

Görüntü İşleme Vize Notları

Bu yazı **MIT** lisanslıdır. Lisanslar hakkında bilgi almak için [buraya](#) bakmanda fayda var.

~ Yunus Emre AK ©

Döküman Renklendirme Yapısı

PDF Başlığı

Ana Başlıklar

Alt Başlıklar

İç Başlıklar

En İç Başlıklar

Tablo Başlığı

Bağlantılar

Değişmez ifadeler

Formüller

Önemli notlar

Terimsel ifadeler

Yorum satırları




İçerikler

- Ders Notlarım Hakkında
- Sayısal Görüntü Örnekleme ve Niceleme, İkili Görüntü İşleme
 - Sayısal Görüntü
 - Siyah-Beyaz Görüntü
- Lineer Filtreleme ve Kenar Belirleme
 - Kenar Belirleme (Edge Detection)
 - Kenar Belirleme Sorunları
 - Kenar Belirleme Yöntemleri
 - Gradyan Tabanlı Kenar Belirleme
 - Laplasyan Tabanlı Kenar Belirleme
 - Marr-Hilderth Kenar Belirleme
 - Canny Kenar Belirleme
 - Gürültü (Noise)
 - Gürültülü Engelleme
 - Frekans Kavramı
 - Lineer Filtreler
 - Alçak Geçirgen Filtreler
 - Yüksek Geçirgen Filtreler
 - Guassian Filtre
 - Laplasyan Fitre
 - LoG (Laplasyan of Guassian)
 - Medyan Filtre
 - Temel Görüntü İşlemleri
- Renk ve Geometrik Dönüşümler
 - Renk Formatları
 - RGB
 - Perspektif İzdüşüm
 - Gemometrik Dönüşümler
 - Homojen Koordinatlar
- Görüntü İyileştirme Metodları
 - Histogram Germe
 - Histogram Eşitleme
 - Python'da Histogram Germe İşlemi
 - Python'da Histogram Eşitleme
- Harici Bağlantılar

Ders Notlarım Hakkında

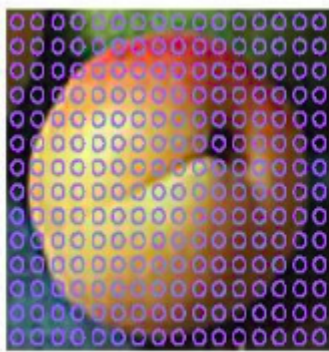
- GI05, GI04 hakkında not alınmıştır

Notlar tam değildir, sorumluluk kabul etmem 

Sayısal Görüntü Örnekleme ve Niceleme, İkili Görüntü İşleme

Sayısal Görüntü

- İkili (*binary*) görüntü
- Gri Ölçekli (*gray scale*) görüntü
- Renkli (*colour*) görüntü



