BSM409 Görüntü İşleme

Bölüm 11
Görüntü Bölütleme

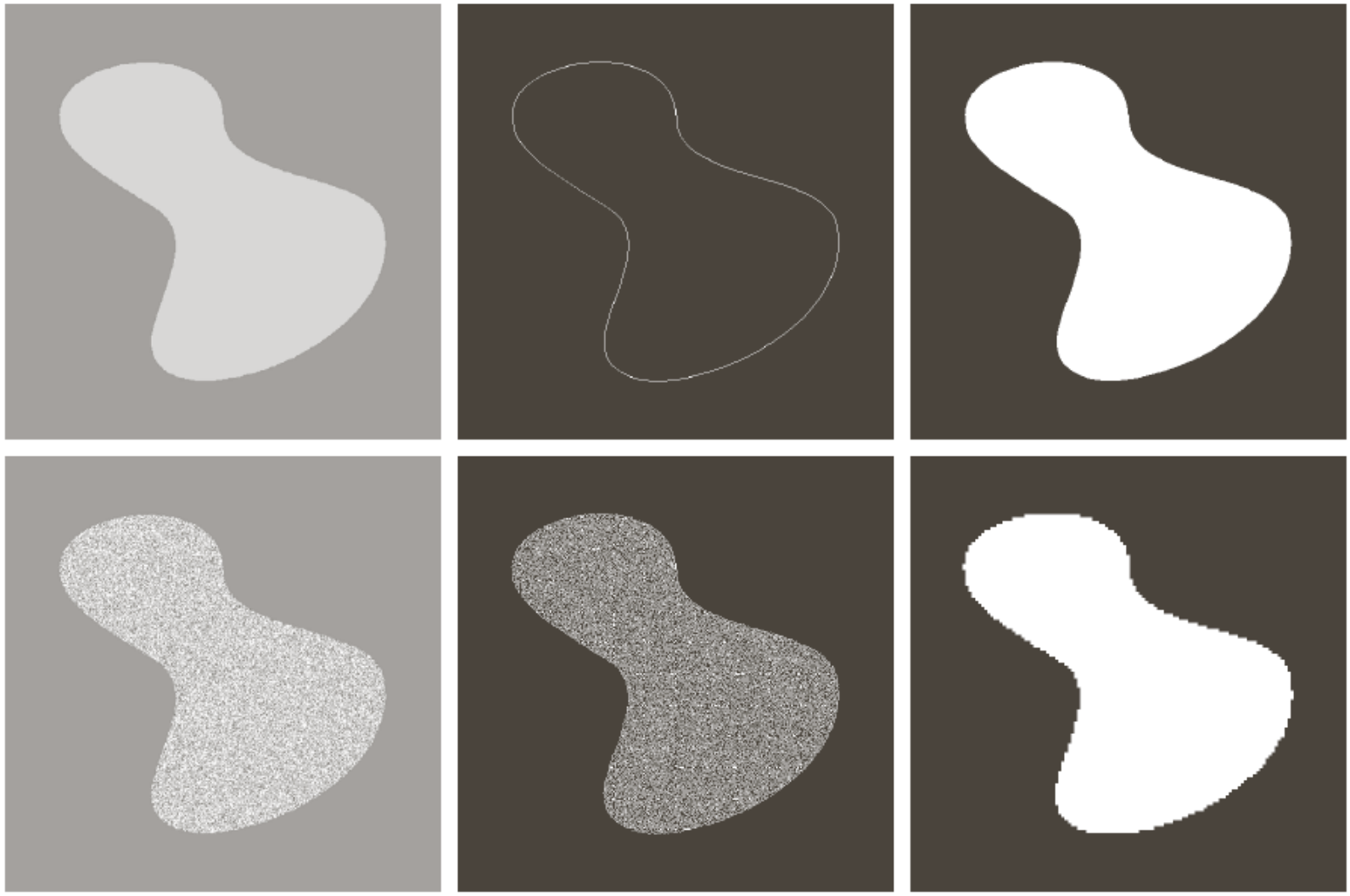
Dr. Öğr. Üyesi Caner ÖZCAN

Bütün, kendi parçalarının toplamına eşittir. ~Euclid
Bütün, kendi parçalarının toplamından daha büyüktür.~Max Wertheimer

İçerik

10. Görüntü Bölütleme

- Temeller
- Nokta, Çizgi ve Kenar Tespiti
- Thresholding
- *Region-Based Segmentation*
- *Segmentation Using Morphological Watersheds*
- *The Use of Motion in Segmentation*



a b c
d e f

ŞEKİL 10.1 (a) Sabit yeğinlikli bölge ihtiva eden görüntü. (b) Yeğinlik süreksizliğinden elde edilen, iç bölge sınırlarını gösteren görüntü. (c) Görüntüyü iki bölgeye bölütleme sonucu. (d) Dokusal bölge ihtiva eden görüntü. (e) Ayrıt hesaplama sonucu. Sınıra ekli çok sayıdaki küçük ayrıtın olması, sadece ayrıt bilgisini kullanak belirli bir sınırı bulmayı zorlaştıracağına dikkat ediniz. (f) Bölgesel özelliklere dayalı bölütleme sonucu.

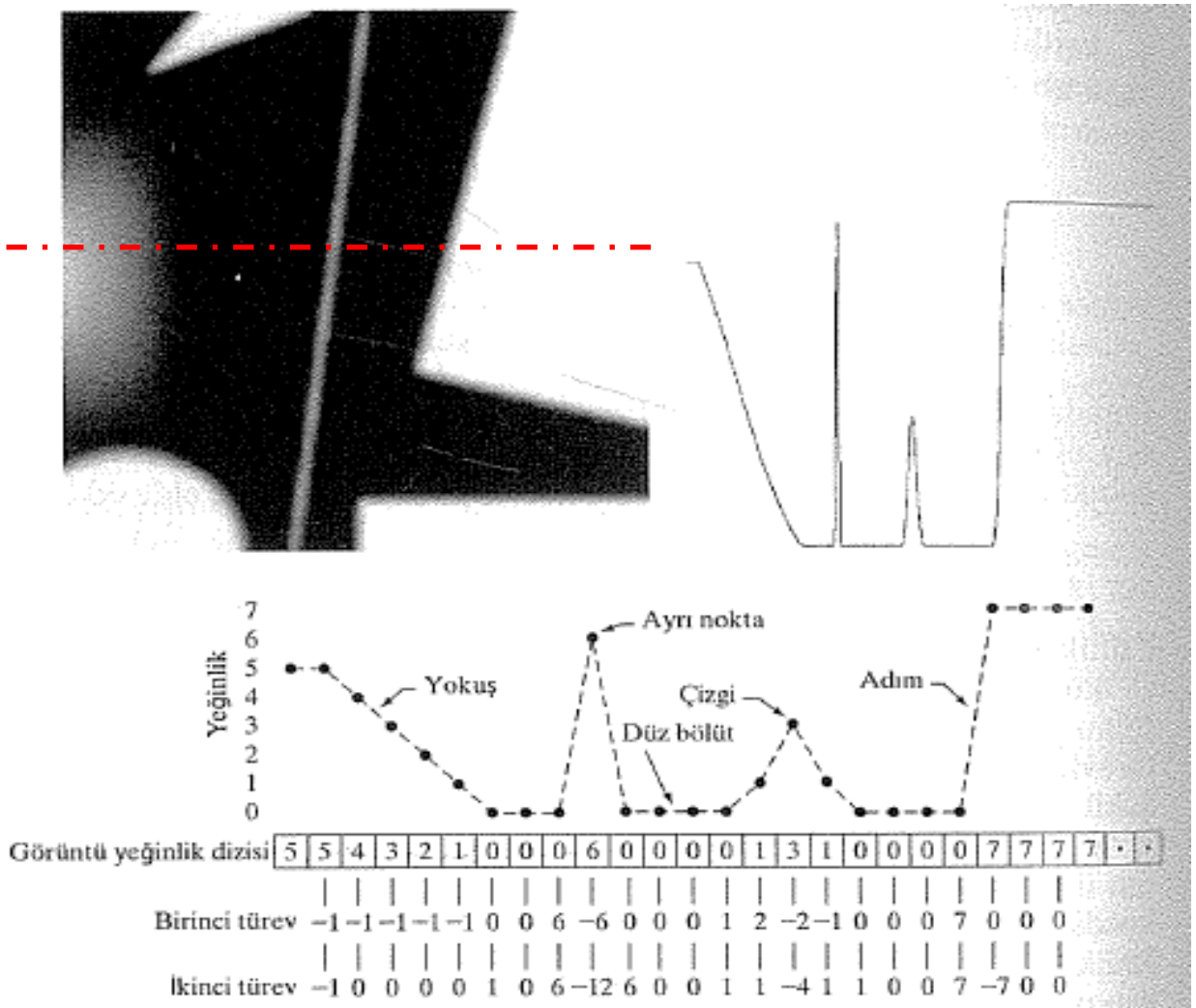
Geçmiş

- Birinci mertebeden türev

$$\frac{\partial f}{\partial x} = f'(x) = f(x+1) - f(x)$$

- İkinci mertebeden türev

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$$



a b
a b
c

FIGURE 10.2 (a) Image. (b) Horizontal intensity profile through the center of the image, including the isolated noise point. (c) Simplified profile (the points are joined by dashes for clarity). The image strip corresponds to the intensity profile, and the numbers in the boxes are the intensity values of the dots shown in the profile. The derivatives were obtained using Eqs. (10.2-1) and (10.2-2).

