Canny

Canny Kenar Yakalama Algoritması

Bilinen en iyi kenar yakalama yöntemi

1. YUMUŞATMA: Görüntüyü Gaussian fonksiyonu ile yumuşat.

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

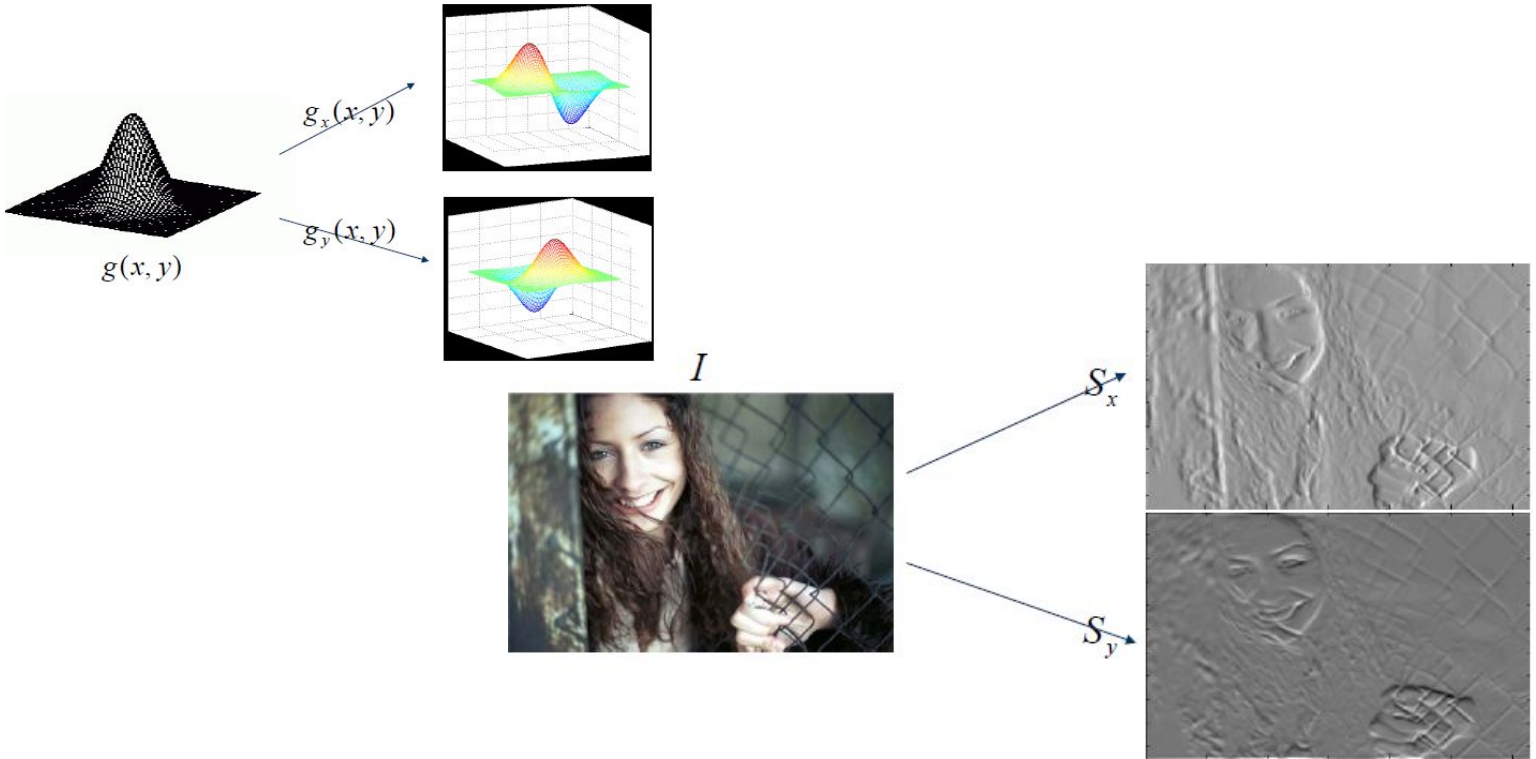
$$f_s(x,y) = G(x,y) \bigstar f(x,y)$$

2. Gradientleri Bulma (Yatay (g_x) ve dikey (g_y))
3. Gradient büyüklüğünü ve yönünü bul
4. KENAR İNCELTME AŞAMASI
5. EŞİKLEME AŞAMASI
6. KENAR BAĞLAMA AŞAMASI

Canny Kenar Yakalama Algoritması

1. ve 2. aşama

İlk iki aşama tek bir işlemle yapılabilir. Bunun için gaussian yumuşatma fonksiyonunun yatay ve dikey türevlerinin görüntüyle konvolüsyona tabi tutulması gerekir.