Gerçek resim



```
115 81 10 30 56 12 34 30 1 70 79 21 145 150 52 136 143 85 115 120 41 128 143 50 85
116 11 14 90 14 85 87 23 90 74 23 73 82 19 67 70 21 40 46 7 33 39 9 84 54 19
42 27 6 19 10 3 56 82 29 102 101 41 206 88 83 204 15 54 197 62 63 179 83 48 158 62
48 140 49 40 52 66 21 66 69 11 40 21 17 25 37 0 24 29 0 83 50 15 2 0 1 13 14
8 243 173 101 231 140 80 238 142 89 233 143 80 210 126 79 114 88 48 152 89 15 123 51
27 104 41 23 55 45 9 36 27 0 28 26 2 29 26 1 40 28 14 13 15 1 224 167 152 240
114 89 227 178 18 227 176 87 233 177 94 213 149 78 196 123 57 141 72 31 138 53 22 121
62 22 126 50 24 181 48 58 140 21 1 12 5 0 14 18 11 5 0 0 237 176 85 244 208 123
241 236 144 238 222 147 221 196 100 315 170 77 190 135 82 134 83 50 76 35 7 113 56 28
156 83 58 107 52 21 31 14 7 9 6 0 20 14 12 255 214 112 242 215 108 248 227 123 239
232 152 229 230 123 232 163 98 208 182 84 176 133 47 142 90 32 39 19 27 89 53 21 171
116 48 114 64 29 75 49 24 10 9 5 11 16 9 237 180 42 249 221 122 241 225 128 240 219
126 240 166 98 218 173 99 180 135 33 218 186 78 189 164 83 136 154 65 112 86 37 181 153
80 122 74 26 60 51 10 19 37 47 16 37 32 223 177 83 126 288 105 243 216 125 238 208
115 225 186 83 228 204 98 224 220 123 210 184 189 182 188 82 150 86 40 118 73 28 148 134
48 109 89 24 75 48 10 27 33 35 47 132 186 216 177 96 223 189 91 234 226 111 234 213
117 217 205 188 218 260 103 218 206 104 207 175 76 177 131 54 142 80 41 106 85 22 183
58 72 80 53 16 78 50 17 9 10 2 54 76 74 108 111 152 218 194 106 228 209 102 228 200
110 212 186 79 225 142 66 196 156 82 88 138 54 155 106 27 32 82 33 55 51 14 87 48
15 31 46 14 10 15 9 11 9 0 64 90 91 54 90 95 230 185 87 212 192 105 214 177 66 288
185 71 192 180 64 178 127 42 170 117 48 139 89 30 102 53 12 84 43 13 79 48 15 72 42
14 10 13 4 12 8 0 89 164 110 58 96 129 130 126 115 186 134 82 198 148 88 183 136 79
174 125 59 169 120 54 146 97 41 118 67 24 80 52 16 75 46 18 58 42 16 13 7 9 10 5
0 13 11 3 66 111 114 70 109 102 78 163 96 57 71 82 192 111 58 145 98 21 152 102 51
130 86 31 110 63 21 83 44 11 89 42 12 28 8 0 7 5 10 16 4 0 17 10 2 30 30 10
58 88 86 53 86 94 98 91 102 85 98 110 54 80 78 23 68 85 29 34 25 63 41 25 21 2
0 6 0 9 17 10 4 11 0 0 34 21 13 47 35 23 38 26 14 87 15 39
```

Sayısal resim

Siyah-Beyaz Görüntü

Binary görüntü olarak da bilinir. 2 boyutlu bir fonksiyon ile gösterilir.

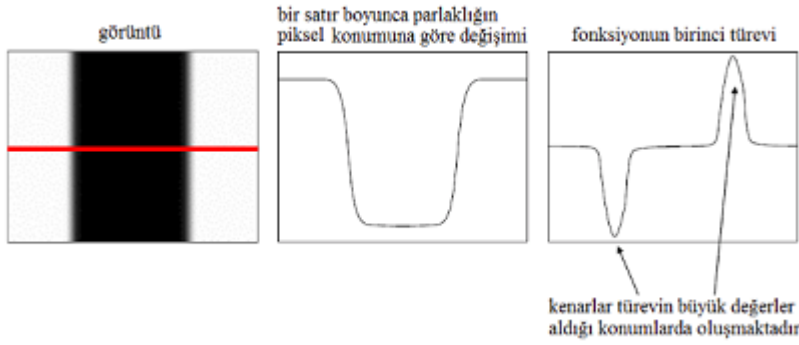
- $f(x,y)$
 - x: Satır (i)
 - y: Sütun (j)

Derinlik değeri (renk boyutu) 1'dir

Lineer Filtreleme ve Kenar Belirleme

Kenar Belirleme (Edge Detection)

- Kenar, görüntü içerisinde parlaklığın sıçrama yaptığı bölgedir.
- Belli eşiğin üstündeki ani değişimler (255'ten 0'a değişim 255'tir)
- Türevin yüksek değer aldığı yerler kenarları oluşturur. (*gradient descent*)