Yalıtılmış Noktaların Tespiti

- Laplas

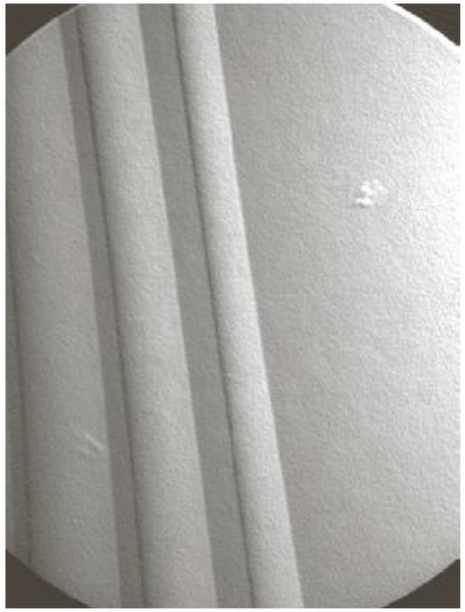
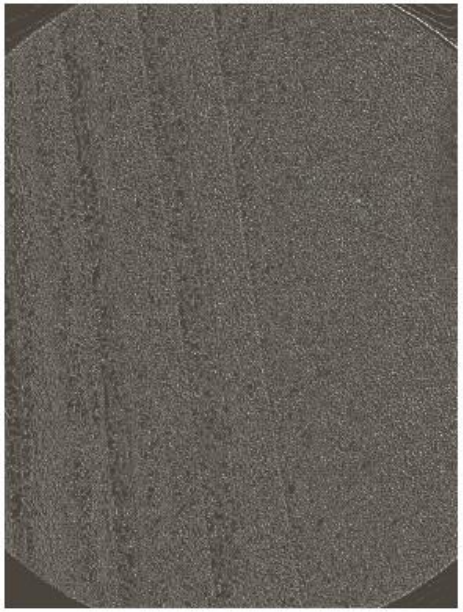
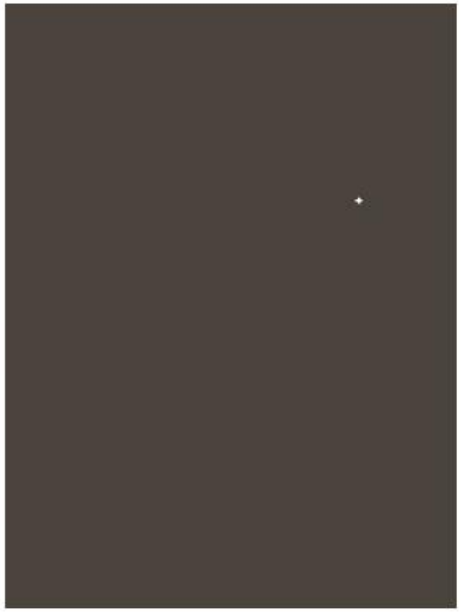
$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$$= f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)$$

$$-4f(x,y)$$

$$g(x,y) = \begin{cases} 1 & \text{if } |R(x,y)| \geq T \\ 0 & \text{diğer} \end{cases} \qquad R = \sum_{k=1}^{9} w_k z_k$$

1	1	1
1	−8	1
1	1	1

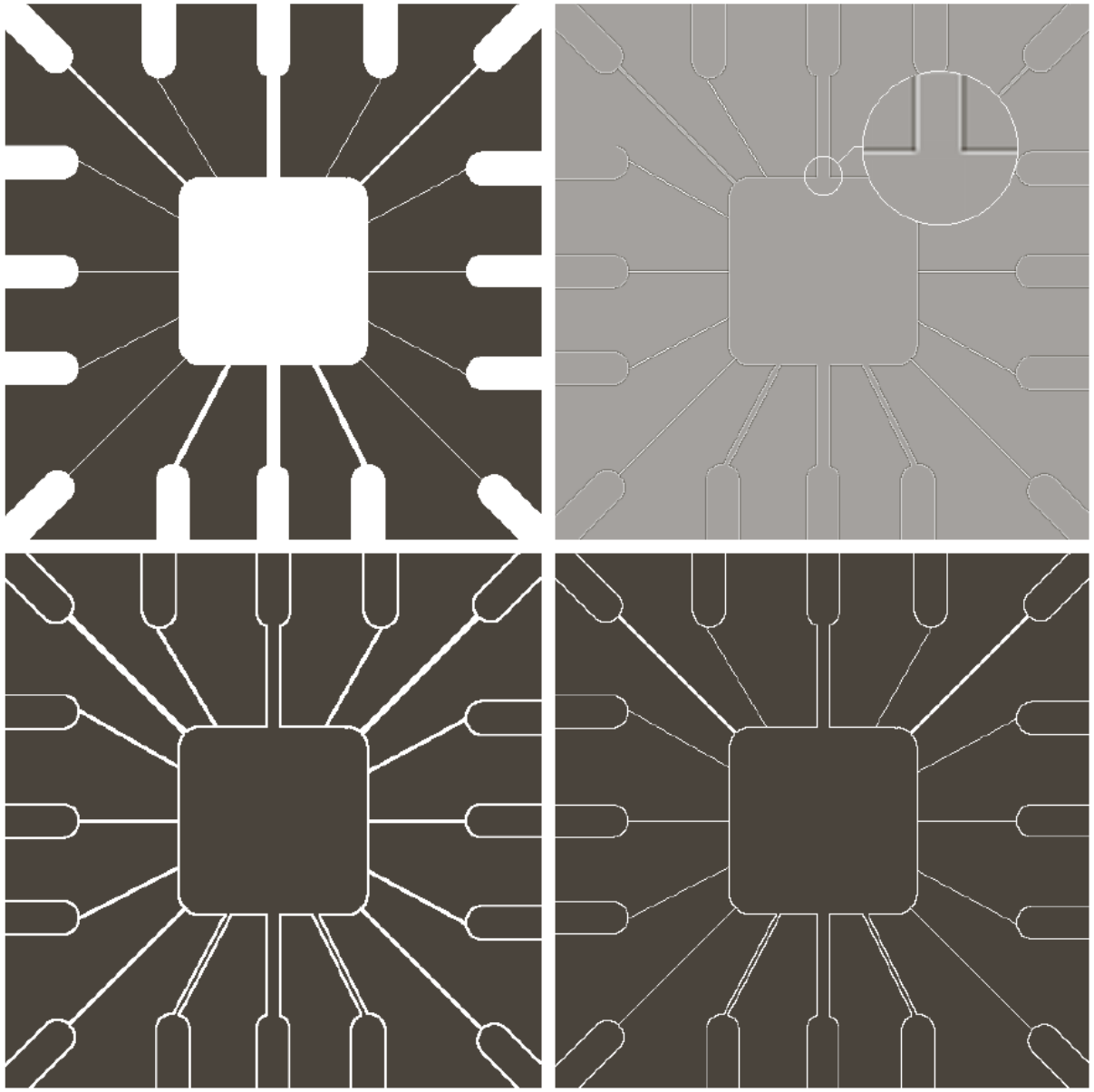




a
b c d
ŞEKİL 10.4
(a) Nokta saptama (Laplas biçimli) maskesi. (b) Bir gözenekli türbin bıçağı X-ışını görüntüsü. Gözenek bir tek siyah pikselden oluşur. (c) Görüntü ile maskenin katlanması sonucu. (d) Bir tek nokta gösteren Eşitlik (10.2-8) 'in kullanılması sonucu (kolaylıkla görülebilmesi için nokta genişletilmiştir). (Orijinal görüntü X-TEK System, Ltd. izniyle alınmıştır)

Çizgi Tespiti

- İkinci türevler daha güçlü tepkiler verir ve birinci türevlerden daha ince çizgiler üretmeyi sağlar.

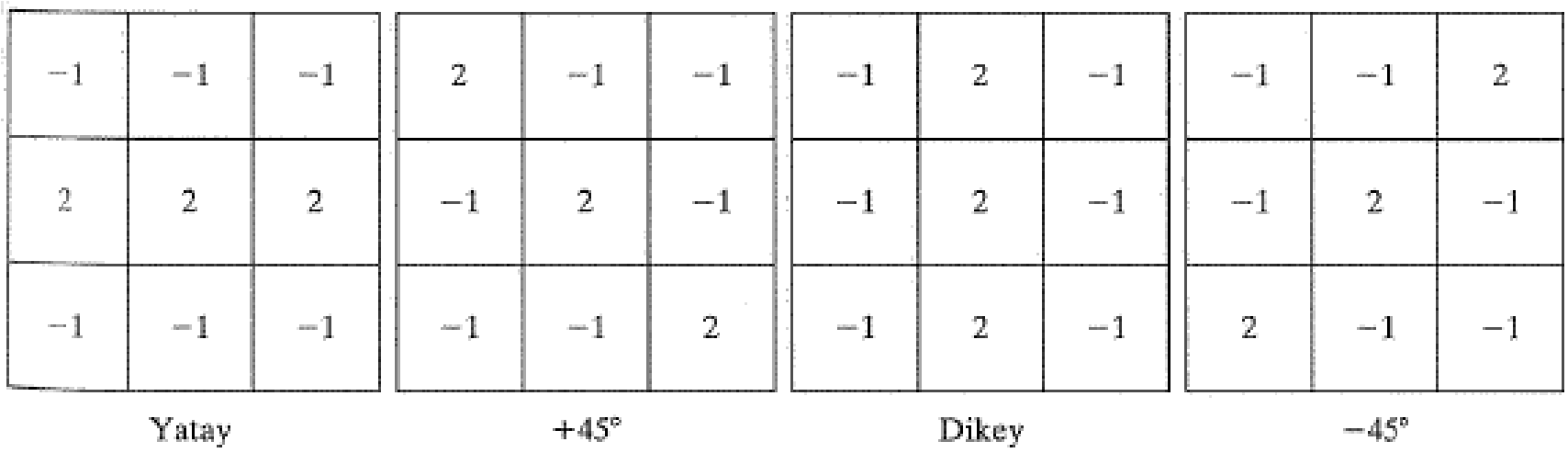
- İkinci türevin çift çizgi efekti düzgün bir şekilde ele alınmalıdır.