Canny Kenar Yakalama Algoritması

3. aşama

(S_x, S_y) Gradient Vector

$$\text{magnitude} = \sqrt{(S_x^2 + S_y^2)}$$

$$\text{direction} = \theta = \tan^{-1}\frac{S_y}{S_x}$$

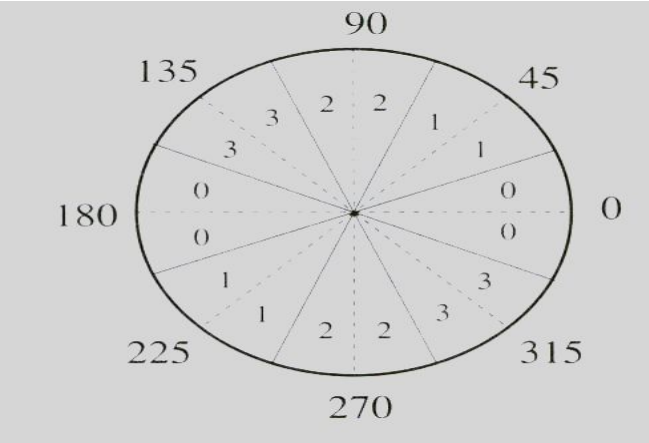
image

gradient magnitude

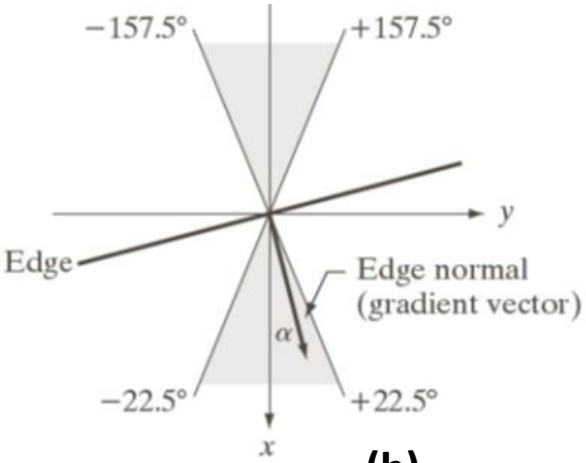
Canny Kenar Yakalama Algoritması

4. aşama: Kenar İnceltme (suppression)

- θ[i,j] açı değerlerini **(a)** şeklini kullanarak 4 sektör haline indirge.
- M[i,j]'deki her 3x3 pencereyi incele.
- Gradient yönüne bak (Gradient yönü kenara diktir). Eğer merkez değer bu yöndeki komşu değerlerin herhangi birinden küçükse o zaman merkez değeri sıfırla (M[i,j]=0). Örneğin **(b)** şeklinde gradient yönünün α olduğu görülüyor, dolayısıyla gri bölgelerdeki magnitude değerlerinden herhangi biri merkezden büyükse pikselin magnitude değeri silinir.



(a)



(b)

Canny Kenar Yakalama Algoritması

4. aşama: Kenar İnceltme (suppression)

$$M = \begin{matrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 3 \\ 0 & 0 & 0 & 1 & 2 & 1 & 3 & 1 \\ 0 & 0 & 2 & 1 & 2 & 1 & 1 & 0 \\ 0 & 1 & 3 & 2 & 1 & 1 & 0 & 0 \\ 0 & 3 & 2 & 1 & 0 & 0 & 1 & 0 \\ 2 & 3 & 2 & 0 & 0 & 1 & 0 & 1 \\ 2 & 3 & 2 & 0 & 1 & 0 & 2 & 1 \end{matrix}$$

lokal maksimum

silinen

şarta bağlı

Kırmızı çizgiler kenara dik olan **gradient vektör yönlerini** göstermektedir. Bunun için ilk olarak θ açı matrisine bakılarak kırmızı çizgi yönü belirleniyor. Bu eksendeki M(i,j) değerlerine bakılıyor. Merkezden büyük bir değer yakalanırsa piksel

