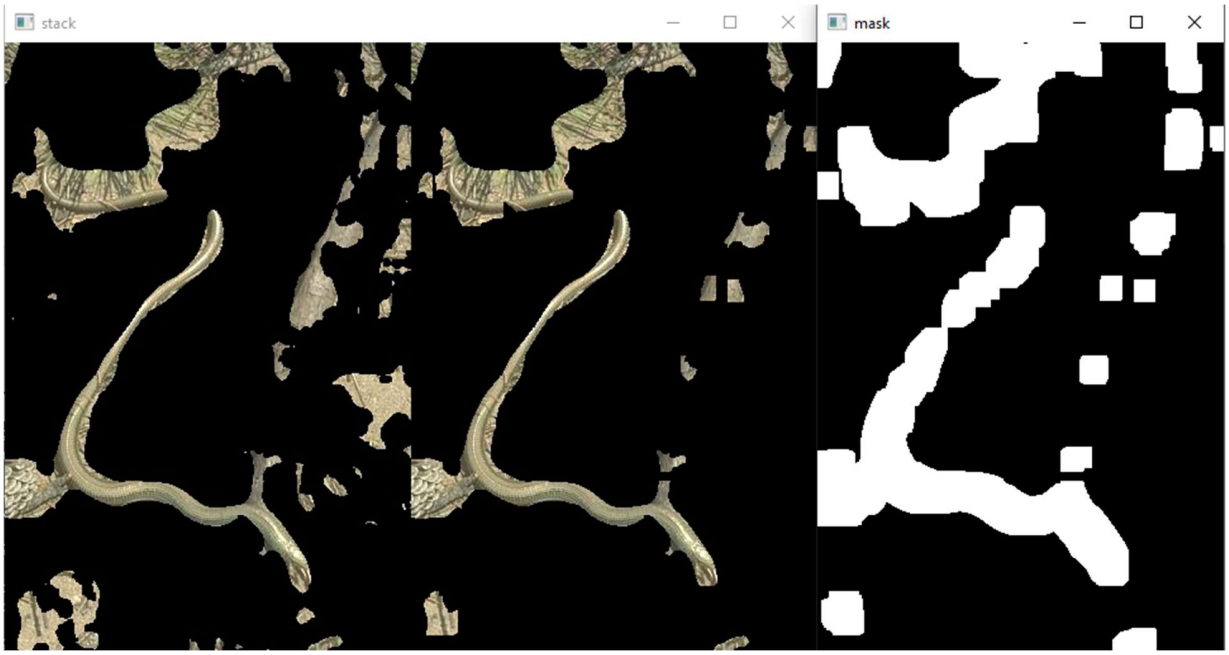
```
import cv2
import numpy as np

image = cv2.imread('hsv3.jpg')
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
blur = cv2.medianBlur(hsv, 15)
lower = np.array([21,41,89])
upper = np.array([34,102,255])
mask = cv2.inRange(blur, lower, upper)
kernel = np.ones((2,2),np.uint8)
mask = cv2.erode(mask,kernel,iterations = 2)
res = cv2.bitwise_and(image,image, mask= mask)
cv2.imwrite('hsv4.jpg',res)
cv2.imshow("mask ",mask)
cv2.imshow('stack', np.hstack([image, res]))
cv2.waitKey(0)
```



8- İşlem

7. işlem sonucunda oluşan resim için hsv değerlerini değiştirerek adımları uyguladım. Ek olarak 5x5 boyutlu kare yapı ile 4 iterasyonlu genişletme işlemi uyguladım.

