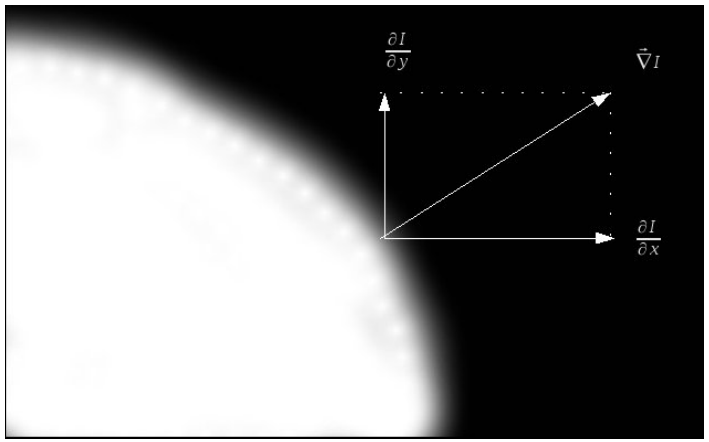
Gradiente - Laplaciano - Canny

IPDI 2C 2014-

DC- FCEyN - UBA Argentina


$\frac{\partial I}{\partial y}$
$\vec{\nabla} I$
$\frac{\partial I}{\partial x}$

Sea $y = f(x, y)$ una función 2-D, el gradiente se define:

$$\nabla f(\vec{x}, y) = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \\ \frac{\partial f}{\partial y} \end{pmatrix}$$

$$\frac{\partial f(x,y)}{\partial x} \rightarrow [f(n+1, m+1) - f(n-1, m+1)] + [f(n+1, m) - f(n-1, m)] + [f(n+1, m-1) - f(n-1, m-1)]$$

$$\frac{\partial f(x,y)}{\partial y} \rightarrow [f(n+1, m+1) - f(n+1, m-1)] + [f(n, m+1) - f(n, m-1)] + [f(n-1, m+1) - f(n-1, m-1)] \quad , , ,$$

Sea f una imagen, el vector $\vec{\nabla} f$, y está definido como:

$$\vec{\nabla} f = \frac{\partial f}{\partial x}\vec{i} + \frac{\partial f}{\partial y}\vec{j} = \begin{pmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \end{pmatrix}^t$$

Los bordes son cambios abruptos de intensidad. Para detectarlos consideramos el gradiente, dado que la dirección del gradiente es la de máxima variación y es perpendicular al borde que se forma en la región donde se encuentra el borde.

Detectar un borde es encontrar la **magnitud del gradiente**:

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

El **ángulo** que forma el vector gradiente con el eje x,

$$\theta_G = arctg\left(\frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial x}}\right)$$

Si consideramos una ventana de 2×2, alrededor de cada punto (i, j) de modo que:

$$\left(\begin{array}{c|c} f(i,j) & f(i,j+1) \\ \hline f(i+1,j) & f(i+1,j+1) \end{array} \right)$$

en este caso una manera de aproximar la derivada primera es mediante:

$$\begin{aligned} \frac{\partial f}{\partial x} &= f(i+1,j) - f(i,j) \\ \frac{\partial f}{\partial y} &= f(i,j+1) - f(i,j) \end{aligned}$$

otra forma es haciendo diferencias cruzadas:

$$\begin{aligned} \frac{\partial f}{\partial x} &= f(i,j) - f(i+1,j+1) \\ \frac{\partial f}{\partial y} &= f(i,j+1) - f(i+1,j) \end{aligned}$$

