### https://medium.com/@rukiyealimci/opencv-ile-g%C3%B6r%C3%BCnt%C3%BC-i%C5%9Fleme-y%C3%B6ntemleri-bb46cfa877fa

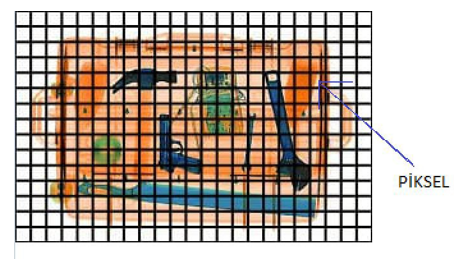
### OpenCV ile Görüntü işleme yöntemleri

Bu yazıda OpenCv kütüphanesi ile belli filtrelerin kullanımını ve filtrelerin amacından bahsediyor olacağım. Detaylı bilgileri <https://opencv.org/> ya da Rafael C. GONZALES’in Digital Image Processing kitabından rahatlıkla ulaşabilirsiniz.

Görüntü analizi; insan görme sisteminin işleyişini taklit ederek nesnelere ait örüntüler üzerinde hesaplamalar yapılarak anlamlı ve faydalı bilgi üretilmesidir.

Görüntü analiz sürecinde hatasız ve gerçekçi bir sonuç elde edilebilmesi için görüntünün bazı özellikleri işlenerek daha belirgin ve anlaşılır hale getirilmesi gerekir. Sayısal görüntü işleme; temelde parlaklık, kontrast, renk vb. görüntüye ait bilgilerin değiştirilmesi, görüntü kirliliklerin giderilmesi, detayların daha belirgin hale getirilmesi gibi görüntü kalitesinin iyileştirilmesine yönelik işlemleri ifade eder.

Görüntüler piksel olarak adlandırılan küçük karelerin birleşiminden oluşur.Piksel topluluklarından oluşan dijital görüntülerde piksel sayısınına göre (inc ve santimetre değerlerinden bağımsız) olarak işlem yapılır. Piksel sayısı ve çözünürlük dijital görüntüde önemli ifadelerdir.

Piksel

Görüntü işlemedeki ilk adım, sayısal görüntünün sayısal kamera ile elde etme işlemidir.

Ön İşleme: Sayısal resim elde edildikten sonraki aşama ise ön-işlemedir. Ön-işleme, elde edilen sayısal görüntüyü kullanmadan önce daha başarılı bir sonuç elde edebilmek amacıyla görüntünün bazı ön işlemlerden geçirilmesidir. Bu işlemler temel olarak;

-Görüntü iyileştirme (image enhancement),

-Görüntü onarma (image restoration) ve

-Görüntü sıkıştırma (image compression)

Görüntünün elde edilmesi ve ön işlemeye tabi tutulması düşük seviyeli görüntü işleme olarak adlandırılır.

Görüntü işleme temel olarak aşağıdaki üç adımı içermektedir:

-Görüntü elde etme araçları aracılığıyla görüntünün içe aktarılması

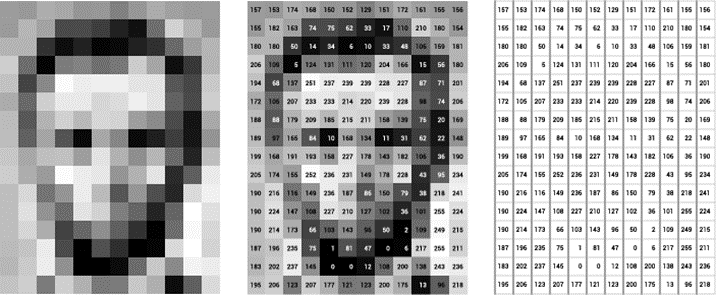
-Görüntüyü analiz etme ve manipüle etme;

-Görüntü analizine dayalı olarak sonucun değiştirilebileceği görüntü veya rapor çıktısı.

Görüntü işleme için kullanılan analog ve dijital görüntü işleme olmak üzere iki tür yöntem vardır.Çıktı ve fotoğraf gibi basılı kopyalar için analog görüntü işleme kullanılabilir. Görüntü analistleri, bu görsel teknikleri kullanırken çeşitli yorumlama temellerini kullanır. Bu yazıda sayısal görüntü işleme ele alınacaktır.