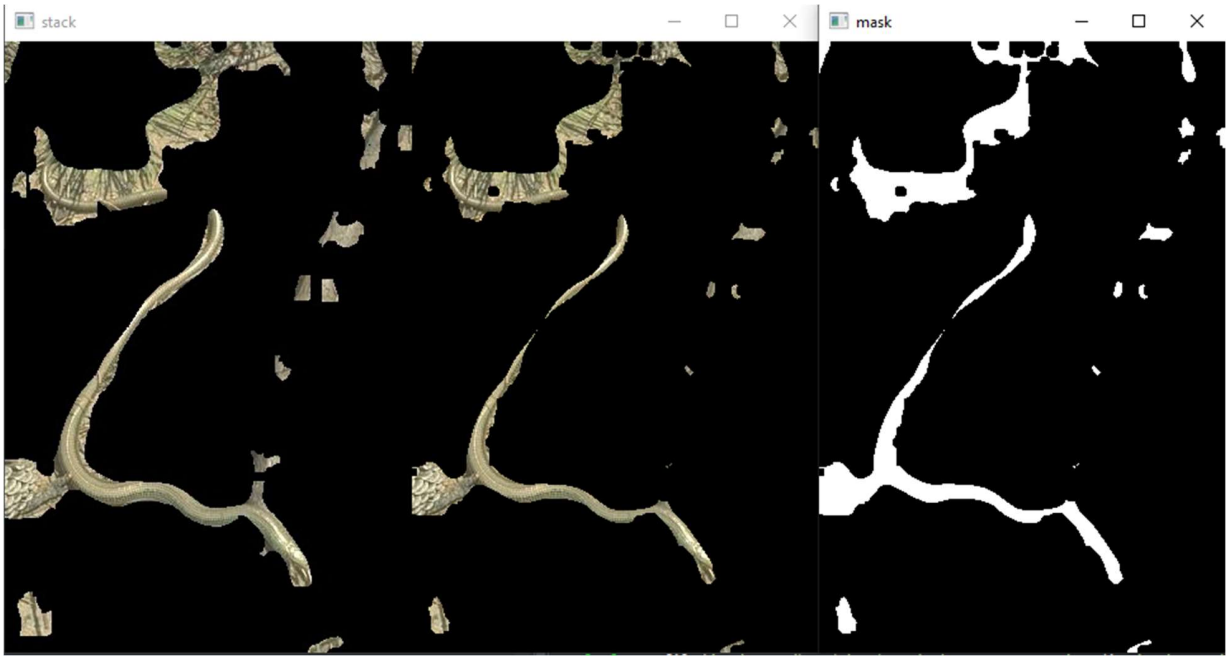
```
import cv2
import numpy as np

image = cv2.imread('hsv4.jpg')
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

blur = cv2.medianBlur(hsv, 15)

lower = np.array([23,36,94])
upper = np.array([38,83,193])

mask = cv2.inRange(blur, lower, upper)

kernel = np.ones((5,5),np.uint8)
mask = cv2.dilate(mask,kernel,iterations = 4)

res = cv2.bitwise_and(image,image, mask= mask)

cv2.imwrite('hsv5.jpg',res)
cv2.imshow("mask ",mask)
cv2.imshow('stack', np.hstack([image, res]))
cv2.waitKey(0)
```



9- İşlem

8. işlem sonucunda oluşan resim için hsv değerlerini değiştirerek adımları uyguladım. Ek olarak 3x3 boyutlu kare yapı ile 2 iterasyonlu erozyon işlemi uyguladım.