Si en cambio son de 3×3, alrededor del punto (i, j)

$$\left(\begin{array}{c|c|c} f(i-1,j-1) & f(i-1,j) & f(i-1,j+1) \\ \hline f(i,j-1) & f(i,j) & f(i,j+1) \\ \hline f(i+1,j-1) & f(i+1,j) & f(i+1,j+1) \end{array}\right)$$

aproximar la derivada primera puede hacerse:

$$\begin{aligned} \frac{\partial f}{\partial x} &= (f(i+1,j-1) + f(i+1,j) + f(i+1,j+1)) - (f(i-1,j-1) + f(i-1,j) + f(i-1,j+1)) \\ \frac{\partial f}{\partial y} &= (f(i,j+1) + f(i,j+1) + f(i+1,j+1)) - (f(i-1,j-1) + f(i,j-1) + f(i+1,j-1)) \end{aligned}$$

Roberts - Prewitt

Roberts

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

Prewitt

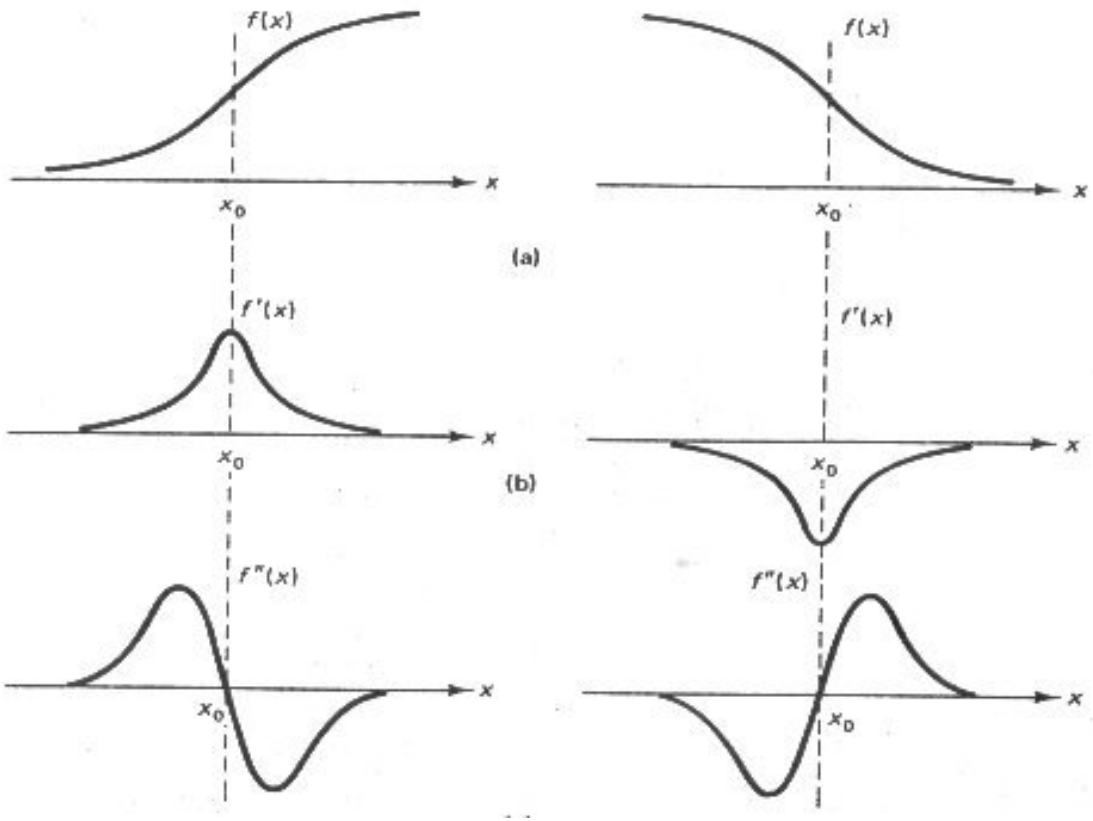
$$\begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$

Sobel

$$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

Estas máscaras son generan convolucinando un filtro de bordes con un filtro de suavizado. Encuentra bordes suavizando ruido.
Tomando el filtro de gradiente $D^t = [-1 \quad 0 \quad 1]$ con una máscara de suavizado $G^t = [1 \quad 2 \quad 1]$, $D \otimes G^t$ permite armar las máscaras correspondientes a la $\frac{\partial f}{\partial x}$ y $\frac{\partial f}{\partial y}$.