Kenar Belirleme Sorunları

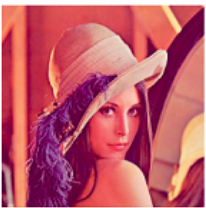
- Gürültü (*noise*)
- Kenar belirleme ve konumlama ölçütleri arasındaki karşılıklı ilişki
- Kenarların çok ölçekli yapısı

Kenar Belirleme Yöntemleri

- Eşik değerini geçmesi koşulunda kenar kabul edilir.
- Gradyan (*gradient*) parlaklık seviyesindeki değişimin en yüksek olduğu yönü belirtir
 - Gradyan, kenar yönüne diktir
- Gradyan genliği (*gradient amplitude*) **kenarın yönü** hakkında bilgi verir
- Gradyan açısı (*gradient angle*) **kenarın kalınlığı** hakkında bilgi verir

Gradyan'a eğim denilebilir.

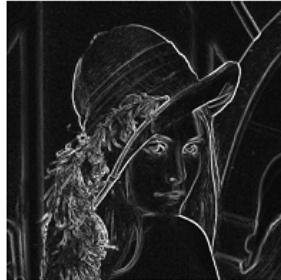
ORJİNAL GÖRÜNTÜ



ROBERTS



PREWITT



LAPLACE



Sobel



Roberts



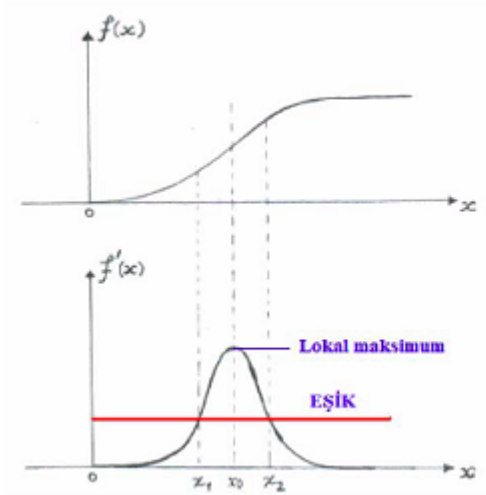
Prewitt



Canny



Gradyan Tabanlı Kenar Belirleme



Prewitt:

-1	0	1
-1	0	1
-1	0	1

1	1	1
0	0	0
-1	-1	-1

Sobel:

-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	-1

Roberts:

0	1
-1	0

1	0
0	-1

x-yönünde maske

y-yönünde maske

Görüntünün birinci türevindeki maks ve min değerlere bakarak kenar belirleme yöntemidir.

Teknik Açıklama

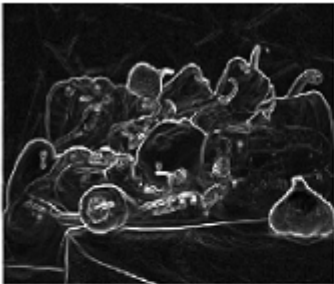
Sobel 2 maske ile, 2 boyutlu eğim (gradyan) ölçümü yapar

Prewitt Sobel'e göre daha basit ama gürültülü sonuçlar elde eder

Robert En basit eğim operatörüdür, köşeden köşeye çapraz geçiş yapar

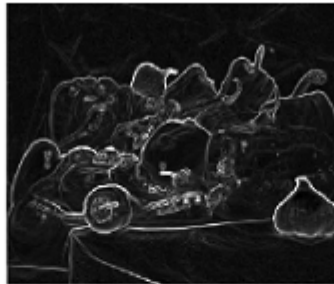
$$G = \sqrt{G_x^2 + G_y^2}$$

Sobel maskesi ile edilen görüntü



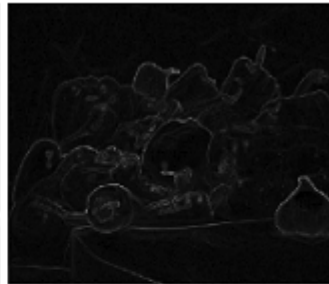
eşik = 77

Prewitt maskesi ile edilen görüntü



eşik = 55

Robert maskesi ile edilen görüntü



eşik = 18



Laplasyan Tabanlı Kenar Belirleme

İkinci türevdeki sıfır geçişleriyle belirleme.