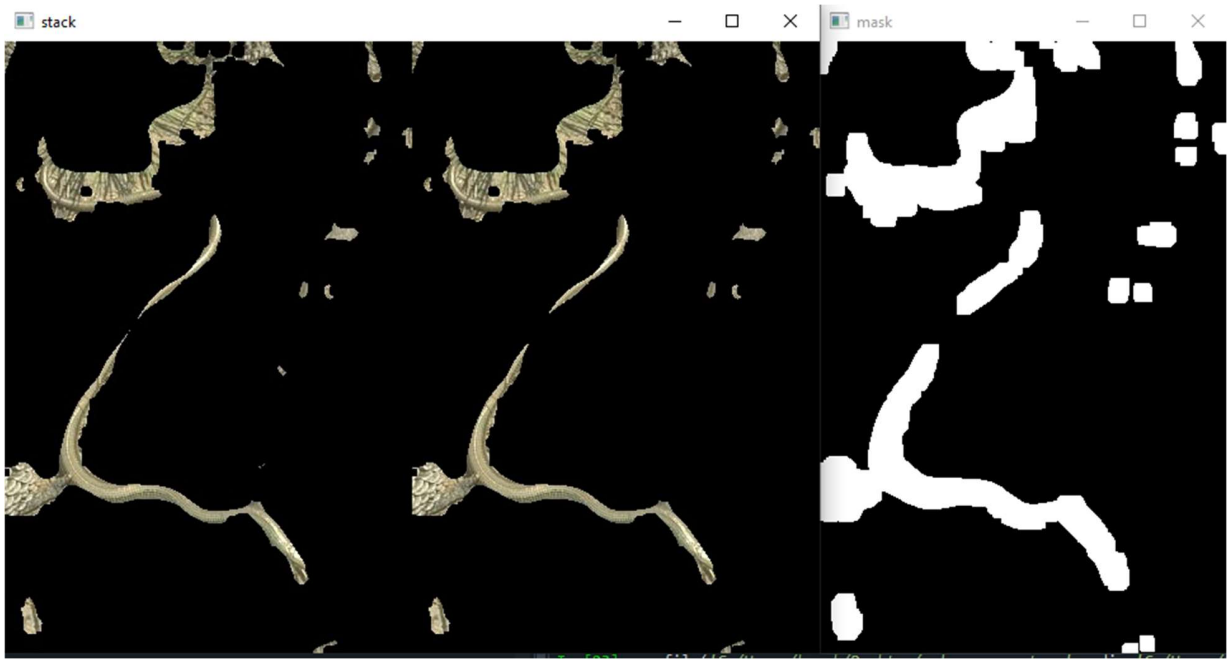
```
import cv2
import numpy as np

image = cv2.imread('hsv5.jpg')
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
blur = cv2.medianBlur(hsv, 9)

lower = np.array([21,41,84])
upper = np.array([35,106,255])

mask = cv2.inRange(blur, lower, upper)
kernel = np.ones((3,3),np.uint8)
mask = cv2.erode(mask,kernel,iterations = 2)
res = cv2.bitwise_and(image,image, mask= mask)

cv2.imwrite('hsv6.jpg',res)
cv2.imshow("mask ",mask)
cv2.imshow('stack', np.hstack([image, res]))
cv2.waitKey(0)
```



10- İşlem

9. işlem sonucunda oluşan resim için hsv değerlerini değiştirerek adımları uyguladım. Ek olarak 3x3 boyutlu kare yapı ile 5 iterasyonlu genişletme işlemi uyguladım.