Laplaciano



El laplaciano de una función $f(x, y)$ es una derivada de segundo orden definida por

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

Se puede aproximar como

$$\frac{\partial f}{\partial x} \rightarrow f_x(n_1, n_2) = f(n_1 + 1, n_2) - f(n_1, n_2)$$

$$\frac{\partial^2 f}{\partial x^2} \rightarrow f_{xx}(n_1, n_2) = f_x(n_1, n_2) - f_x(n_1 - 1, n_2) =$$

$$(f(n_1 + 1, n_2) - f(n_1, n_2)) - (f(n_1, n_2) - f(n_1 - 1, n_2)) =$$

$$f(n_1 + 1, n_2) - 2f(n_1, n_2) + f(n_1 - 1, n_2)$$

$$\frac{\partial^2 f}{\partial y^2} \rightarrow f_{yy}(n_1, n_2) = f_y(n_1, n_2) - f_y(n_1, n_2 - 1) =$$

$$(f(n_1, n_2 + 1) - f(n_1, n_2)) - (f(n_1, n_2) - f(n_1, n_2 - 1)) =$$

$$f(n_1, n_2 + 1) - 2f(n_1, n_2) + f(n_1, n_2 - 1)$$

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \rightarrow f(n_1 + 1, n_2) - 4f(n_1, n_2) + f(n_1 - 1, n_2) +$$

$$f(n_1, n_2 + 1) + f(n_1, n_2 - 1)$$

$$\nabla^2 f \rightarrow -4f(i,j) + f(i-1,j) + f(i,j-1) + f(i,j+1) + f(i+1,j)$$

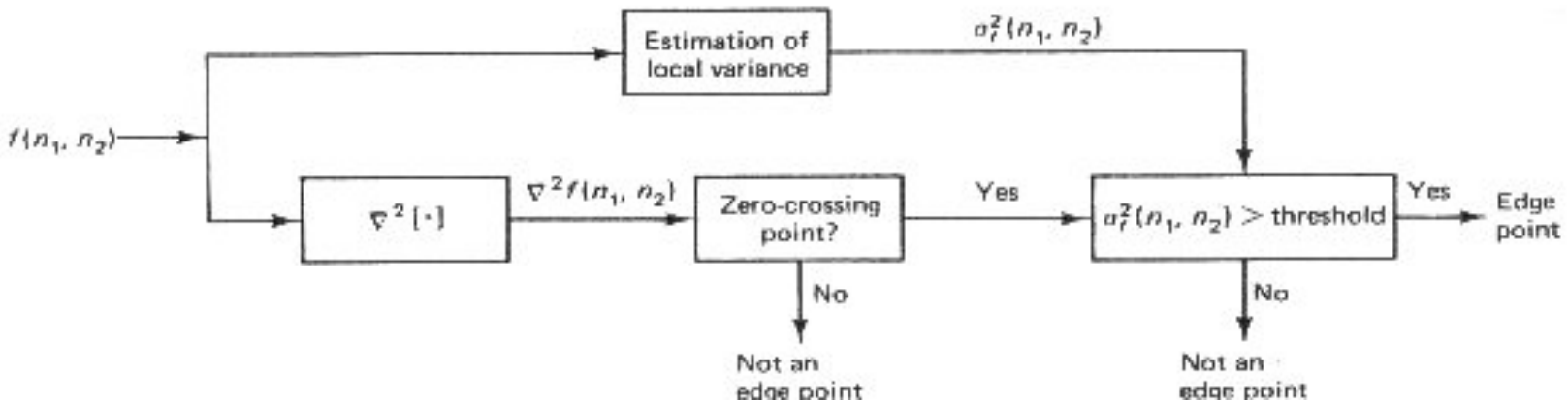
$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Su uso más frecuente es encontrar la ubicación de los bordes, utilizando sus propiedades de paso por 0, *zero crossing*.