- İkinci türev 1.nin max noktasındayken 0 olur, 0 noktaları tespit edilir
- Marr-Hilderth
- Canny ✨

$$L(x,y) = \nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}$$

Marr-Hilderth Kenar Belirleme

- LoG (*Laplacian of Guassion*)'un 0 Kesişimini ele alır
- Ön işlem olarak yumuşatma (*gauss filter*) kullanır

```
>>a=imread('cameraman.tif');
>>l=fspecial('laplacian',1);
>>icz=edge(a,'zerocross',l);
>>imshow(icz)

>>log=fspecial('log',13,2);
>>icz=edge(a,'zerocross',log);
>>imshow(icz)
```

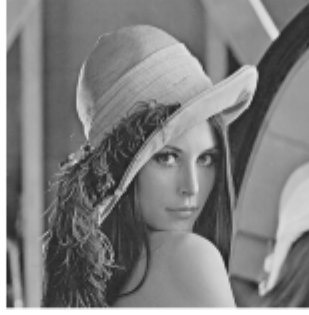


Canny Kenar Belirleme

- Çok fazla kullanılır

$$\begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix}$$

- Gradyan büyüklüğü ve yönü belirlenir
- Birden fazla *pixel* kalınlıktaki kenarlar, inceltme ile bir *pixel* kalınlığa düşürülür
 - İnceltme, q bir kenarsa, komşularından daha büyük değer almalıdır
- Büyük ve küçük olmak üzere iki eşik değeri (*threshold*) tanımlanır
 - Eşik değeri yüksek seçilirse kalın kenarlar, düşük seçilirse ince kenarlar ve gürültü tespit edilir
 - Büyük olan ile kalın kenar eğrileri belirlenir
 - Küçük olan ile eğriler devam ettirilir
 - Komşularının gradyan açıları yakın değerler alıyorsa kenara dahil edilir



Gradyan genliđi



eřikleme



inceltme



yüksək eřik
(kuvvetli kenarlar)



düşük eřik
(zayıf kenarlar)



histerisis (çift) eřikleme

Gürültü (Noise)

Tür	Açıklama
Tuz ve biber (<i>salt & pepper</i>)	Rastgele siyah ve beyaz piksellerin oluşması
İmpuls (<i>impulse</i>)	Rastgele beyaz piksellerin oluşması
Gauss	Parlaklık seviyelerinde gauss dağılımına uyan değişimlerin oluşması



Orijinal



Tuz ve biber gürültüsü



İmpuls gürültüsü



Gauss gürültüsü

Gürültülü Engelleme

Gauss fonksiyonu ile çarpılarak gürültü söndürülebilir.

Frekans Kavramı

Mesafeye göre gri seviye değişiminin miktarını ifade eder.

- 0'dan 255 değişimi veya tam tersi yüksek frekans
- 200'den 220 değişimi veya tam tersi düşük frekans

Lineer Filtreler

Filtreler **frekans**'a göre *pixel*'leri temizlemek için kullanılır.