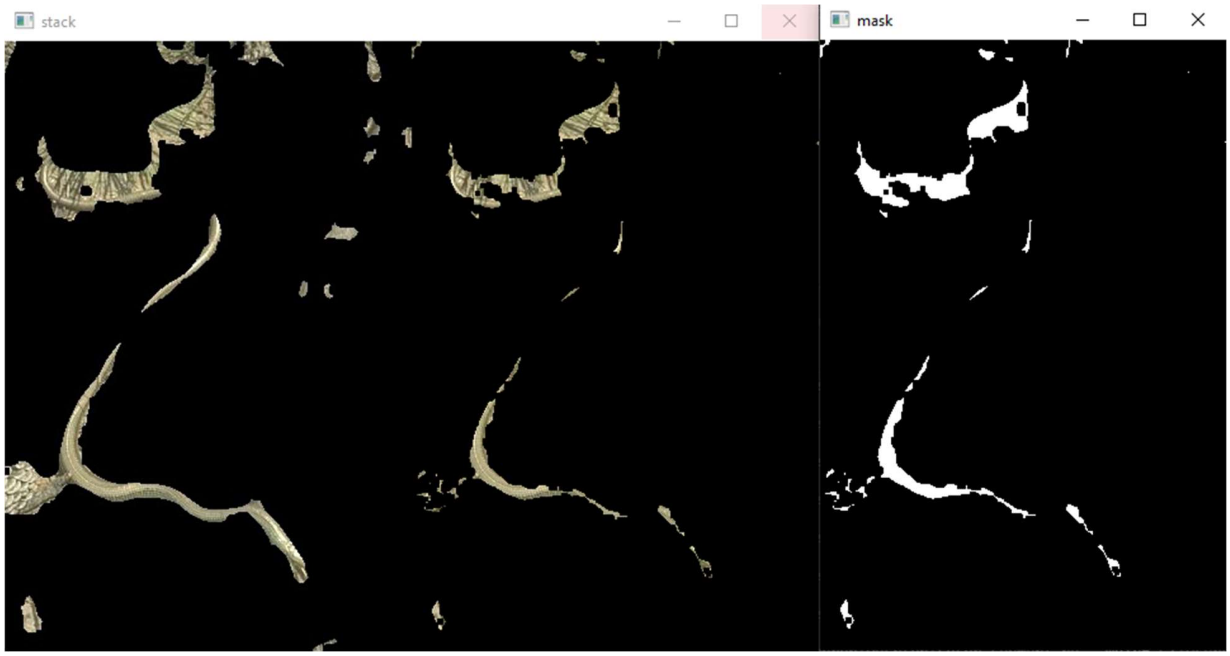
```
import cv2
import numpy as np

image = cv2.imread('hsv6.jpg')
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
blur = cv2.medianBlur(hsv, 5)

lower = np.array([22,50,100])
upper = np.array([34,99,235])

mask = cv2.inRange(blur, lower, upper)

kernel = np.ones((3,3),np.uint8)
mask = cv2.dilate(mask,kernel,iterations = 5)

res = cv2.bitwise_and(image,image, mask= mask)

cv2.imwrite('hsv7.jpg',res)
cv2.imshow("mask ",mask)
cv2.imshow('stack', np.hstack([image, res]))
cv2.waitKey(0)
```



11- İşlem

10. işlem sonucunda oluşan resim için hsv değerlerini değiştirerek adımları uyguladım. Ek olarak 4x4 boyutlu kare yapı ile 1 iterasyonlu erozyon işlemi uyguladım. Oluşan maskeyi kaydettim.

```
import cv2
import numpy as np

image = cv2.imread('hsv7.jpg')

hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

blur = cv2.medianBlur(hsv, 5)

lower = np.array([22,50,100])
upper = np.array([34,99,235])

mask = cv2.inRange(blur, lower, upper)
kernel = np.ones((4,4),np.uint8)
mask = cv2.erode(mask,kernel,iterations = 1)


res = cv2.bitwise_and(image,image, mask= mask)

cv2.imwrite('hsv8.jpg',res)
cv2.imwrite('mask.jpg',mask)
cv2.imshow("mask ",mask)
cv2.imshow('stack', np.hstack([image, res]))
cv2.waitKey(0)
```