- Los resultados del laplaciano generan muchos bordes falsos, en general en regiones donde la varianza local es pequeña. Un caso especial es considerar el background como constante, en cuyo caso la derivada segunda será cero.
- En caso de zero crossing es donde aparecerán cambios de signo, indice que hay un borde.
- método para remover falsos contornos consiste chequear que la varianza local sea suficientemente grande en esos puntos según se muestra en el esquema 1.

Detección de Bordes basada en el Laplaciano

Sistema basado en el laplaciano, reduce la cantidad de bordes falsos.



La varianza local $\sigma_f^2(n_1, n_2)$ puede estimarse como:

$$\sigma_f^2(n_1, n_2) = \frac{1}{(2M+1)^2} \sum_{k_1=n_1-M}^{n_1+M} \sum_{k_2=n_2-M}^{n_2+M} [f(n_1, n_2) - m_f(k_1, k_2)]^2$$

donde

$$m_f(k_1, k_2) = \frac{1}{(2M+1)^2} \sum_{k_1=n_1-M}^{n_1+M} \sum_{k_2=n_2-M}^{n_2+M} f(n_1, n_2)$$

que es el valor medio, y M cercano a 2(LIM).

Como $\sigma_f^2(n_1, n_2)$ se compara con un umbral, el factor de escala $\frac{1}{(2M+1)^2}$ puede suprimirse.
La varianza local σ_f^2 se evalúa para aquellos (n_1, n_2) que son zero crossing para $\nabla^2 f(n_1, n_2)$ En el esquema 1 se observa que la varianza está estrechamente relacionada con la magnitud del gradiente, comparar $\sigma_f^2(n_1, n_2)$ con un umbral es similar a comparar el gradiente con un umbral.
Cuando $\nabla^2 f(n_1, n_2) = 0$ existe la posibilidad de que allí haya un borde.
El sistema evalúa si $\sigma_f^2(n_1, n_2)$ supera cierto umbral predeterminado, *sólo* en los puntos zero crossing de $\nabla^2 f(n_1, n_2)$,.

Detección de bordes según Marr y Hildreth

Marr y Hildreth sugiere que la imagen original sea bandalimitada. Para *bandalimitar* una imagen M&H propone un la respuesta al impulso:

$$h(x, y) = e^{-(x^2+y^2)/(2\pi\sigma^2)}$$

y la respuesta en frecuencia:

$$H(\Omega_x, \Omega_y) = 2\pi^2\sigma^2 e^{-\pi\sigma^2(\Omega_x^2+\Omega_y^2)/2}$$

que es el filtro gaussiano donde σ deterina la frecuencia de corte(cutoff frecuency)

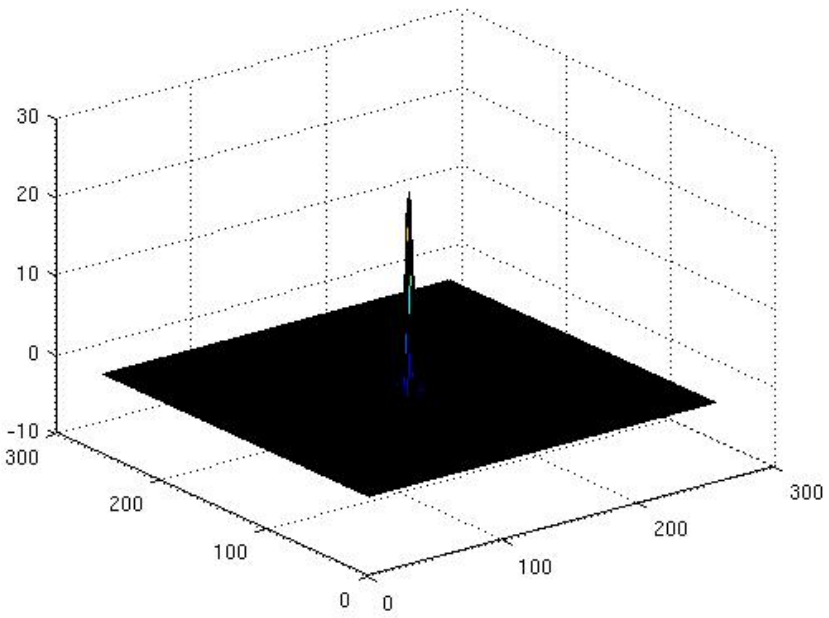
$$\nabla^2(f(x,y)*h(x,y)) = f(x,y)*[\nabla^2 h(x,y)] = f(x,y)*[\frac{\partial^2 h(x,y)}{\partial x^2} + \frac{\partial^2 h(x,y)}{\partial y^2}]$$