Canny Kenar Yakalama Algoritması

4. aşama: Kenar İnceltme (suppression)

$$M_{thin} =$$

0	0	0	0	0	0	0	3
0	0	0	0	2	1	3	0
0	0	2	1	2	0	0	0
0	0	3	0	0	0	0	0
0	3	2	0	0	0	0	0
0	3	0	0	0	1	0	1
0	3	0	0	1	0	2	0

Yanlış kenarlar kalabilir

- Gürültüden dolayı kenar inceltme aşaması birçok yanlış kenarı içerebilir.

Canny Kenar Yakalama Algoritması

5. aşama: Eşikleme

Klasik Eşikleme

- Bir eşik T uygulayarak yanlış kenar sayısını azalt
  - T den küçük tüm değerleri 0 yap
  - İyi bir T değeri seçmek çok zordur.
  - T çok küçük seçilirse yanlış kenarlar kalır (False Positives)
  - T büyük seçilirse doğru kenarlar silinebilir (False Negatives)

Canny Kenar Yakalama Algoritması

5. aşama: Eşikleme

Canny-Çift Eşik Kullanma

- Kenarı inceltilmiş görüntüde çift eşik kullan
  - $T_2 = 2T_1$
  - Çıkışta iki imge elde edilir
  - T_2 sonucu güçlü kenarları içerir. Kenarlarda kopukluklar vardır.
  - T_1 sonucu zayıf kenarları içerir. Yanlış kenar sayısı çoktur.
  - T_1 ve T_2 sonucu birleştirilir.

Güçlü kenar pikseli: $G_{T_2}(x,y) = M_{thin}(x,y) \geq T_2$

Zayıf kenar pikseli: $G_{T_1}(x,y) = M_{thin}(x,y) \geq T_1$

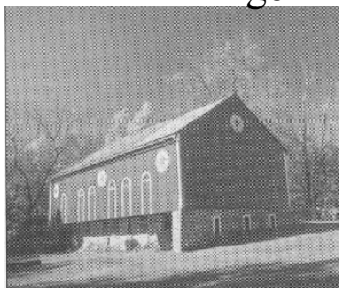
G_{T_1} içerisinden G_{T_2} yi temizlemek için ilk olarak aşağıdaki işlem yapılır:

$$G_{T_1}(x,y) = G_{T_1}(x,y) - G_{T_2}(x,y)$$

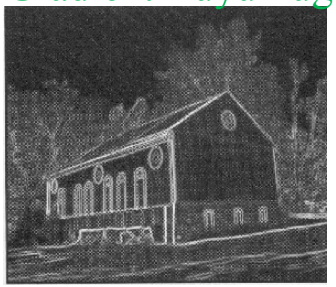
Canny Kenar Yakalama Algoritması

5. aşama: Eşikleme

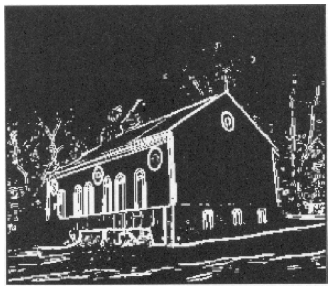
İmge



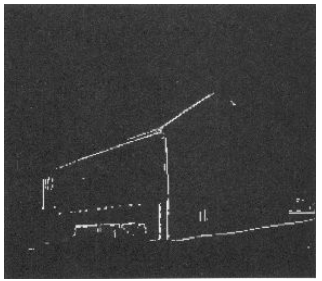
Gradient Büyüklüğü (M)



Düşük Eşik (G_{T_1})