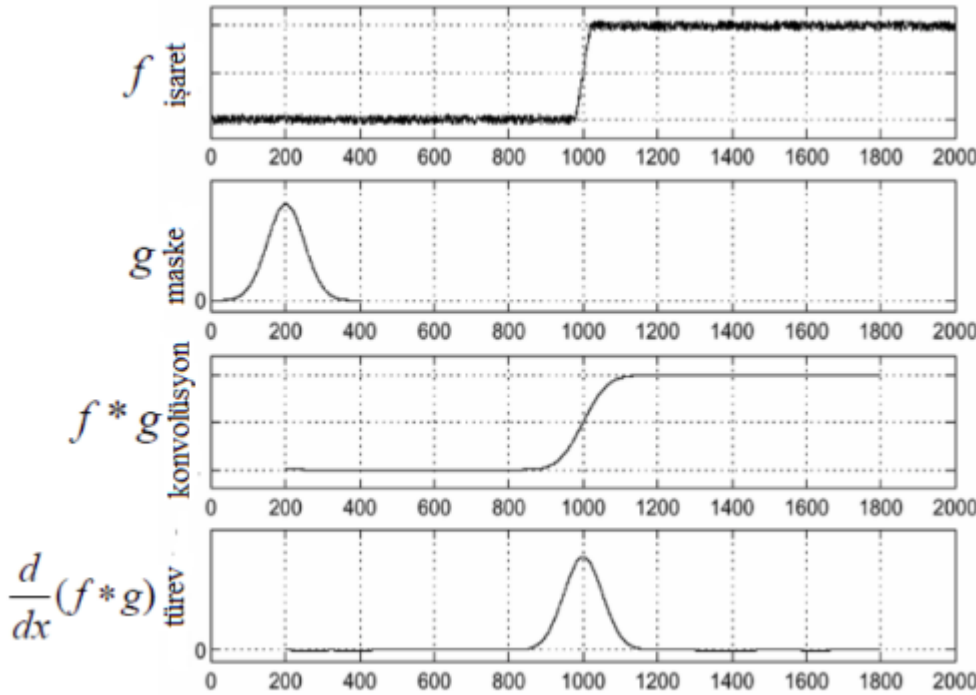
Alçak Geçirgen Filtreler

- Gürültüyü yok eder (*noise cleaning*)
- Görüntüyü yumuşatır (*smoothing*)
- Kenarları bulanıklaştırır (*blurring*)

Filtre	Açıklama
Guassian	Sert ton değişiklerini azaltır ve görüntünün daha yumuşak olmasını sağlar. Maskenin artması bulanıklığı ve kenar kalınlığını artırır
Laplasyan	Sayısal olarak en yakın iki <i>pixel</i> 'in x ve y düzlemine göre türevidir. Gürültüye çok duyarlıdır
LoG	Önce huassian

Yüksek Geçirgen Filtreler

Görüntü içerisindeki detayları, kenarları ve gürültüyü ortaya çıkarır.



Guassian Filtre

- Alçak geçirgen filtredir
- Sert ton değişiklerini azaltır
- Görüntünün yumuşak olmasını sağlar
- Maskenin artması bulanıklığı ve kenar kalınlığını artırır

Laplasyan Fitre

- En yakın iki *pixel*'in x ve y düzlemine göre türevini hesaplar
- Gürültüye karşı çok duyarlıdır

LoG (Laplasyan of Guassian)

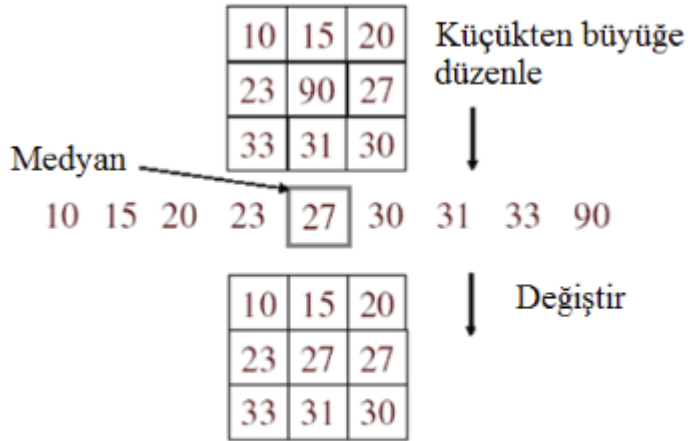
2 filtreleme tekniğın sıralı olarak birleştirilmiř halidir

- Laplansyan gürültüye çok duyarlıdır
- Gürültü, **guassian filtre** ile azaltılır ve görüntü yumuřatılır
- Sonra laplasyan filtre uygulanır

Medyan Filtre

Gaussian Filtre'si gürültüyü giderirken görüntüyü bulanıklařtırır. Medyan filtre:

- Görüntüyü bulanıklařtırmadan gürültüyü engeller
- *Pixel* deęerinin komřu *pixel* deęerlerine göre medyanı alınır



Medyan Filtre



Bozulmuş görüntü



3x3 medyan filtresi uygulanmış

Medyan Filtre



7x7 medyan filtresi uygulanmış



Üç kere 3x3 medyan filtresi uygulanmış

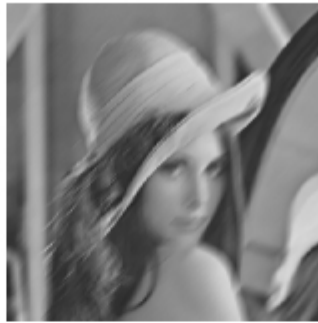
Temel Görüntü İşlemleri

İşlem	Yapılma Yöntemi
Bulanıklaştırma (<i>blur</i>)	<i>P</i> pixel değerlerin çevresindeki <i>pixel</i> değerleri ile ortalamasının hesaplanması
Keskinleştirme (<i>sharpen</i>)	Orjinal görüntüye kenarları bulunmuş görüntü eklenir (Maskedeki merkez değeri 1 artırılarak)
Kabartma	Resme 3D efekti verir, merkezin bir tarafındaki <i>pixel</i> değerlerinden diğer taraftakilerin çıkarılması ile yapılır. Negatif olanlar gölge, pozitif olanlar aydınlık yüzey olur. Görüntünün çoğu gri tonlarına dönüşecektir

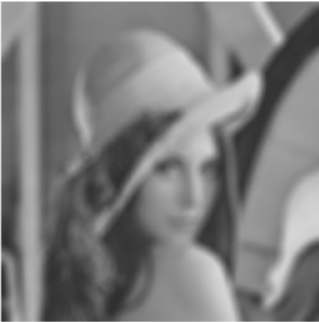
Original Image



Motion Blurred Image



Blurred Image



Sharpened Image



Kabartma



Renk ve Geometrik Dönüşümler

Renk Formatları

Her bir renk için 8bit'lik bir tanımlama var. (255)

Format	Açıklama	Kullanım Alanı
RGB	Işığa eş değer, genel kullanılan method	TV, PC vs.
CMYK	Boya renklerini taklit eder, baskılarda kullanılır	Printer
HSI		
YIQ		

RGB

- Cihaza ve donanıma bağlı bir renk formatıdır
- RGB ile kodlanan dosyalar az yer kaplar
- RGB: Red Green Blue
- CMYK: Cyan, Magenta, Yellow, Key (Key siyah rengi temsil eder)
 - Key (siyah) renk, baskıda kullanılmazsa, teorideki karşılığını sağlamaz
- RGB beyaza odaklı, CMYK siyaha odaklı hareket eder
 - max RGB: Beyaz
 - max CMYK: Siyah
 - $CMY = 1 - RGB$

Perspektif İzdüşüm

3D resmi 2D'ye geçirince derinlik verisinin kaybolma sebebi, benzerlerlik teoreminden kaynaklanır.

Mutlak siyah varsa boşluk gibi görünür.

Gemometrik Dönüşümler

- Öteleme
- Ölçekleme
- Döndürme

Her birinde homojen koordinatlar kullanılır.

Homojen Koordinatlar

Fazlalık olan kısımlara 1, diğer alanlara değişkenler verilir. [xy1] vs.

Matrikslerde çarpım işlemleri daha kolaydır.

Görüntü İyileştirme Metodları

Çok koyu ya da çok açık görüntüler üzerinde uygulanır.

Metod	Açıklama
Histogram Germe	Verilerin aralığını arttırma işlemi
Histogram Eşitleme	Her renk değeri için eşit sayıda pixel olmasını sağlama

Histogram Germe

Pixel değerlerinin aralığını genişletme işlemi olarak da bilinir.