$$\nabla^2 h(x,y) = \frac{e^{-(x^2+y^2)/(2\pi\sigma^2)}}{(\pi\sigma^2)^2}(x^2+y^2-2\pi\sigma^2)$$

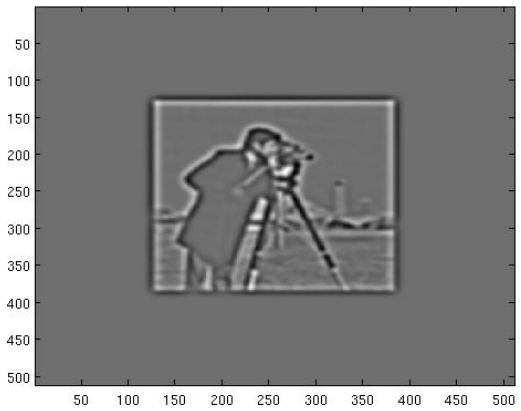
$$F[\nabla^2 h(x,y)] = -2\pi^2\sigma^2 e^{-\pi\sigma^2(\Omega_x^2+\Omega_y^2)/2}(\Omega_x^2+\Omega_y^2)$$

```
%generar máscara gaussiana(toda la imagen)
sigma=4.
[x,y]=meshgrid(1:256,1:256);
a=x.*x+y.*y;
b=exp(-a/(2*pi*sigma));
c=(x.*x+y.*y-2*pi*(sigma*sigma));
d=b.*c;
surf(d)
```