Görüntülerin doğal özelliklerini işlemek ve işlemek için kullanılan teknik prosedürü basitleştirmek için, tek tek piksellerden oluşan bir görüntüyü bir fonksiyonu olarak tanımlanabilir. Her pikselin de kendi değeri bulunmaktadır. Gri tonlamalı bir görüntü için, her piksel 0 ile 255 arasında bir yoğunluğa sahip olacaktır, 0 siyah ve 255 beyaz olmak üzere f (x, y) tanımlandığında, piksel konumundaki f (x, y) görüntünün yoğunluğunu verir:

f: [a, b]x [c, d] → [0,255]

gri tonlamalı bir görüntüde piksel değerlerinin tablosu [1]

Renkli bir görüntü ise Kırmızı, Yeşil ve Mavi (RGB) kombinasyonundan oluşmaktadır. Bu nedenle, görüntünün her pikseli üç kanala sahiptir ve 1𝑥3 vektörü olarak temsil edilir. Üç rengin 0'dan 255'e kadar tamsayı değerleri olduğundan, toplam 256 ∗ 256 ∗256 = 16.777.216 kombinasyon veya renk seçeneği bulunmaktadır.

RGB formatda bir görüntünün piksel değerinin gösterilmesi

Bir görüntünün temel özellikleri anlaşıldığına göre filtrelerden bahsedebilriz.

### ****Görüntü işlemede filtreler****

Nesnenin ortaya çıkması için belirli filtreler uygulanmalıdır. Filtreler genelde 3x3 matrislerdir. Bunlara çekirdek matris (kernel matrix) denir. Filtreleme işlemi ise kısaca bu matrislerin görüntü üzerinde gezdirilip, üst üste gelen değerlerin çarpımları ve oluşan tüm çarpımların toplamlarının filtre elemanlarının toplamına bölünmesi ile gerçekleştirilmiş olur. Her seferinde sadece merkez pikselin, yani ortadaki pikselin değeri değiştirilir. Bu işlemler sırayla tüm pikseller için yapılır ve görüntü güncellenir. Bu yazıda temel filtreler 2 ana başlık altında geliştirilecektir.

1. Nokta (piksel) bazlı işlemler
2. Morfolojik işlemler

### ****Nokta (piksel) bazlı işlemler****

Çıkış görüntüsünün bir pikselini oluşturmak için, giriş görüntüsünün bir pikseli üzerinde yapılan işlemlerdir. Kendisine komşu olan piksellerden bağımsız olarak her pikselin kendi renk bilgisinin değiştirilmesiyle yapılır.

**Gri seviye dönüştürücü**

RGB Renk Uzayı: Bu renk uzayı Red Green Blue yani kırmızı, yeşil ve mavi renklerin baş harfi ile adlandırılmıştır. Renkler bir küp olarak tanımlanır bu tanımla sayesinde 3 değişkenli bir dizi elde edilir. Bu dizi elemanları olan hücreler yani pikseller, bir rengi tutabilmek için 3 renk olan kırmızı, yeşil ve mavinin belirli yoğunlukta karıştırılması ile elde edilen renk kodunu tutarlar[3].

Spesifik olarak bir renk, kırmızı, yeşil ve mavi için 0'dan 255'e kadar üç tamsayı değeri kullanılarak tanımlanır; burada sıfır değeri koyu ve 255 değeri parlak anlamına gelir. Değerler verildiğinde, bu üç temel rengi değerlerine göre ağırlıklandırarak karıştırdığımızda nihai renk tanımlanır.

Gri tonlamalı görüntüler siyah, beyaz ve aradaki tüm gri tonlarında işlenir. Herhangi bir gri değerin RGB kodlaması, üç eşit sayıdan oluşan bir kümedir, yani (x, x, x), burada x, 0 ile 255 arasında bir tamsayıdır. Örneğin, beyaz (255,255,255), siyah (0,0,0) ve orta gri (127,127,127). Sayılar ne kadar yüksek olursa, gri o kadar açık olur.