- Resmin sahip olduğu en düşük ve en yüksek pixel değeri bulunur
 - $eski_{max}, eski_{min}$
- İstenen en yüksek ve en düşük pixel aralıkları belirlenir
 - Genelde 0, 255 değerleri seçilir
 - $yeni_{max}, yeni_{min}$
- Her bir pixel, yeni başlangıç ve bitiş noktasına göre değerler alır

$$yeni_i = ((yeni_{max} - yeni_{min}) / (eski_{max} - eski_{min})) \cdot (eski_i - eski_{min}) + yeni_{min}$$

Histogram Eşitleme

Her bir parlaklık seviyesi için aynı sayıda pixel bulunmasını sağlayarak resmin pixellerinin dengeli (uniform) dağılımda olması amaçlanır.

- Her pixel ton değerinin resmin içinde hangi oranda olduğu $p_r(r_k)$ hesaplanır
 - $P_r(r_k) = n_k / n$
 - n : Toplam pixel sayısı
 - n_k : k. pixel sayısı
- Kümülatif olasılık fonksiyonu s_k hesaplanır
 - $s_k = T(r_k) = \sum_{j=0}^k P_r(r_k) = \sum_{j=0}^k n_j / n$
- Ters dönüşüm yapılarak, hangi renk tonu yerine hangisinin geleceği hesaplanır
 - $r_k = T^{-}(s_k) = L * T(r_k)$
 - L : Maksimum pixel değeri (255)

Pythonda Histogram Germe İşlemi

```
def histogram_stretching(image: Image, new=(0, 255)):
    """Histogram Germe

    Arguments:
        image {PIL.Image} -- Resim

    Keyword Arguments:
        new {(min, max)} -- tuple (default: {(0, 255)})

    Returns:
        PIL.Image -- Gerilmiş resim
    """

    def difference(variable: tuple):
        return variable[1] - variable[0]

    np_image = np.array(image) # Resmi numpy.ndarray formatına çevirme
    flatten_img_np = np_image.reshape(-1) # Resmi tek boyuta indirgeme

    # Histogram germe denklemi
    old = flatten_img_np.min(), flatten_img_np.max()
    for i in range(0, len(flatten_img_np)):
        flatten_img_np[i] = (difference(new) / difference(old)) * \
            (flatten_img_np[i] - old[0]) + new[0]

    # Aynı boyutlardaki yeni resmi oluşturma
    return Image.fromarray(flatten_img_np.reshape(np_image.shape))
```

Python'da Histogram Eşitleme

```
def histogram_equalization(image: Image):  
    """Histogram eşitleme  
  
    Arguments:  
        image {PIL.Image} -- Resim  
  
    Returns:  
        PIL.Image -- Resim  
    """  
  
    np_image = np.copy(image) # Numpy formatına çevirme  
    flatten_image = np_image.flatten() # Resmi tek boyuta indirgeme  
  
    # Pixel bilgilerini alma  
    pixel_num = len(flatten_image)  
    max_pixel_num = flatten_image.max()  
    min_pixel_num = flatten_image.min()  
  
    # Pixel dağılımını hesaplama  
    pixel_manager = {} # Pixel yönlendirici  
    cumulative_probability = 0 # Kümülatif pixel bulunma olasılığı  
    for i in range(min_pixel_num, max_pixel_num + 1):  
        pixel_count = 0 # Pixel'in tekrar etme sayısı  
        for pixel in flatten_image:  
            if i == pixel:  
                pixel_count += 1  
        cumulative_probability += pixel_count / pixel_num  
        pixel_manager[f'{i}'] = round(  
            max_pixel_num * cumulative_probability  
        )  
  
    for i in range(len(flatten_image)):  
        flatten_image[i] = pixel_manager[f'{flatten_image[i]}']  
  
    return Image.fromarray(flatten_image.reshape(np_image.shape))
```

Ek kaynak için [buraya](#) bakabilirsin.

Harici Bağlantılar

- [Python ile Görüntü İşleme: Histogram, Normalleştirilmiş Histogram ve Histogram Eşitleme](#)
- [Edge Detection](#)