a b
c d

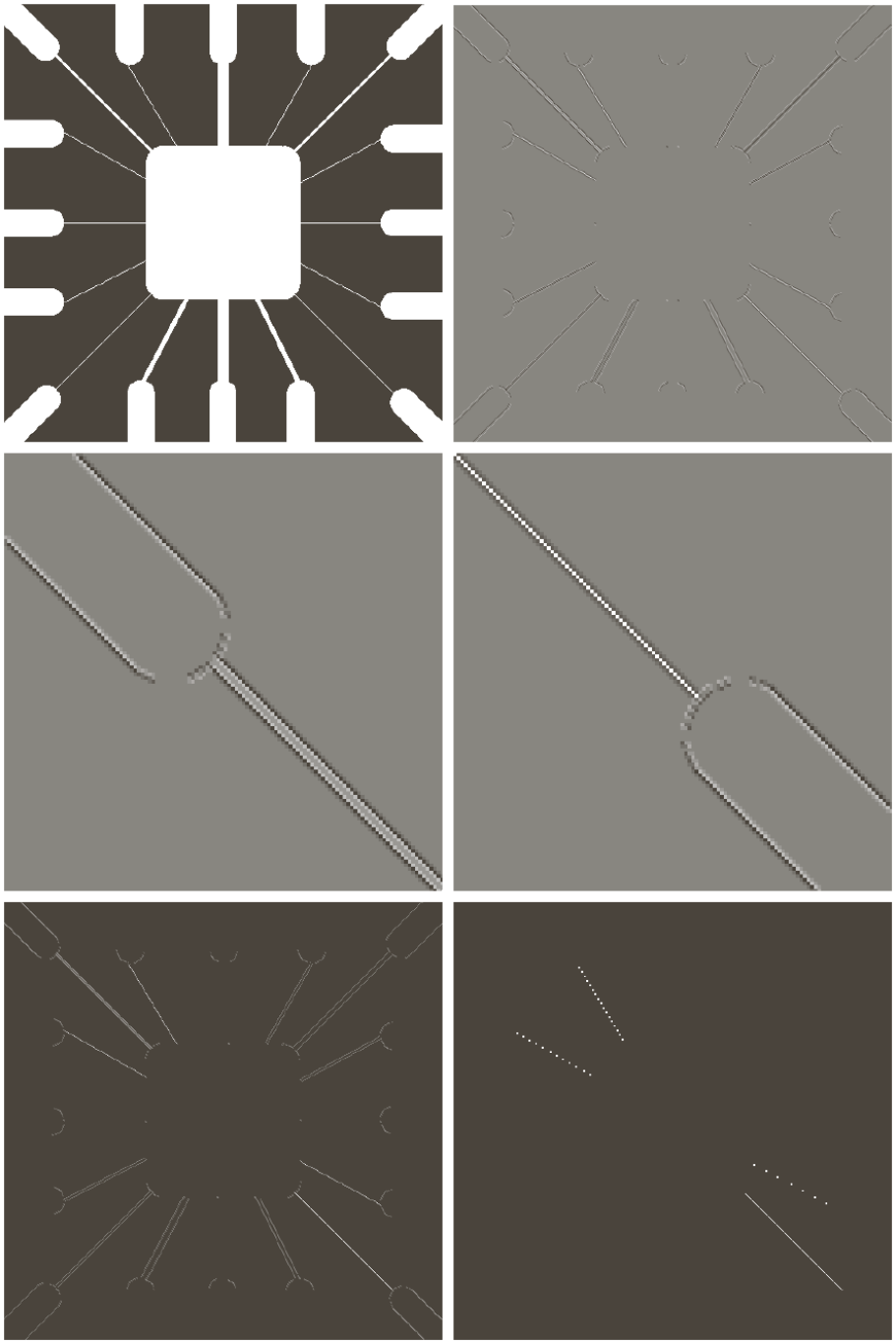
ŞEKİL 10.5
(a) Orijinal görüntü. (b) Laplas görüntüsü; büyütülmüş kısım Laplasın pozitif/ negatif çift-çizgi etkisi özelliğini göstermektedir. (c) Laplasın mutlak değeri. (d) Laplasın pozitif değerleri.

Belirli Yönlerde Çizgi Tespiti



−1	−1	−1	2	−1	−1	−1	2	−1	−1	−1	2
2	2	2	−1	2	−1	−1	2	−1	−1	2	−1
−1	−1	−1	−1	−1	2	−1	2	−1	2	−1	−1
Yatay			+45°			Dikey			−45°		

ŞEKİL 10.6 Çizgi saptama maskeleri. Açılar Şekil 2.18(b)'deki eksen sistemine uygundurlar.

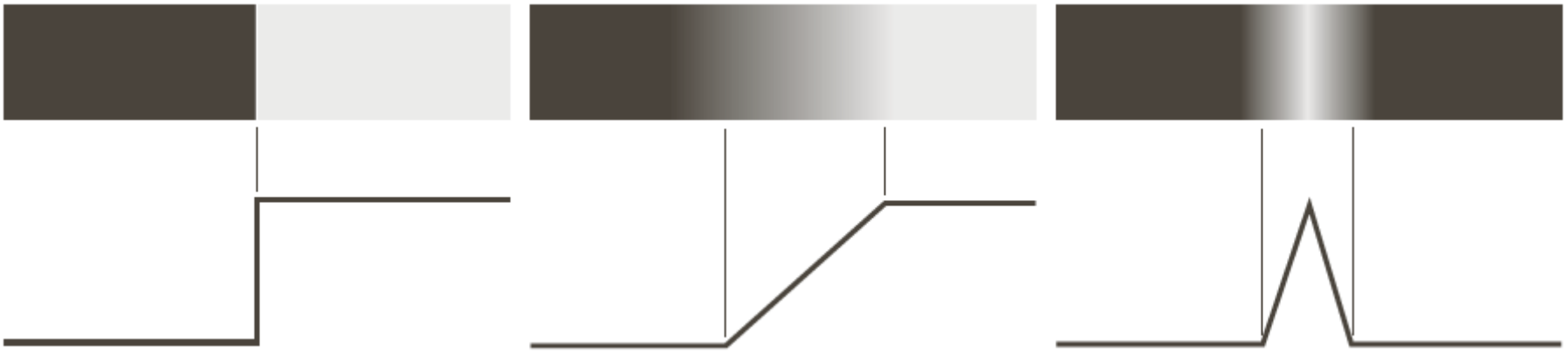


a b
c d
e f

ŞEKİL 10.7
(a) Bir telli-bağlantı şablonu görüntüsü. (b) Şekil 10.6 daki +45° çizgi algılayıcı maskesi ile işlem sonucu. (c) (b)'nin sol üst bölgesinin büyütülmüş görüntüsü. (d) (b)'nin sağ alt bölgesinin büyütülmüş görüntüsü. (e) Bütün negatif değerleri sıfır yapılmış (b)'deki görüntü. (f) (e)'deki g görüntüsü için $g \geq T$ koşulunu sağlayan bütün noktalar (beyaz olanlar). ((f)'deki noktalar anlaşılırlığı artırmak için büyütülmüşlerdir.)