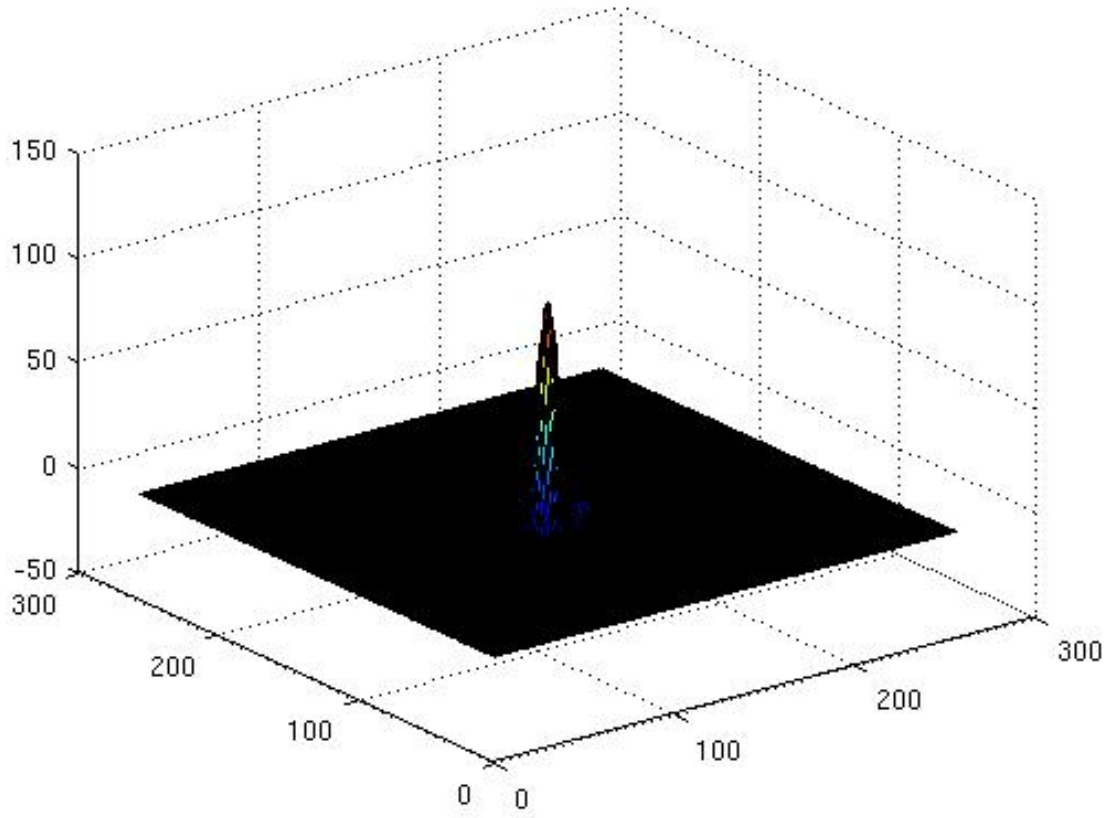
```
%generar máscara gaussiana
[x,y]=meshgrid(1:10,1:10);
x=x-5;
y=y-5;
sigma=1;
a=x.*x+y.*y;
b=exp(-a/(2*pi*(sigma*sigma)));
c=(x.*x+y.*y-2*pi*(sigma*sigma));
d=-b.*c;
figure(1),surf(d)
```


30
20
10
0
-10
300
200
100
0
0
100
200
300

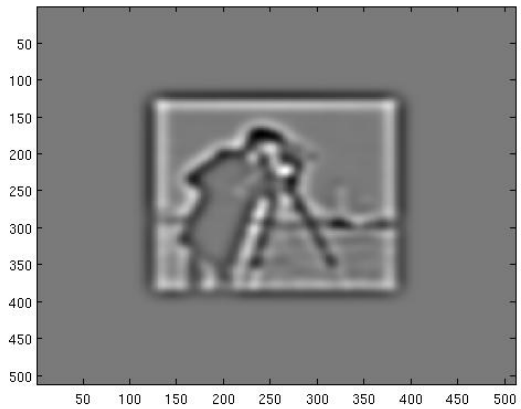
(a)