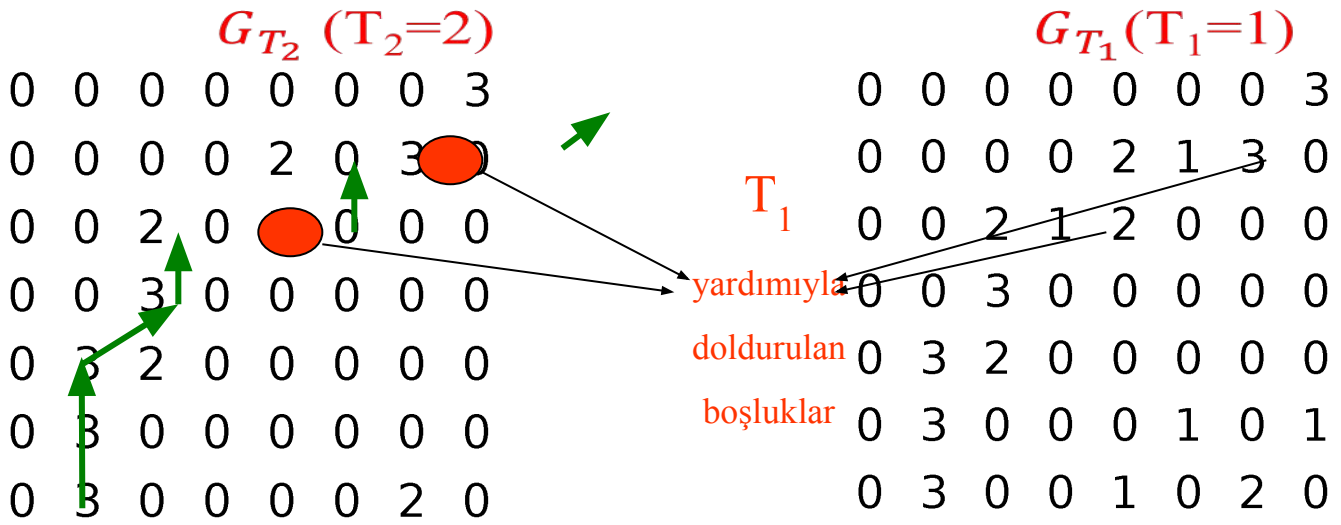


Yüksek Eşik (G_{T_2})