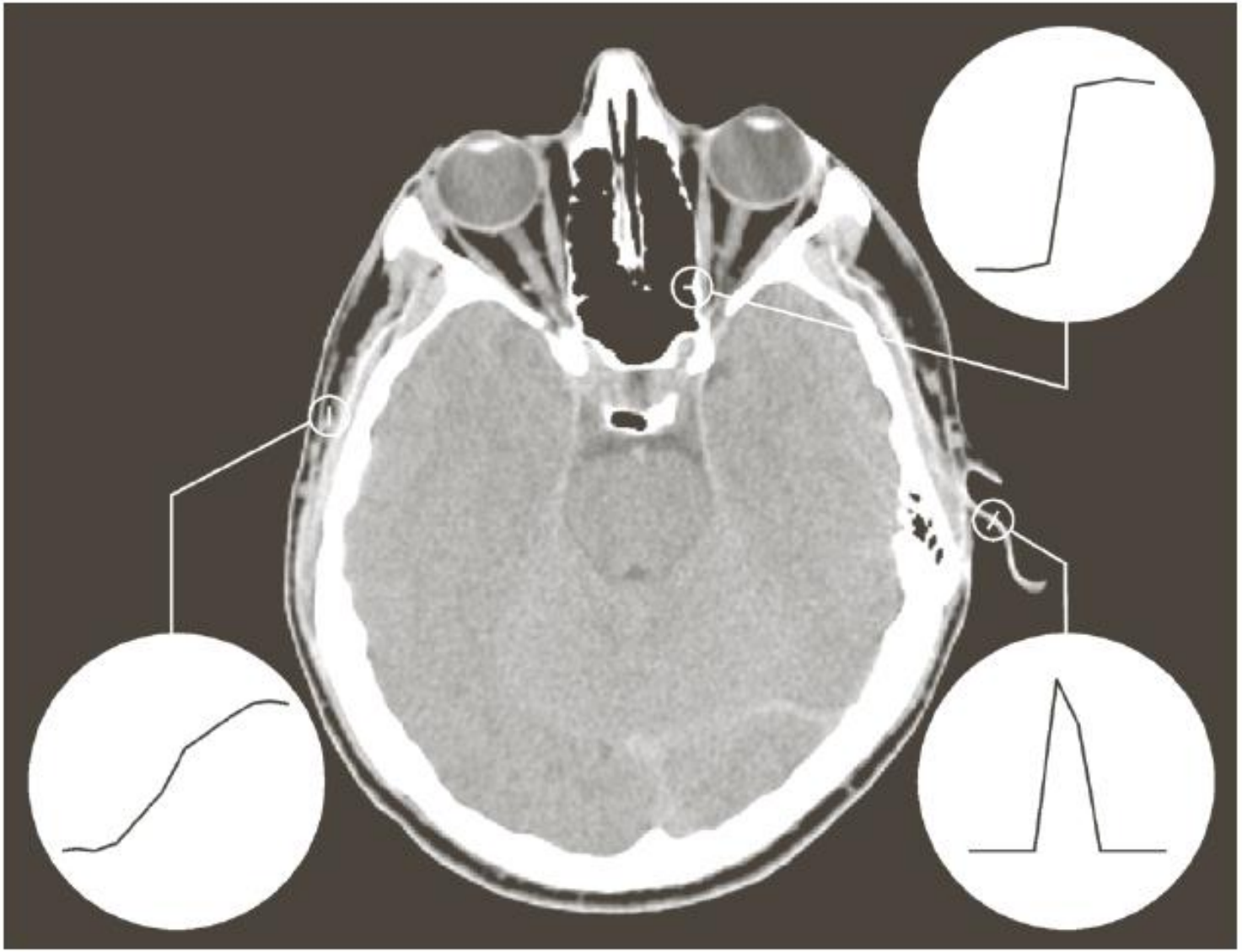
Kenar Tespiti

- Kenarlar, parlaklık işlevinin aniden değiştiği piksellerdir.
- Kenar modelleri

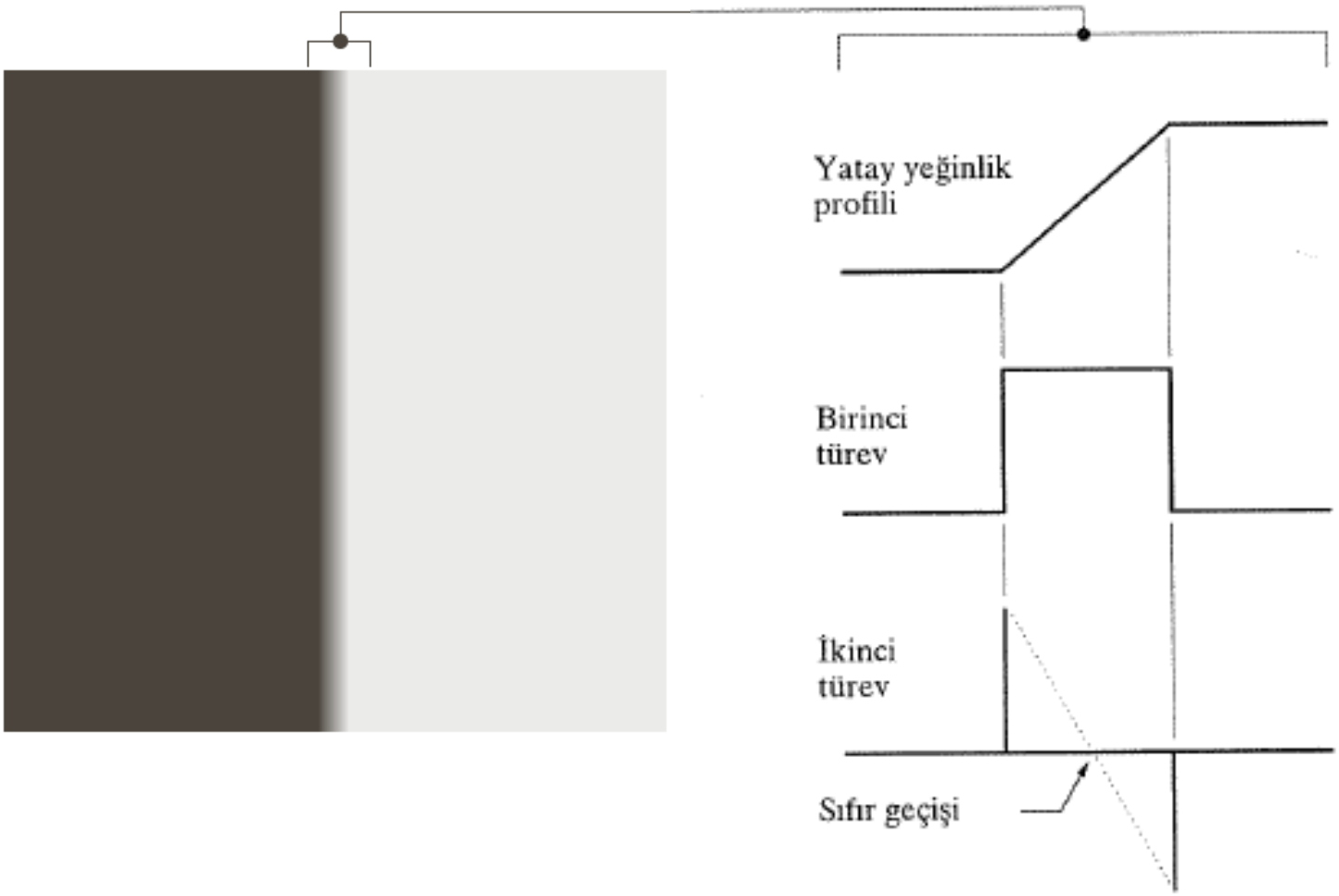


a b c

ŞEKİL 10.8
Soldan sağa, bir adım, bir yokuş, ve bir çatı ayrıtı modelleri (ideal gösterimler) ve bunların uygun yeğinlik profilleri.

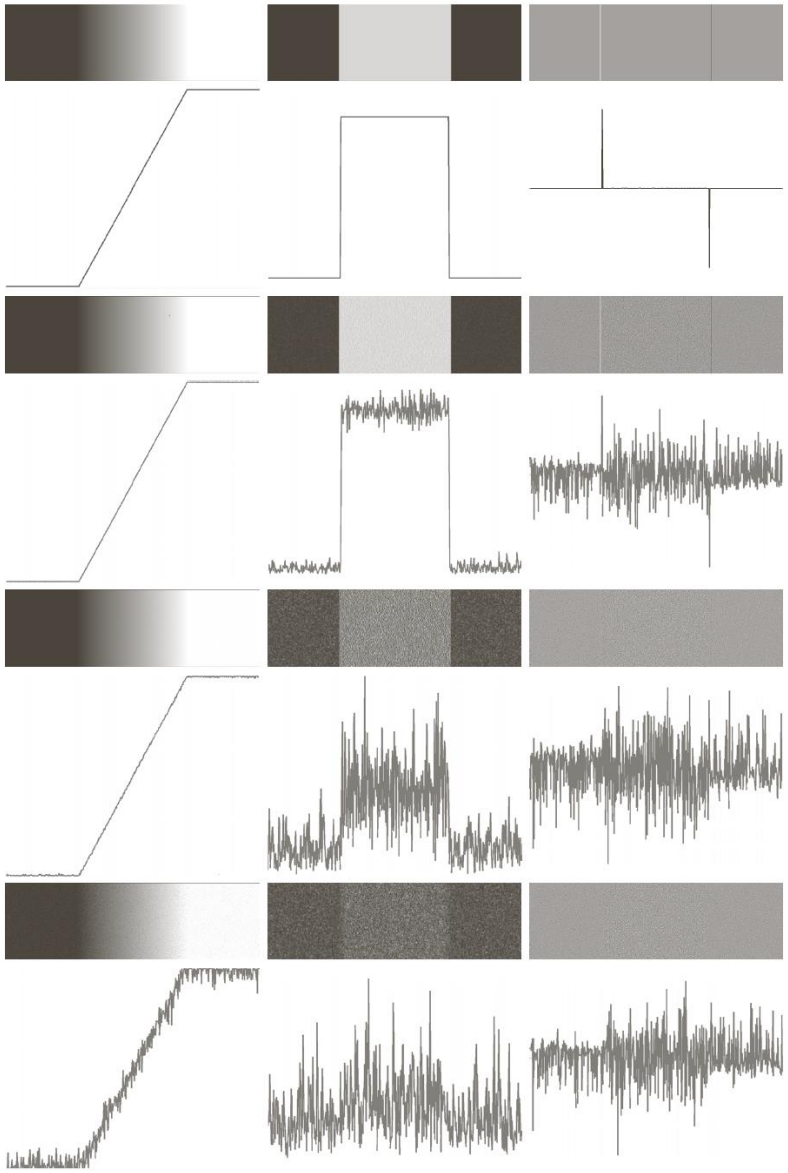


ŞEKİL 10.9 Güncel yokuş (yukarı, sol), adım (üst, sağ), ve çatı ayrıtı profillerini gösteren (büyütülmüş) 1508 × 1970 boyutlarında bir görüntü. Profiller, küçük dairelerde kısa çizgi bölütleri ile belirtilen alanlarda koyudan parlak renge doğru verilmiştir. Yokuş ve "adım" profilleri açıklığı (erimi) sırasıyla 9 piksel ve 2 pikseldir. Çatı ayrıtının tabanı 3 pikseldir. (Orijinal görüntü Dr. David R. Pickens, Vanderbilt University'nin izniyle alınmıştır.)



a b

ŞEKİL 10.10
(a) İdeal bir dikey yokuş ayrıtı ile ayrılmış sabit yeğinliğe sahip iki bölge. (b) Birinci ve ikinci türevleri ile birlikte bir yatay yeğinlik profilini gösteren ayrıt yakınındaki detaylar.



ŞEKİL 10.11 Birinci sütun: Sıfır ortalamalı ve standart sapmaları sırasıyla 0.0, 0.1, ve 10.0 yeğinlik seviyelerine sahip rasgele Gauss gürültüsü ile bozulmuş bir yokuş ayrıtının görüntü ve yeğinlik profilleri. İkinci sütun: Birinci-türev ve yeğinlik profilleri. Üçüncü sütun: İkinci-türev ve yeğinlik profilleri.

Birinci Mertebeden Türev Kullanarak Temel Kenar Tespiti

$$\nabla f \equiv grad(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

The magnitude of ∇f

$$M(x, y) = \text{mag}(\nabla f) = \sqrt{g_x{}^2 + g_y{}^2}$$

The direction of ∇f

$$\alpha(x, y) = \tan^{-1}\left[\frac{g_x}{g_y}\right]$$

The direction of the edge

$$\phi = \alpha - 90^\circ$$

Birinci Mertebeden Türev Kullanarak Temel Kenar Tespiti

$$\text{Edge normal: } \nabla f \equiv grad(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

$$\text{Edge unit normal: } \quad \nabla f \,/\, \text{mag}(\nabla f)$$

In practice,sometimes the magnitude is approximated by

$$\text{mag}(\nabla f) = \left|\frac{\partial f}{\partial x}\right| + \left|\frac{\partial f}{\partial y}\right| \text{ or } \text{mag}(\nabla f) = \max\left(\left|\frac{\partial f}{\partial x}\right|, \left|\frac{\partial f}{\partial y}\right|\right)$$

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	0	0	−1
0	1	1	0

Roberts

−1	−1	−1	−1	0	1
0	0	0	−1	0	1
1	1	1	−1	0	1

Prewitt

−1	−2	−1	−1	0	1
0	0	0	−2	0	2
1	2	1	−1	0	1

Sobel

a
b c
d e
f g

ŞEKİL 10.14
Bir görüntünün 3 × 3 lük bölgesi (z'ler yeğinlik değerleridirler) ve z_5 olarak etiketlenmiş noktadaki gradyanı hesaplamada kullanılacak olan değişik maskeler.