12- İşlem

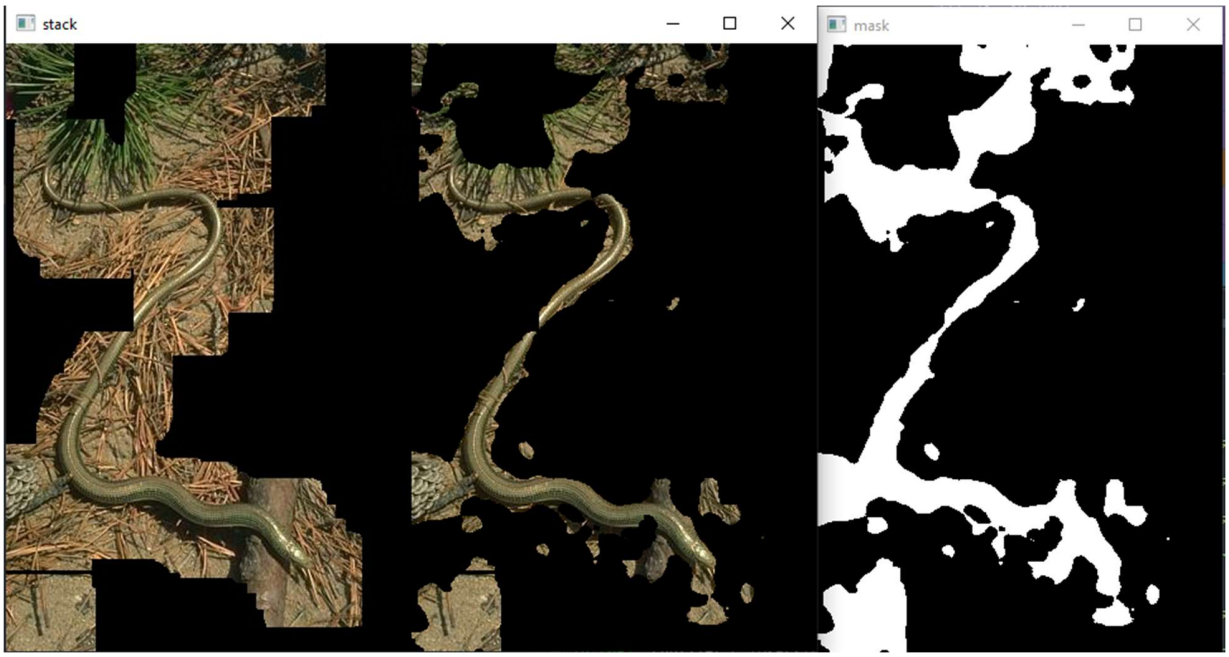
11. işlem sonucunda kaydettiğim maskeye 2x2 boyutunda kare yapı ile 1 iterasyonlu erozyon işlemi uyguladım. Ardından 4x4 boyutunda kare yapı ile 23 iterasyonlu genişletme işlemi uyguladım. Oluşan maskeyi, orijinal resme uyguladım.

```
import cv2
import numpy as np

image = cv2.imread('sayisal.jpg')
mask = cv2.imread('mask.jpg')

hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
lower = np.array([17,67,24])
upper = np.array([24,214,212])

kernel = np.ones((2,2),np.uint8)
mask = cv2.erode(mask,kernel,iterations = 1)

kernel = np.ones((4,4),np.uint8)
mask = cv2.dilate(mask,kernel,iterations = 23)

res = cv2.bitwise_and(image,mask, mask=None)

cv2.imwrite('hsv9.jpg',res)
cv2.imshow("mask ",mask)
cv2.imshow('stack', np.hstack([image, res]))
cv2.waitKey(0)
```

13- İşlem

12. işlem sonucunda oluşan resme hsv değerlerini değiştirerek adımları uyguladım.