50
100
150
200
250
300
350
400
450
500
50 100 150 200 250 300 350 400 450 500

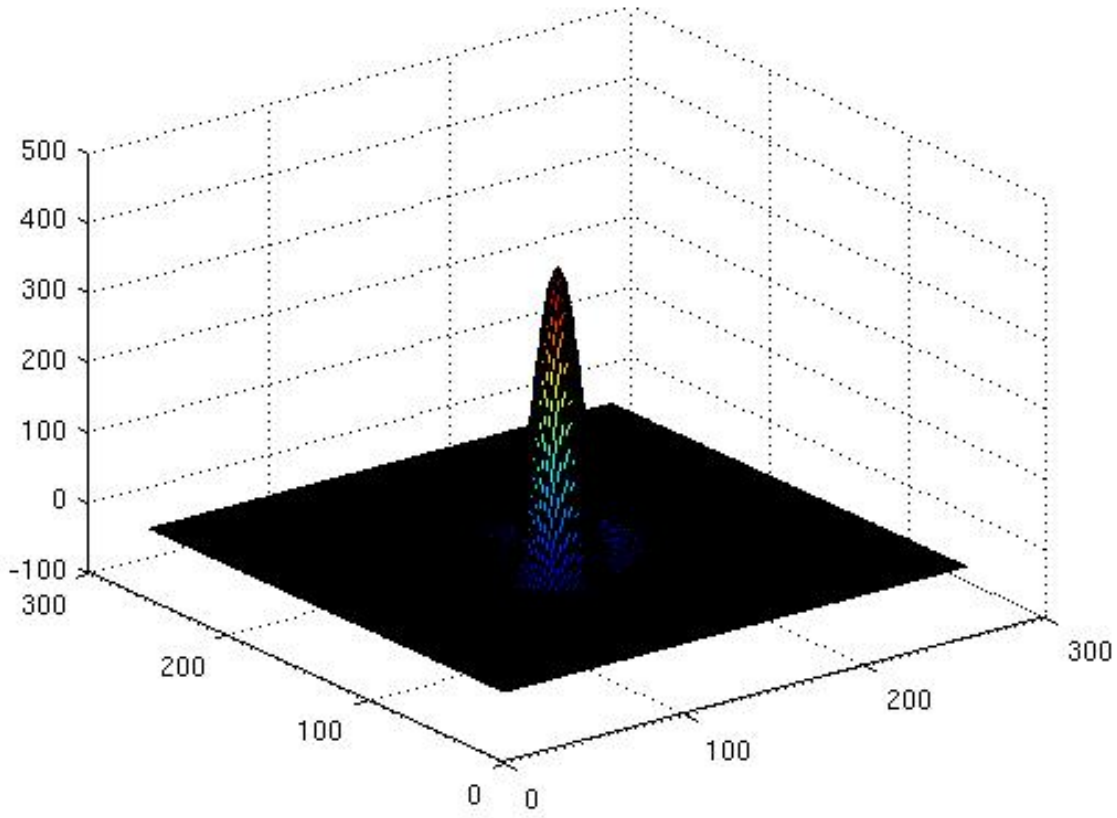
(b)


150
100
50
0
-50
300
200
100
0
0
100
200
300

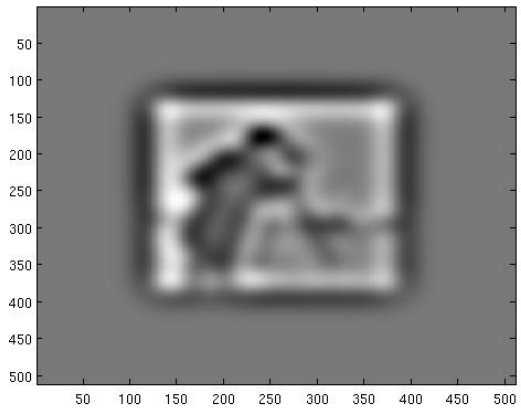
(c)


50
100
150
200
250
300
350
400
450
500
50
100
150
200
250
300
350
400
450
500

(d)


500
400
300
200
100
0
-100
300
200
100
0
0
100
200
300

(e)


50
100
150
200
250
300
350
400
450
500
50
100
150
200
250
300
350
400
450
500

(f)

Ejercitación en práctica: Implementar

1. Método del Laplaciano
2. Método del Laplaciano agregándole evaluación local de varianza
3. Método del Laplaciano del Gaussiano(Marr- Hildreth) para distintos valores de σ.

Biliografía: *Two Dimensional Signal and Image Processing*, Jae S.Lim, Prentice Hall Signal Processing series.

Criterios de detección de Canny

- Evitar la eliminación de bordes importantes y no suministrar falsos bordes.
- Localización: minimizar la distancia entre la posición real y la localizada del borde.
- Convertir los bordes múltiples en uno solo.

Síntesis

- Detección
- Localización
- Respuesta que integre las respuestas múltiples correspondientes a un único borde.