# -\*- coding: utf-8 -\*-  
"""  
Created on Sat Mar 4 14:25:25 2023  
  
@author: rukiy  
"""  
  
import cv2  
# RGB formatındaki resmin Siyah beyaz formata çevirilmesi  
image = cv2.imread("input.jpg", cv2.IMREAD\_GRAYSCALE)  
# ya da  
İmage\_gray = cv2.imread("input.jpg", 0)  
# ya da  
gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)  
cv.imshow("image", image)  
cv.imshow("İmage\_gray", İmage\_gray)  
cv.imshow("gray", gray)  
#her üç dönüşümün sonucunun aynı olduğu gözlemlenir.  
  
cv.waitKey(0)  
cv.destroyAllWindows()

**Görüntü Eşikleme Threshold**

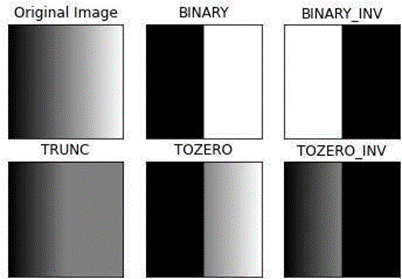
Eşikleme işlemi, bir görüntüyü ikili (Binary) görüntüye çevirme işlemi yani bir görüntüde bulunan her bir pikseli siyah ya da beyaz piksel ile değiştirme işlemidir.

Görüntü yoğunluğu sabit bir 𝑇’den (eşik değeri) küçükse o piksel siyah piksel olarak sınıflandırılırken görüntü yoğunluğu, sabit bir T’den (eşik değeri) büyükse o piksel beyaz piksel olarak sınıflandırılır. Bu işlem medikal görüntü işleme, doku analizi, nesne algılama, parmak izi ve belge tarama ve analizi gibi işlemlerde temel yapı niteliğindedir. Buradaki kritik nokta ise uygun bir eşik değeri kullanarak görüntü içerisindeki nesneleri arka plandan ayırmaktır. Eğer 𝑇 değeri olması gereken daha fazla ya da az olursa görüntünün içerdiği anlamlı bilgi ciddi miktarda zarar görecektir. Bunun sonucu olarak, elde edilen ikili sonuç görüntüsü bir problemin çözümü için geliştirilen sistemin başarısını olumsuz etkileyecektir.



En basit eşikleme yöntemleri, görüntü yoğunluğu 𝐼𝑖,𝑗 bazı sabit sabitlerden daha azsa görüntüdeki her pikseli siyah bir pikselle veya görüntü yoğunluğu bu sabitten büyükse beyaz bir pikselle değiştirir[2]

𝐼𝑖,𝑗 < 𝑇

Eşik değerine göre eşitlenmiş thereshold

Bu kod parçacığı ile farklı threshold yöntemlerinin ve eşik değerleinin sonuçlara etkisi gözlemlenebilir. Yukarıdaki görselin mantığını anlamanızı sağlayacaktır.

# -\*- coding: utf-8 -\*-  
"""  
Created on Sat Mar 4 14:25:25 2023  
  
@author: rukiy  
"""  
  
import cv2 as cv  
from matplotlib import pyplot as plt  
  
image= cv.imread('image.jpg')  
image=cv.cvtColor(image, cv.COLOR\_BGR2GRAY) #gri seviyeye dönüştürmek için kullanılır  
  
ret, thresh1=cv.threshold(image,127,255,cv.THRESH\_BINARY)  
ret, thresh2=cv.threshold(image,50,255,cv.THRESH\_BINARY)  
ret, thresh3=cv.threshold(image,127,127,cv.THRESH\_BINARY)  
  
  
ret, thresh4= cv.threshold(image, 127, 255,cv.THRESH\_BINARY\_INV)  
  
ret, thresh5= cv.threshold(image, 250, 255,cv.THRESH\_MASK) # hepsini kapartıyor  
# ret, thresh6= cv.threshold(image, 127, 255,cv.THRESH\_OTSU)  
# ret, thresh7= cv.threshold(image, 127, 255,cv.THRESH\_TOZERO)  
# ret, thresh8= cv.threshold(image, 127, 255,cv.THRESH\_TOZERO\_INV)  
# ret, thresh9= cv.threshold(image, 127, 255,cv.THRESH\_TRIANGLE)  
# ret, thresh10= cv.threshold(image, 127, 255,cv.THRESH\_TRUNC)  
  
plt.subplot(231),plt.imshow(image),plt.title('orijinal')  
plt.subplot(232),plt.imshow(thresh1, 'gray'),plt.title('127,255,cv.THRESH\_BINARY')  
plt.subplot(233),plt.imshow(thresh2, 'gray'),plt.title('50,255,cv.THRESH\_BINARY')  
plt.subplot(234),plt.imshow(thresh3, 'gray'),plt.title('127,127,cv.THRESH\_BINARY')  
plt.subplot(235),plt.imshow(thresh4, 'gray'),plt.title('THRESH\_BINARY\_INV')  
plt.subplot(236),plt.imshow(thresh5, 'gray'),plt.title('THRESH\_MASK')  
  
  
cv.destroyAllWindows()