0	1	1	−1	−1	0
−1	0	1	−1	0	1
−1	−1	0	0	1	1

Prewitt

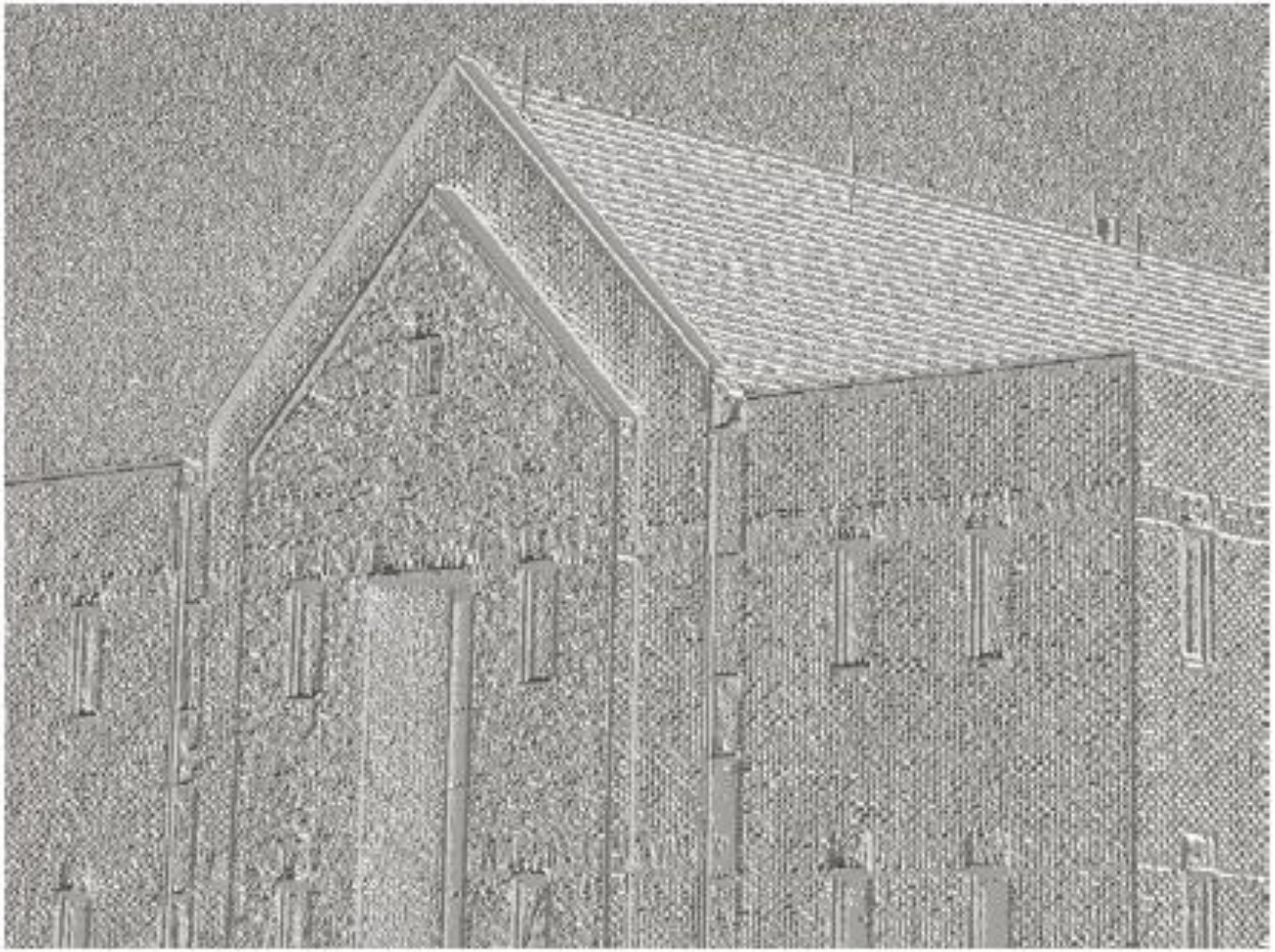
0	1	2	−2	−1	0
−1	0	1	−1	0	1
−2	−1	0	0	1	2

Sobel

a b
c d

ŞEKİL 10.15
Diyagonal ayrıtları saptama için Prewitt ve Sobel maskeleri

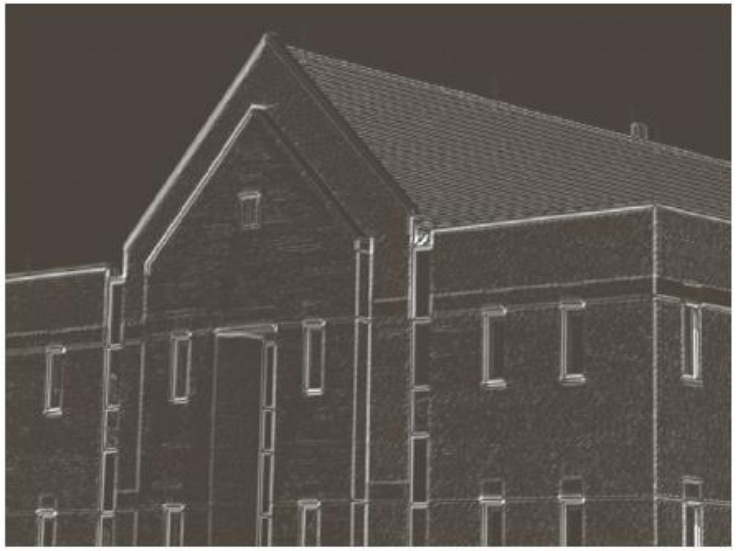
a b
c d
ŞEKİL 10.16
(a) Yeğinlik değerleri [0, 1] aralığında ölçeklenmiş 834 × 1114 piksel boyutlarındaki orijinal görüntü. (b) Görüntüyü süzmek için Şekil 10.14(f)'deki maskeyi kullanarak elde edilen x yönündeki $|g_x|$ gradyan bileşeni. (c) Şekil 10.14(g)'deki maskeyi kullarak elde edilen $|g_y|$ gradyanı. (d) $|g_x| + |g_y|$ gradyan görüntüsü.



ŞEKİL 10.17
Eşitlik (10.2-11)'i kullanarak hesaplanan gradyan açı görüntüsü. Bu görüntüdeki sabit yeğinlik alanları, gradyan vektörü yönünün bu bölgelerdeki bütün piksel konumlarında aynı olduklarını göstermektedir.

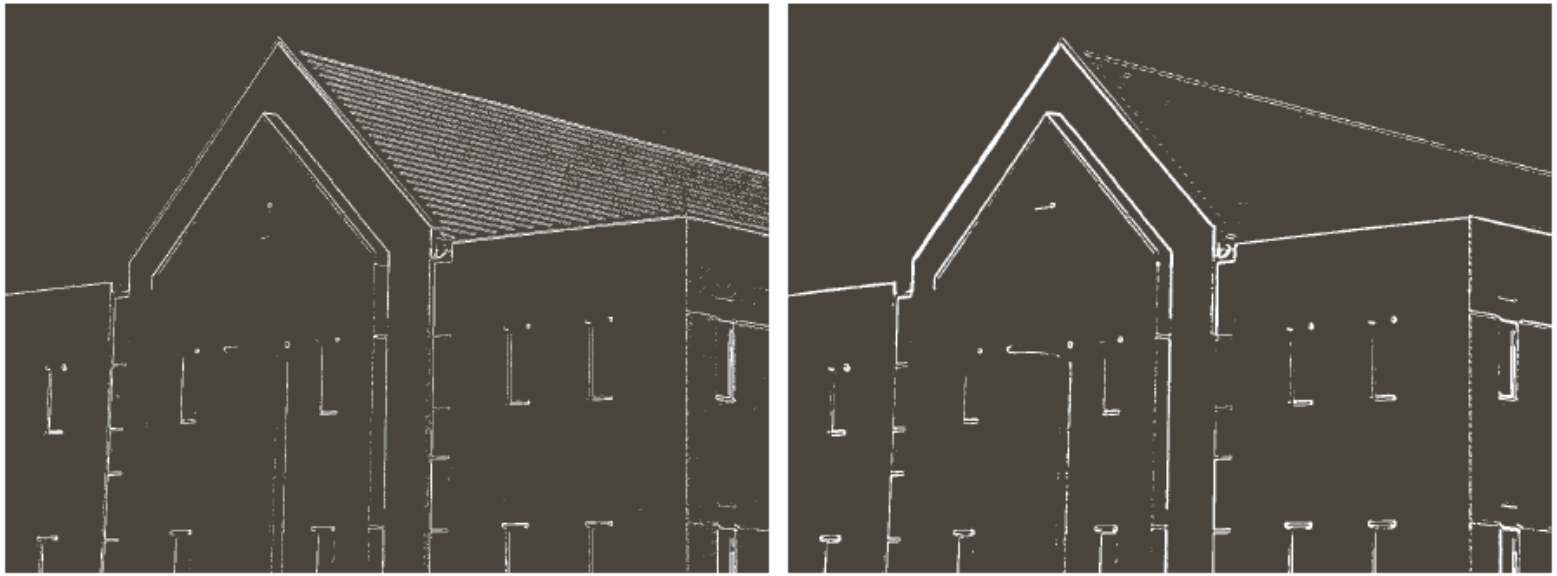
a b
c d

ŞEKİL 10.18
Ayrıt saptama amacıyla 5 × 5 bir ortanca süzgeç kullanarak yumuşatılmış orijinal görüntü ile oluşturulan Şekil 10.16'daki aynı görüntü dizisi.



a b

ŞEKİL 10.19 Diyagonal ayrıt saptama. (a) Şekil 10.15(c)'deki maskeyi kullanarak elde edilen sonuç. (b) Şekil 10.15(d)'deki maskeyi kullanarak elde edilen sonuç. Her iki durumdaki giriş görüntüsü Şekil 10.18(a)'da verilmiştir.



a b

ŞEKİL 10.20 (a) Görüntüdeki en yüksek değerin % 33'ü olarak seçilen eşik ile Şekil 10.16(d)'deki görüntüye eşik uygulanmış durumu; bu görüntü gradyan görüntüsündeki tuğla ayrıntılarının birçoğunu yok edebilecek yeterli seviyededir. (b) Görüntüdeki en yüksek değerin % 33'üne eşit bir eşik seviyesi kullanarak, Şekil 10.18(d)'deki görüntünün eşik uygulanmış hali.

Eşikleme (Thresholding)

- Thresholding görüntü işlemede temel olarak kullanılan yapılardan birisidir.
- Thresholding sayesinde eşik değerleri belirleriz.
- Görüntü piksellerden belirlediğimiz eşik değerlerin altında kalan pikseller sıfıra dönüştürülürken, üstünde kalan alanlar ise maksimum olarak belirlediğimiz piksele çıkarılır.
- Peki bu ne işe yarar?
  - Görüntü üzerinde çalışacağımız kısımları ön plana çıkararak temelini oluşturur.

Eşikleme (Thresholding)

- Bir çok thresholding çeşidi vardır. Bunlar 3 ana başlık altında sınıflandırılmıştır.
  - Simple thresholding
  - Adaptive thresholding
  - Otsu thresholding

!! Thresholding uygulamak için temel kural, gri tonlamalı görüntü olmalıdır.

Simple Thresholding

- Her piksel için aynı eşik değeri uygulanır. Piksel değeri eşik değerinden küçük ise sıfıra ayarlanır, aksi takdirde maksimum olarak belirlenen değere ayarlanır.
- Simple thresholding parametresi için kullanabileceğimiz 5 farklı flag vardır.
  - cv2.THRESH_BINARY
  - cv2.THRESH_BINARY_INV
  - cv2.THRESH_TRUNC
  - cv2.THRESH_TOZERO
  - cv2.THRESH_TOZERO_INV

Adaptive Thresholding

- Simple thresholding işleminde tek tip bir eşik ile tüm piksellere aynı uygulama yapılıyor.
- Kullanacağımız görsele göre ya da yapacağımız işe göre her piksel için aynı eşik değeri kullanmak işimize yaramayabilir.
- Dolayısıyla bu noktada Adaptive thresholding tercih edilebilir.
- Resim etrafında küçük bir bölgeye göre bir pikselin eşiği alınıyor.
- Yani burada biraz daha gruplama, bölgesellendirme yapılarak eşik değeri alma durumu oluyor.