Algoritmo de Canny

- Obtención del gradiente: obtener magnitud y orientación del vector gradiente en cada píxel.
- Supresión de no máximos: Conseguir el adelgazamiento del ancho de los bordes, obtenidos con el gradiente, hasta lograr bordes de un píxel de ancho.
- Histéresis de umbral: Aplicar esta función de histéresis que está basada en dos umbrales; se quiere reducir la posibilidad de bordes falsos.

1. Algoritmo para Obtención del gradiente

Aplicar un filtro gaussiano para eliminar ruido y suavizar la imagen.
gradiente en cada píxel.

Máscaras gaussianas sugeridas para suavizar que se obtiene promediando los valores de intensidad de los pixels en el entorno de vecindad con una máscara de convolución con un cierto σ.

(a)

$$\frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

(b)

$$\frac{1}{115}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

Algoritmo para obtener el gradiente

Entrada: I imagen, máscara de convolución H.
Salida: I_m magnitud del gradiente, I_o orientación del gradiente
1. Suavizar la imagen I mediante el filtro gaussiano H y obtener la imagen de salida J.
2. Para cada pixel (i, j) en J, obtener la magnitud y orientación del gradiente según: detecc_de_bordes.pdf
3. Obtener I_m e I_o

2. Supresión no máximos del gradiente

I_m e I_o son entradas.
Las orientaciones son 0°, 45°, 90° y 135° con respecto al eje horizontal.
Para cada píxel se encuentra la dirección que más aproxime a la dirección del ángulo de gradiente.
Si el valor de la magnitud de gradiente es menor que al menos uno de sus dos vecinos en la dirección del ángulo obtenida en el paso anterior, entonces se asigna el valor 0 a dicho píxel, sino se asigna el valor que tenga la magnitud del gradiente.
La salida de este segundo paso es la imagen I_n de bordes, es decir, $I_m(i, j)$, después de la supresión no máxima de puntos de borde.

Algoritmo: Supresión no máximos

Entrada: I_m e I_o
Salida: I_n
Dadas cuatro direcciones d_1, d_2, d_3 y d_4 que representan a $0°, 45°, 90° y 135°$ con respecto al eje horizontal
1. Para cada (i,j) :
1.1. Encontrar la dirección d_k que mejor se aproxima a la dirección $I_o(i,j)$, o sea la perpendicular al borde.
1.2. Si $I_m(i,j)$ es menor a al menos uno de sus dos vecinos en la dirección d_k , al píxel (i, j) de I_n se le asigna el valor 0 o sea $I_n(i,j) = 0$ (supresión), sino $I_n(i,j) = I_m(i,j)$.
2. Devolver I_n

3. Umbral por Histéresis

I_n suele contener máximos locales creados por el ruido. Para dar solución y eliminar dicho ruido se calcula el umbral por histéresis:

- considerar la imagen obtenida del paso anterior
- considerar la matriz de orientaciones de los puntos de borde de la imagen
- tomar dos umbrales U_{min} y U_{max}
- Para cada pixel de la imagen, si el valor de la magnitud del pixel considerado es mayor a U_{max}, entonces ese pixel es marcado como borde
- Se considera el siguiente pixel vecino no explorado que está en la direccion perpendicular al gradiente y que sean mayor a U_{min} y se lo marca como borde
- A partir de dicho punto seguir las cadenas de máximos locales conectados en ambas direcciones perpendiculares a la normal del borde siempre que sean mayores a U_{min}. Así se marcan todos los puntos explorados y se almacena la lista de todos los puntos en el contorno conectado. Es así como en este paso se logra eliminar las

Por último: Cerrar los contornos

Este paso consiste en cerrar los contornos que pudiesen haber quedado abiertos por problemas de ruido.
Un método muy utilizado es el algoritmo de Deriche y Cocquerez. Este algoritmo utiliza como entrada una imagen binarizada de contornos de un píxel de ancho.
El algoritmo busca los extremos de los contornos abiertos y sigue la dirección del máximo gradiente hasta cerrarlos con otro extremo abierto.