**Histogram**

Histogram, görüntü yoğunluğu dağılımının istatistiksel karakteristiğini inceleme fırsatı sunan bir araçtır. Geleneksel boyamanın ve klasik fotoğrafçılığın prensiplerinden biri şudur, zengin ton aralığı ile resimlerin kontrastı artırılmalıdır. Renkli bir görüntü içerisinde kontrast ve parlaklık gibi renk değerlerini içermektedir. Histogram manipülasyonu görüntü iyileştirme için etkili bir şekilde kullanılmaktadır[4].

Gri seviyeli bir resimde histogramı gözlemlemek için çubuk grafiği kullanılır. Yatay eksen gri düzey değerlerini gösterir. Sıfırdan başlar ve gri seviye sayısına kadar gider. ( 0–256 arasındaki değerleri içerir) Her bir dikey çubuk görüntüde oluşan gri seviye oluşum sayısını gösterir. Her bir gri seviyesinden ne kadar bulunduğunu gözlemlememizi sağlar.

# -\*- coding: utf-8 -\*-  
"""  
Created on Sat Mar 4 14:25:25 2023  
  
@author: rukiy  
"""  
  
import cv2   
from matplotlib import pyplot as plt  
  
image= cv2.imread("image.jpg",0)  
# Histogram değerinin hesaplanması  
histogram = cv2.calcHist([image], [0], None, [256], [0,  
256])  
  
  
# Grafik olarak çizdirilmesi  
plt.plot(histogram)  
plt.show()

Bu kod gri tonlamalı bir resmi, cv2.calcHist() kullanarak histogram değerlerini hesaplayacak ve ardından histogramı çizmek için matplotlib’i kullanacaktır. cv2.calcHist() işlevi birkaç bağımsız değişken alır: giriş görüntüsü, histogramın hesaplanacağı kanal (bu durumda, gri tonlamalı görüntüyü kullanıyoruz ve kanalı 0'a ayarlıyoruz), görüntüye uygulanacak bir maske (biz maske kullanmıyoruz, bu yüzden None olarak ayarladık), histogramda kullanılacak bölme sayısı (her olası yoğunluk değeri için bir tane olmak üzere 256 bölme kullanıyoruz) ve dikkate alınacak yoğunluk değerleri aralığı (kullanıyoruz 0–256 aralığı).Bu, histogram değerlerinin basit bir çizgi grafiğini oluşturacaktır.

**Histogram Eşitleme**

Bir görüntüdeki renk değerlerinin belli bir yerde yığılmış olmasından kaynaklanan renk dağılımı bozukluğunu çözmek için kullanılan bir işlemdir.

Histogram eşitleme yöntemi ile görüntünün histogramının daha eşit olarak dağıtılması için görüntüdeki piksellerin yoğunluk değerlerini ayarlayarak bir görüntünün kontrastını arttıran bir görüntü işleme yöntemidir. Bu, görüntüdeki ayrıntıları daha görünür hale getirmeye ve görüntünün genel kontrastını iyileştirmeye yardımcı olur [5].

# -\*- coding: utf-8 -\*-  
"""  
Created on Sat Mar 4 14:25:25 2023  
  
@author: rukiy  
"""  
  
import cv2 as cv  
  
image= cv.imread('image.jpg')  
# Histogram eşitleme  
equalized\_image = cv2.equalizeHist(image)  
  
cv2.imshow("equalized\_image",equalized\_image)  
  
cv.waitKey(0)  
cv.destroyAllWindows()

**Adaptive Histogram Eşitleme**

Uyarlanabilir histogram eşitleme (AHE), tüm görüntü boyunca küresel olarak değil, histogram eşitlemesini yerel olarak uygulayarak bir görüntünün kontrastını arttıran bir görüntü işleme yöntemidir [5]. Bu, görüntünün kontrastını geliştirirken görüntünün genel yapısını korumaya yardımcı olur.

AHE, giriş görüntüsünü küçük bölgelere veya “fayanslara” bölerek ve her bir karolara ayrı ayrı histogram eşitleme uygulayarak çalışır. Bu, küresel aydınlatma değişikliklerinin etkilerinin azaltılmasına yardımcı olur ve görüntünün genel yapısını korumaya yardımcı olur [6].