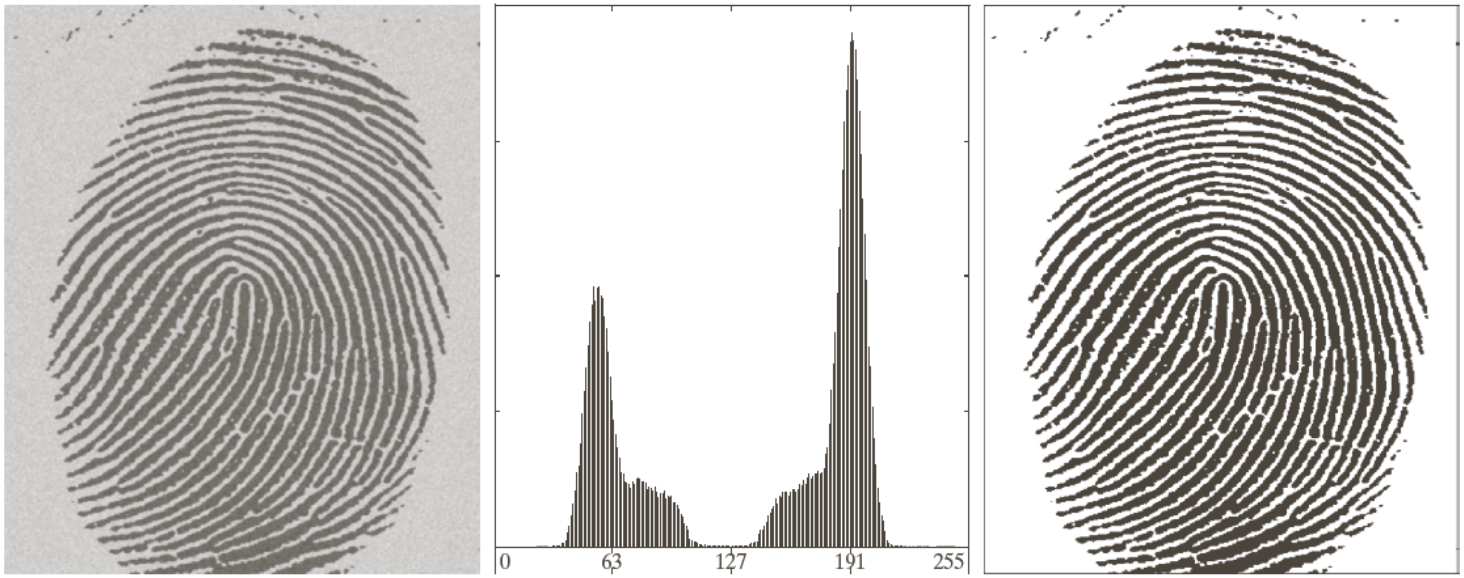
Otsu Thresholding

- Simple thresholding işleminde her piksel için aynı eşik değeri uygulanıyordu ve gürültülü görseller ya da karmaşık görseller için çok temiz bir sonuç elde edilemiyordu.
- Adapter thresholding ise bunu biraz daha iyileştiriyordu. Belli bölgelerde ortalamasını ya da gaussian'ını alarak buna göre işlemler yapıyordu ve daha temiz sonuçlar elde edilebiliyordu.
- Otsu thresholding ise, bir değer seçme zorunluluğunu ortadan kaldırıyor. Ve bu işlemi kendisi otomatik olarak belirliyor.
- Otsu thresholding ile genellikle daha temiz ve daha net görüntü elde edilebiliyor.
- Kullanılan resme göre ya da bizim belirlediğimiz eşik değerlerine göre ve istediğimiz sonuca göre thresholding türlerinin kullanım yerleri değişmektedir.