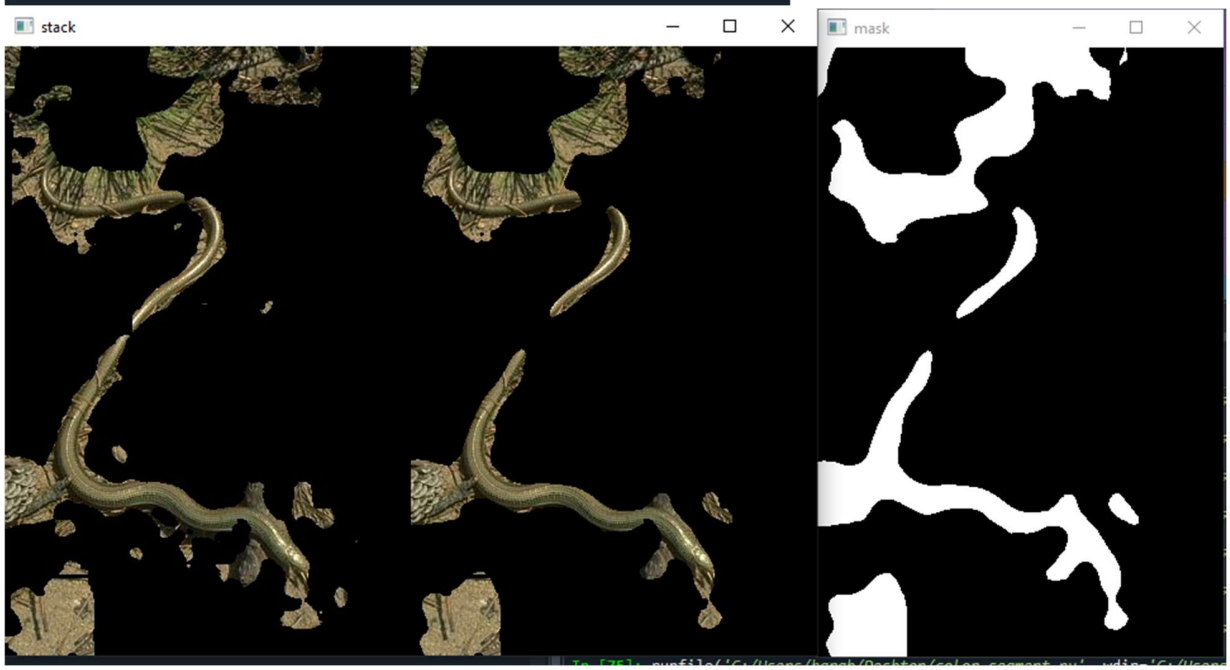
```
import cv2
import numpy as np

image = cv2.imread('hsv9.jpg')
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

blur = cv2.medianBlur(hsv, 15)

lower = np.array([20,56,34])
upper = np.array([34,255,207])

mask = cv2.inRange(blur, lower, upper)
res = cv2.bitwise_and(image,image, mask= mask)

cv2.imwrite('hsv10.jpg',res)
cv2.imshow("mask ",mask)
cv2.imshow('stack', np.hstack([image, res]))
cv2.waitKey(0)
```



14- İşlem

13. işlem sonucunda oluşan resme hsv değerlerini değiştirerek adımları uyguladım. Ek olarak 2x2 boyutunda kare yapı ile 2 iterasyonlu erozyon işlemi uyguladım.

```
import cv2
import numpy as np

image = cv2.imread('hsv10.jpg')
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

blur = cv2.medianBlur(hsv, 21)

lower = np.array([20,56,34])
upper = np.array([34,255,207])

mask = cv2.inRange(blur, lower, upper)


kernel = np.ones((2,2),np.uint8)
mask = cv2.erode(mask,kernel,iterations = 2)

res = cv2.bitwise_and(image,image, mask= mask)

cv2.imwrite('hsv11.jpg',res)
cv2.imshow("mask ",mask)
cv2.imshow('stack', np.hstack([image, res]))
cv2.waitKey(0)
```



15- İşlem

14. İşlem sonucunda oluşan resmi önce gri tona ardından siyah beyaza çevirdim. Ardından canny kenar bulma algoritması ile resmin kenarlarını buldum.

Bulduğum kenar bilgileri ile orijinal resimde işaretleme yaparak işlemleri bitirdim.

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

img = cv2.imread('hsv11.jpg')
img1 = cv2.imread('sayisal.jpg')
gray_image = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
(thresh, output2) = cv2.threshold(gray_image, 0, 255, cv2.THRESH_BINARY)
output2 = cv2.Canny(output2, 0, 255)
cv2.imwrite('son.jpg',output2)
plt.imshow(output2)
plt.show()

lines = cv2.HoughLinesP(output2, 1, np.pi/180,30)
for line in lines:
    x1,y1,x2,y2 = line[0]
    cv2.line(img1,(x1,y1),(x2,y2),(0,255,0),4)
plt.imshow(img1)
cv2.imwrite('son1.jpg',img1)
```