# -\*- coding: utf-8 -\*-  
"""  
Created on Sat Mar 4 14:25:25 2023  
  
@author: rukiy  
"""  
  
import cv2 as cv  
  
# adaptive histogram eşitleme clip limit --> 2.0  
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))  
  
image= cv.imread('image.jpg')  
  
# Apply adaptive histogram equalization   
equalized\_image = clahe.apply(image)  
  
cv2.imshow("equalized\_image",equalized\_image)  
  
cv.waitKey(0)  
cv.destroyAllWindows()

Bu kod bir giriş gri tonlamalı görüntü yükleyecek, 2.0 klip sınırına ve (8,8) karo ızgara boyutuna sahip bir Clahe nesnesi oluşturacak ve daha sonra clahe.apply(image) yöntemini kullanarak giriş görüntüsüne adaptive histogram eşitleme uygulayacak. Klips sınırı, görüntüye uygulanan maksimum kontrast geliştirmeyi kontrol eder ve karo ızgara boyutu, görüntünün histogram eşitleme amacıyla bölündüğü karoların boyutunu belirtir.

### Morfolojik işlemler

Bir görüntüdeki özelliklerin şekli (veya morfolojisi) ile ilgilenen bir dizi görüntü işleme tekniğini açıklar. Görüntüdeki istenmeyen parazit ve gürültüleri ortadan kaldırmakta kullanılır.

Morfolojik görüntü işleme, bir görüntüdeki özelliklerin şekli veya morfolojisi ile ilgili non-lineer işlemlerin bir koleksiyonudur.

Morfolojik işlemler sayısal değerlerine değil, yalnızca piksel değerlerinin göreli sıralamasına dayanır ve bu nedenle özellikle ikili görüntülerin işlenmesi için uygundur. Morfolojik işlemler, bir yapılandırma elemanı adı verilen küçük bir pencereye sahip bir görüntüyü tarar ve karşılık gelen piksel komşuluğu ile karşılaştırılır. Bazı işlemler, elemanın yakın piksellere “uyup uymadığını” kontrol ederken, bazı işlemler “isabet edip etmediğini” veya “kesişip kesişmediğini” kontrol eder [4].

A, görüntü; B, yapılandırma elemanı olmak üzere; A Å B ↔Genişletme

A Q B ↔ Aşındırma

A o B = ( A Q B) Å B ↔ Açma işlemi

A ● B = ( A Å B ) Q B ↔ Kapama işlemi

Genişletme ve Aşındırma (Erosion ve Dilation)

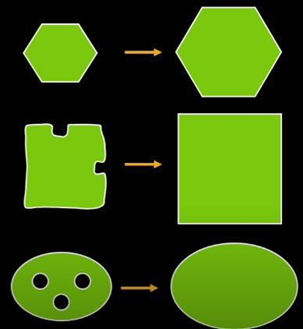
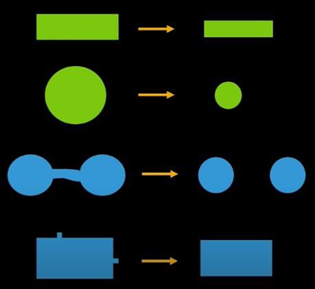
Yayma işlemidir. İkili imgedeki nesneyi büyütmeye ya da kalınlaştırmaya yaramaktadır. Sayısal bir görüntüyü genişletmek görüntüyü yapısal elemanla kesiştiği bölümler kadar büyütmek demektir. Genişletme, ikili bir görüntünün 1'lerinin bağlı kümelerini genişletir. Ayrıca özellikleri büyütür ve delikleri doldurur. İkili bir görüntüde, komşu piksellerden herhangi birinin 1 değerine sahipse 1'e ayarlanır[5].

𝐴 ⊕ 𝐵 = ⋃ 𝐴𝑏

𝑏∈𝐵

Burada 𝐴𝑏 b tarafından çevrilmiş A dır [7].