Thresholding Kod Örneği

```python
import cv2
image = cv2.imread("p.png",0)

#simple thresholding
ret1, thresh1 = cv2.threshold(image,127,255,cv2.THRESH_BINARY)

#otsu thresholding
ret2,thresh2 = cv2.threshold(image,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)

#otsu thresholding işlemi sonucunda 2 tane çıktı döndürüleceği için 2 tane değişken tanımlarız.
#otsu ile simple de cv2.threshold() fonksiyonu kullanılır.
#Aralarındaki fark: otsu thresholding'te otomatik olarak yapılmaktadır, hiçbir değişken veya değer
#seçme zorunluluğu yoktur. simple thresholding işleminde ise parametre olarak değerleri belirtmeliyiz.

#otsu thresholding işleminde threshold() fonksiyonu parametreleri:
#1.parametre: kullanacağımız resimdir.
#2. ve 3. parametrelerde bir eşik değeri belirlememiz istenir. simple thresholdingteki gibi belli
#bir eşik değeri vermiyoruz. Minimum ve maksimum değer belirtiyoruz. BGR değerleri 0 ile 255 arasında
#olduğu için eşik değerinin 0 ile 255 aralığında olduğunu belirten geniş bir değer belirliyoruz ve kendisi
#otomatik olarak 0 ın altına inmeden ve 255 in üstüne çıkmadan eşik değerini otomatik olarak
#seçiyor.
#4.parametre: cv2.THRESH_BINARY+cv2.THRESH_OTSU flag'i ile otsu thresholding yapılır.

cv2.imshow("orjinal resim",image)
cv2.imshow("simple thresholding",thresh1)
cv2.imshow("otsu thresholding",thresh2)

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

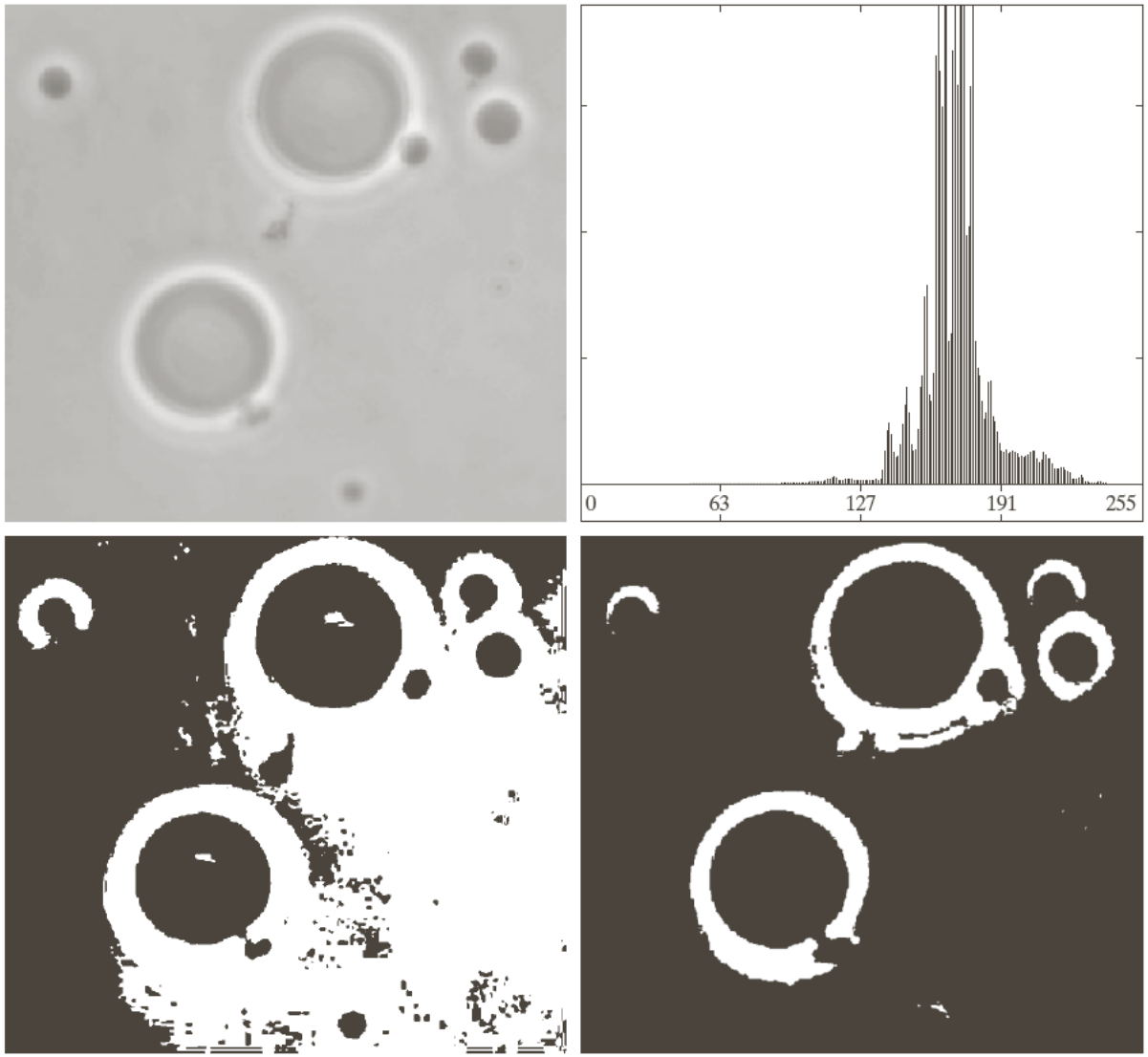
Thresholding Örneği



a b c

FIGURE 10.38 (a) Noisy fingerprint. (b) Histogram. (c) Segmented result using a global threshold (the border was added for clarity). (Original courtesy of the National Institute of Standards and Technology.)

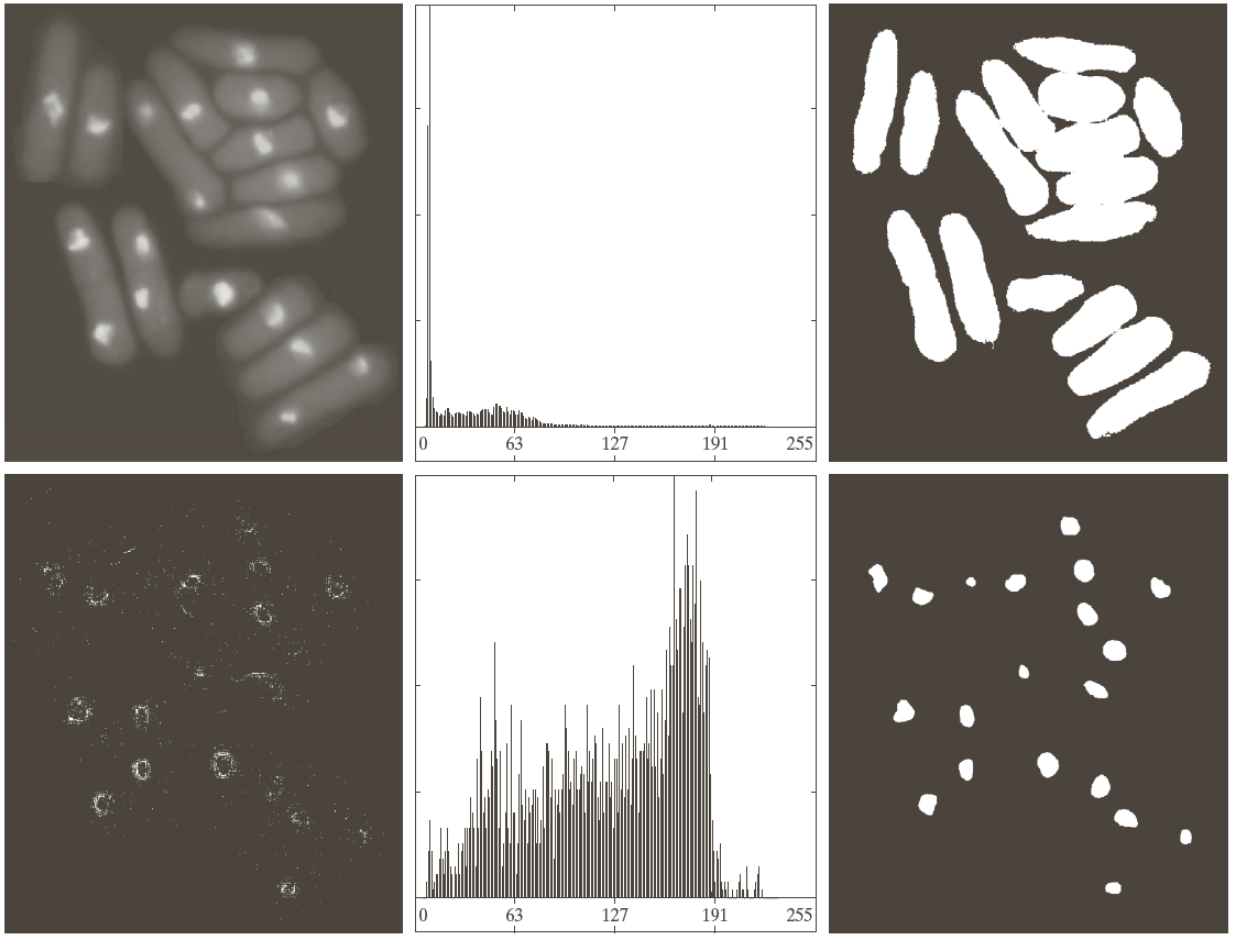
Thresholding Örneği



a b
c d

FIGURE 10.39
(a) Original image.
(b) Histogram (high peaks were clipped to highlight details in the lower values).
(c) Segmentation result using the basic global algorithm from Section 10.3.2.
(d) Result obtained using Otsu's method.
(Original image courtesy of Professor Daniel A. Hammer, the University of Pennsylvania.)

Thresholding Örneği



a b c
d e f

FIGURE 10.43 (a) Image of yeast cells. (b) Histogram of (a). (c) Segmentation of (a) with Otsu's method using the histogram in (b). (d) Thresholded absolute Laplacian. (e) Histogram of the nonzero pixels in the product of (a) and (d). (f) Original image thresholded using Otsu's method based on the histogram in (e). (Original image courtesy of Professor Susan L. Forsburg, University of Southern California.)

Canny Kenar Tespiti

- Beyaz gürültüyle bozulmuş basamak kenarlar için uygunudur.

- **Amaç**

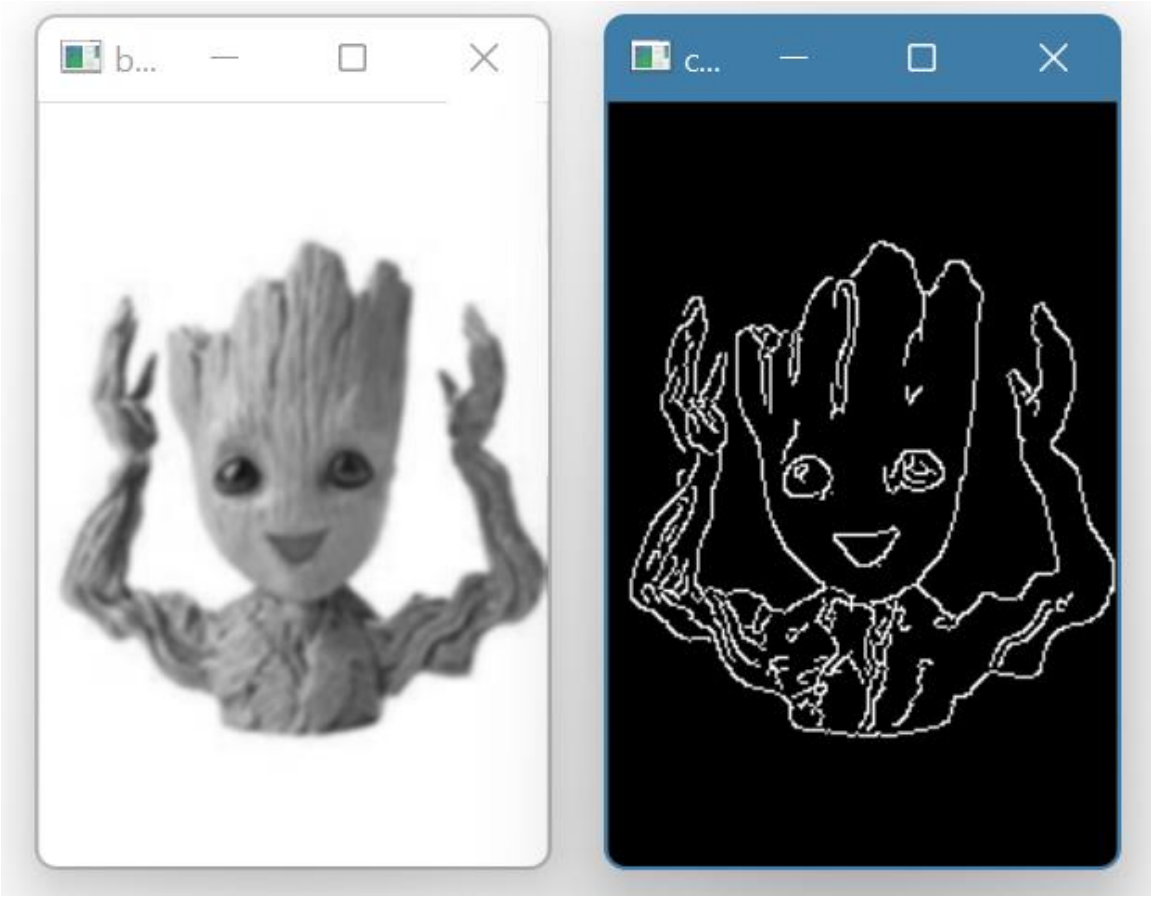
1. **Düşük hata oranı**
   Tespit edilen kenarlar gerçek kenara olabildiğince yakın olmalıdır.
2. **Kenar noktaları iyi lokalize edilmeli**
   Bulunan kenarlar gerçek kenarlara olabildiğince yakın olmalıdır.
3. **Tek kenar noktası yanıtı**
   Gerçek kenarın çevresinde yerel maksimum sayısı minimum olmalıdır.

Canny Kenar Tespiti Örnek

```
1   import cv2
2   import numpy as np
3
4   image = cv2.imread("groot.jpg")
5
6   #Resmi griye dönüştürelim:
7   gray = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
8
9   #1.parametre: Hangi resmi griye çevireceğimizin bilgisidir.
10  #2.parametre: cv2.COLOR_BGR2GRAY flag'i ile BGR dan GRAY'e değiştirme sağlanmıştır.
11
12  #Görüntüyü blurlayalım:
13  blur =  cv2.GaussianBlur(gray,(5,5),0)
14
15  #Resim gri resme dönüştürülmüştür ve sonrasında blurlama işlemi yapılmıştır. Şimdi bunun üstüne canny
16  #algoritmasını uygulayalım:
17  canny = cv2.Canny(blur,75,225)
18
19  #2.parametre ile alt eşik değeri 50 olarak, 3.parametre ile üst eşik değeri 150 olarak belirlenmiştir.
20
21  cv2.imshow("blurred image",blur)
22  cv2.imshow("canny image",canny)
23
24  cv2.waitKey(0)
25  cv2.destroAllWindows()
26
```

Canny Kenar Tespiti Örnek

Çıktı:



Sol taraftaki blur işleminin uygulandığı resim ve sağ taraftaki canny algoritmasının uygulanması sonucunda elde edilen resimdir.

Canny Kenar Tespiti Örnek

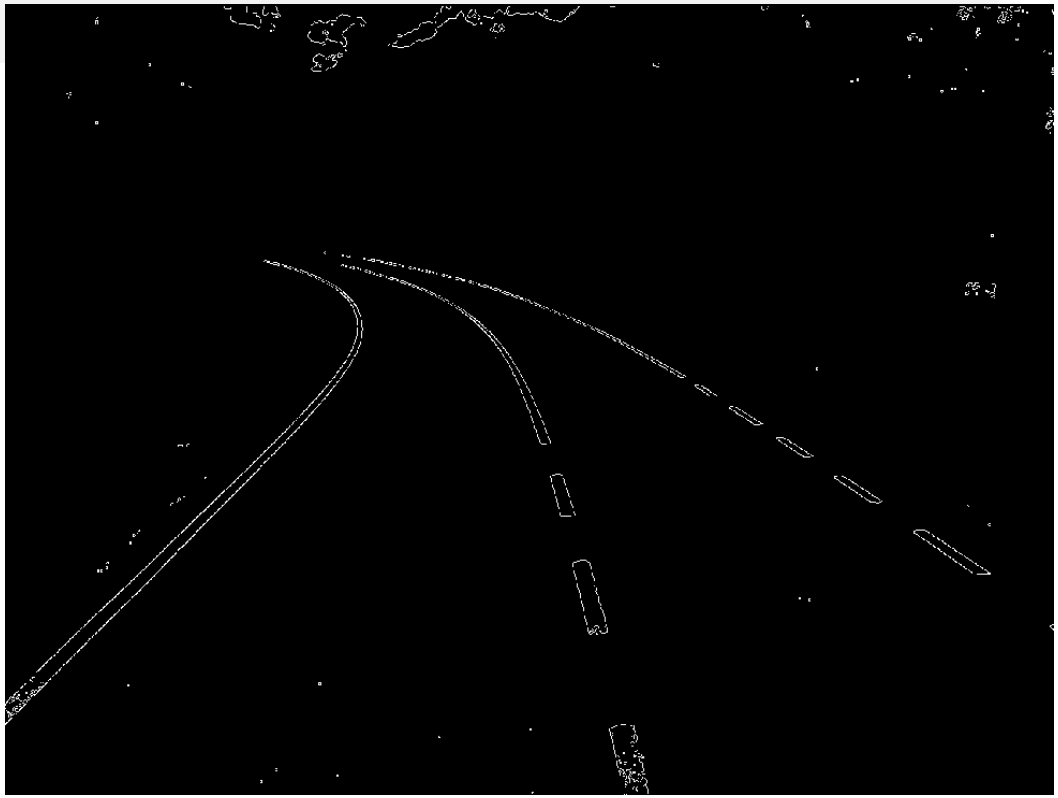
- Gerçekleştirilen işlemler şu şekildedir:
  - Canny, kenar algılama için görüntü işlemede çokça kullanılan algoritmalardan birisidir.
  - Kenar belirlemek için belli eşik değerleri verilir.
  - Kenar algılama işlemlerini gri resim üzerinden yapmamız gerektiğinden öncelikle resim gri tona çevrilmiştir. Ve daha sonra blurlama işlemi gerçekleştirilmiştir.
  - Blurlama işlemi yapma sebebimiz; kenar algılarken eşik değeri belirleyeceğiz fakat resmin bazı kısımlarında istemediğimiz kenarlar olabilir. O kenarları olabildiğince tölere edebilmek adına blurlama (yumuşatma) işlemi yapıyoruz.

Canny Kenar Tespiti Örnek

- Araçlar için otonom sürüş pilotu eğitmek istediğimizi düşünelim.
- Bu problem için yandaki resim incelendiğinde öncelikle yapay zeka pilotumuzun yolu ve şeritleri anlayabilmesi gerekir.
- Bunun için OpenCV kütüphanesinden destek alalım.

Canny Kenar Tespiti Örnek