Canny Kenar Yakalama Algoritması

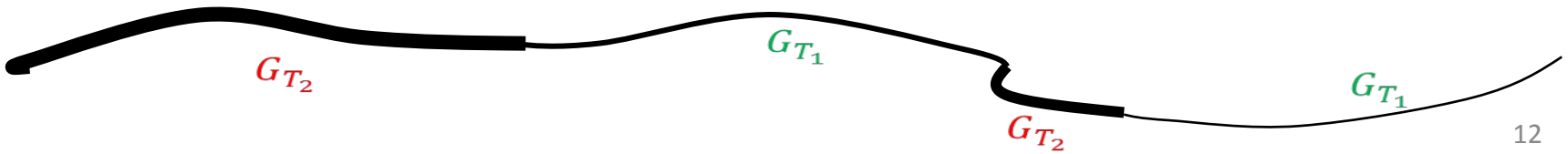
5. aşama: Eşikleme



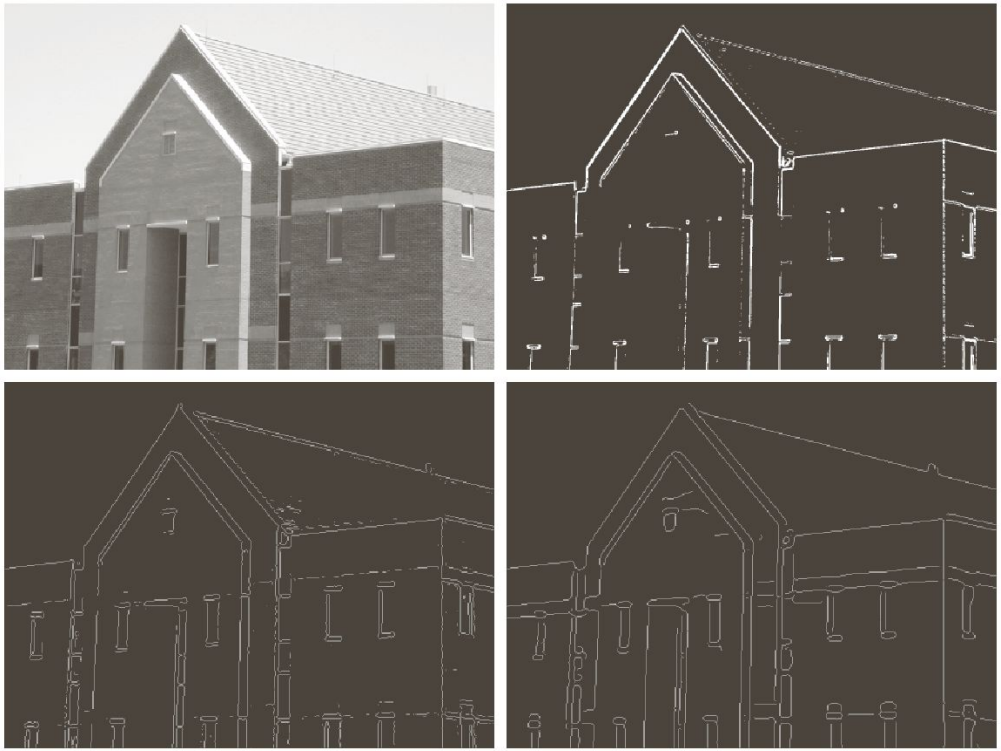
Canny Kenar Yakalama Algoritması

6. aşama: Kenar Bağlama

1. G_{T_2} daki ziyaret edilmeyen bir sonraki kenar pikseline p konumlan
2. p nin 8 komşuluğunda olan G_{T_1} deki tüm zayıf pikselleri geçerli kenar pikseli olarak işaretle.
3. G_{T_2} deki kesin kenarların tamamı bu şekilde ziyaret edildiyse 4. aşamaya geç, değilse 1. aşamaya geç.
4. G_{T_1} deki geçerli kenar piksellerini 1, diğerlerini 0 olarak işaretle.
5. G_{T_1} deki sıfır olmayan piksellerin tamamını G_{T_2} a ekle.
6. Canny algoritmasının sonucu: G_{T_2}



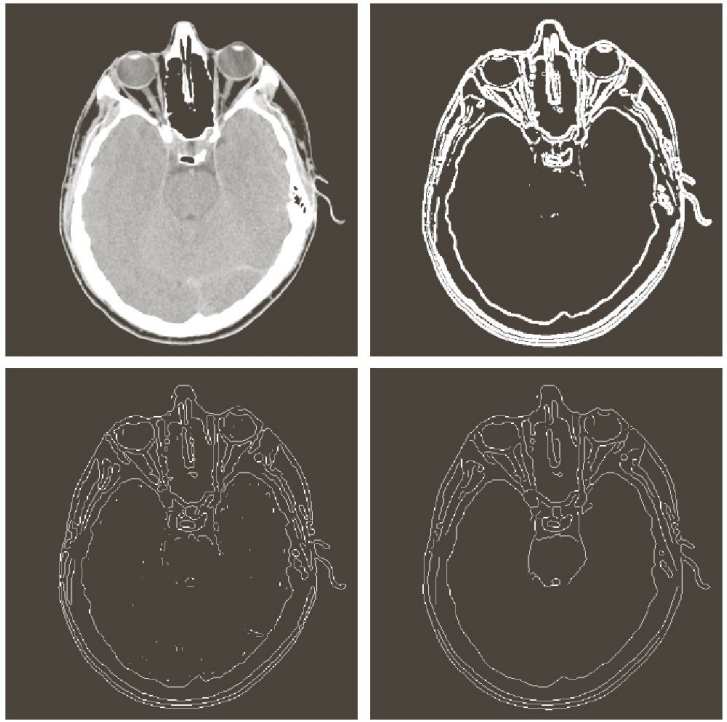
Canny Örnek-1



a b
c d

FIGURE 10.25
(a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1].
(b) Thresholded gradient of smoothed image.
(c) Image obtained using the Marr-Hildreth algorithm.
(d) Image obtained using the Canny algorithm. Note the significant improvement of the Canny image compared to the other two.

Canny Örnek-2



a b
c d

FIGURE 10.26
(a) Original head CT image of size 512 × 512 pixels, with intensity values scaled to the range [0, 1]. (b) Thresholded gradient of smoothed image. (c) Image obtained using the Marr-Hildreth algorithm. (d) Image obtained using the Canny algorithm. (Original image courtesy of Dr. David R. Pickens, Vanderbilt University.)