Görüntüdeki nesneleri küçültmeye ya da inceltmeye yarayan işlemdir. Aşındırma işlemi genişletme işleminin tersi gibi düşünülebilir. Aşındırma işlemi görüntünün içerisindeki nesneleri ufaltır, delik varsa genişletir ve bağlı nesneler ayrılma eğilimi gösterir.

genişletme işlemiaşındırma işlemi

# -\*- coding: utf-8 -\*-  
"""  
Created on Sat Mar 4 14:25:25 2023  
  
@author: rukiy  
"""  
  
import cv2 as cv  
  
image= cv.imread('image.jpg')  
  
  
# Erosion ve dilation işlemleri içn kernelin oluşturulması  
 kernel = cv2.getStructuringElement(cv2.MORPH\_RECT, (3,3))  
  
# Resme erosion işleminin uygulanması   
eroded\_image = cv2.erode(image, kernel)  
# Resme dilation işleminin uygulanması   
dilated\_image = cv2.dilate(image, kernel)  
  
cv2.imshow("eroded\_image",eroded\_image)  
cv2.imshow("dilated\_image",dilated\_image)  
  
cv.waitKey(0)  
cv.destroyAllWindows()

**Açma ve kapama işlemi (Opening ve Closing)**

Morfolojik görüntü işleme yönteminde temel olarak kullanılan iki işlem bulunmaktadır: Genişletme ve aşındırma. Diğer morfolojik işlemler bu iki işlem kullanılarak farklı formüllerde elde edilir

Açılış, erozyonun genişlemesidir,[8] genişletme ve aşındırma işlemlerini ardışık uygulanmasıyla gerçekleşir. Bu işlemle birbirine yakın iki nesne görüntüde fazla bozulmaya neden olmadan ayrılmış olmaktadır.

𝐴 ∘ 𝐵 = (𝐴 ⊖ 𝐵) ⊕ 𝐵

Kapama, dilatasyonun erozyonudur, [8] Aşındırma ve genişletme işleminin ardılış uygulanmasıyla yapılan bir işlemdir. Bu işlemle birbirine yakın iki nesne görüntüde fazla bozulmaya neden olmadan bağlanmış olmaktadır.

𝐴 ⋅ 𝐵 = (𝐴 ⊕ 𝐵) ⊖ 𝐵

# -\*- coding: utf-8 -\*-  
"""  
Created on Sat Mar 4 14:25:25 2023  
  
@author: rukiy  
"""  
  
import cv2 as cv  
  
image= cv.imread('image.jpg')  
  
# Erosion ve dilation işlemleri içn kernelin oluşturulması   
kernel = cv2.getStructuringElement(cv2.MORPH\_RECT, (3,3))  
  
# Resme açma işleminin uygulanması  
opened\_image = cv2.morphologyEx(image, cv2.MORPH\_OPEN, kernel)  
  
# Resme kapama işleminin uygulanması  
closed\_image = cv2.morphologyEx(image, cv2.MORPH\_CLOSE, kernel)  
  
cv2.imshow("opened\_image",opened\_image)  
cv2.imshow("closed\_image",closed\_image)  
  
cv.waitKey(0)  
cv.destroyAllWindows()

Python ve OpenCV’de açılış ve kapanış yapmak için, erozyon ve dilatasyonu tek bir işlemle birleştiren cv2.morphologyEx() işlevini kullanabilirsiniz. İşlem türünü fonksiyon içerisine parametre olarak alır ve uygular.

[1]” smartera,”[online] <https://smartera.com.tr/gercek-zamanli-nesne-takibireal-time-object-detection-w-yolo-python/>

[2] “lidertek”[online] <https://www.lidertek.com.tr/panel/uploads/product_v/xray1.jpg>

[3]Kumar, T. Verma,K. “A Theory Based on Conversion of RGB image to Gray image “ 2010

[4]Rafael C. GONZALES, Richard E. WOODS, 2008. Digital Image Processing Third Edition, Pearson Prentice Hall, New Jersey 07458

[5]”Towards Data Science,” [Online]. Available: [https://towardsdatascience.com/histogram-equalization-5d1013626e64.](https://towardsdatascience.com/histogram-equalization-5d1013626e64.%20%20%5b25) [25Eylül 2022].

[6]”Towards Data Science,” [Online]. Available: [https://towardsdatascience.com/histogram-equalization-5d1013626e64.](https://towardsdatascience.com/histogram-equalization-5d1013626e64.%20%20%5b25) [25Eylül 2022].

[7]”Wikipedia,” [Online]. Available: [https://en.wikipedia.org/wiki/Dilation\_(morphology).](https://en.wikipedia.org/wiki/Dilation_%28morphology%29.)

[8]] J. Serra, “Image Analysis and Mathematical Morphology”.