```
img = cv2.imread(img_path)
gray_image = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
(thresh, output2) = cv2.threshold(gray_image, 200, 255,
cv2.THRESH_BINARY)
output2 = cv2.Canny(output2, 180, 255)
plt.imshow(output2)
plt.show()
```



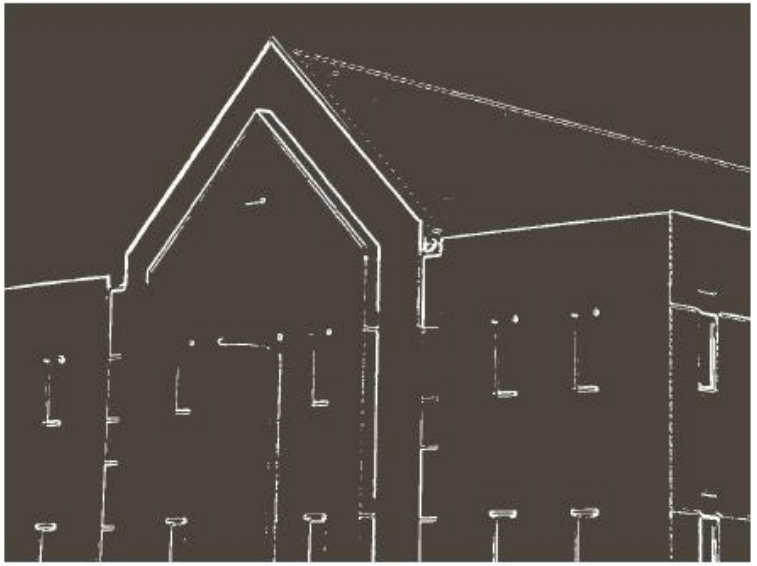
Canny Kenar Tespiti Örnek

```
img = cv2.imread(img_path)
gray_image = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
(thresh, output2) = cv2.threshold(gray_image, 200, 255,
cv2.THRESH_BINARY)
output2 = cv2.GaussianBlur(output2, (5, 5), 3)
output2 = cv2.Canny(output2, 180, 255)
plt.imshow(output2)
plt.show()
```

GaussianBlur etkisini daha iyi gözlemleyebilmek adına aynı işlemler bu defa bulanıklaştırma kullanılarak yapılmıştır.

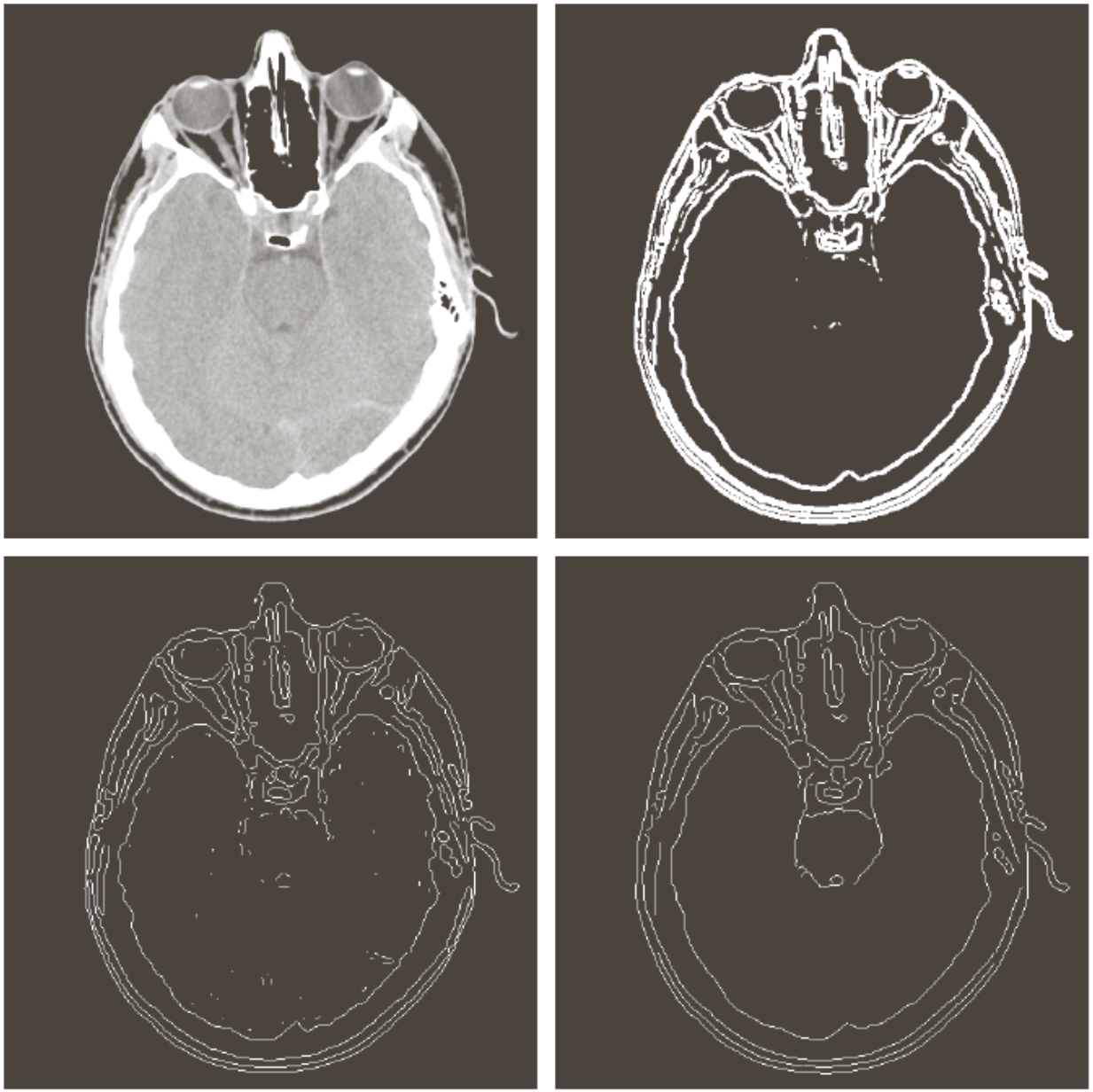
Canny Kenar Tespiti: Özet

- Giriş görüntüsünü Gauss filtresi ile pürüzsüz hale getirin
- Gradyan büyüklüğü ve açı görüntülerini hesapla
- Gradyan büyüklüğü resmine nonmaxima bastırma uygulayın
- Kenarları algılamak ve bağlamak için çift eşik değerleme ve bağlantı analizi kullanın



a b
c d
ŞEKİL 10.25
(a) Yeğinlik değerleri [0, 1] aralığında ölçeklenmiş olan 834 × 1114 boyutlarındaki görüntü. (b) Yumuşatılmış görüntünün eşik uygulanmış gradyanı. (c) Marr-Hildreth algoritması kullanılarak elde edilen görüntü. (d) Canny algoritması kullanılarak elde edilen görüntü. Diğer iki görüntü ile karşılaştırıldığında Canny algoritmasındaki iyileştirmeye dikkat ediniz.

$T_L = 0.04; T_H = 0.10; \sigma = 4$ and a mask of size 25×25



a b
c d
ŞEKİL 10.26
(a) Yeğinlik değerleri [0, 1] aralığında ölçeklenmiş olan 512 × 512 piksel boyutlarındaki orijinal kafa CT görüntüsü. (b) Yumuşatılmış görüntünün eşik uygulanmış gradyanı. (c) Marr-Hildreth algoritması kullanılarak elde edilen görüntü. (d) Canny algoritması kullanılarak elde edilen görüntü. (Orijinal görüntü Dr. David R. Pickens, Vanderbilt University izniyle alınmıştır.)

$T_L = 0.05; T_H = 0.15; \sigma = 2$ and a mask of size 13×13

Görüntü Bölütleme

- Image segmentation is an image processing task in which the image is segmented or partitioned into multiple regions such that the pixels in the same region share common characteristics.

Kaynaklar

- Sayısal Görüntü İşleme, Palme Yayıncılık, Üçüncü Baskıdan Çeviri (*Orj: R.C. Gonzalez and R.E. Woods: "Digital Image Processing", Prentice Hall, 3rd edition, 2008*).
- "Digital Image Processing Using Matlab", Gonzalez & Richard E. Woods, Steven L. Eddins, Gatesmark Publishing, 2009
- Ders Notları, CS589-04 Digital Image Processing, F.(Qingzhong) Liu, http://www.cs.nmt.edu/~ip
- Ders Notları, BIL717-Image Processing, E.Erdem
- Ders Notları, EBM537-Görüntü İşleme, F.Karabiber
- https://www.youtube.com/watch?v=6ISRjlB4yTY
- https://medium.com/@adem.akdoğan/
- https://machinelearningknowledge.ai/image-segmentation-in-python-opencv/
- Bekir Aksoy, Python ile İmgeden Veriye Görüntü İşleme ve Uygulamaları, Nobel Akademik Yayıncılık