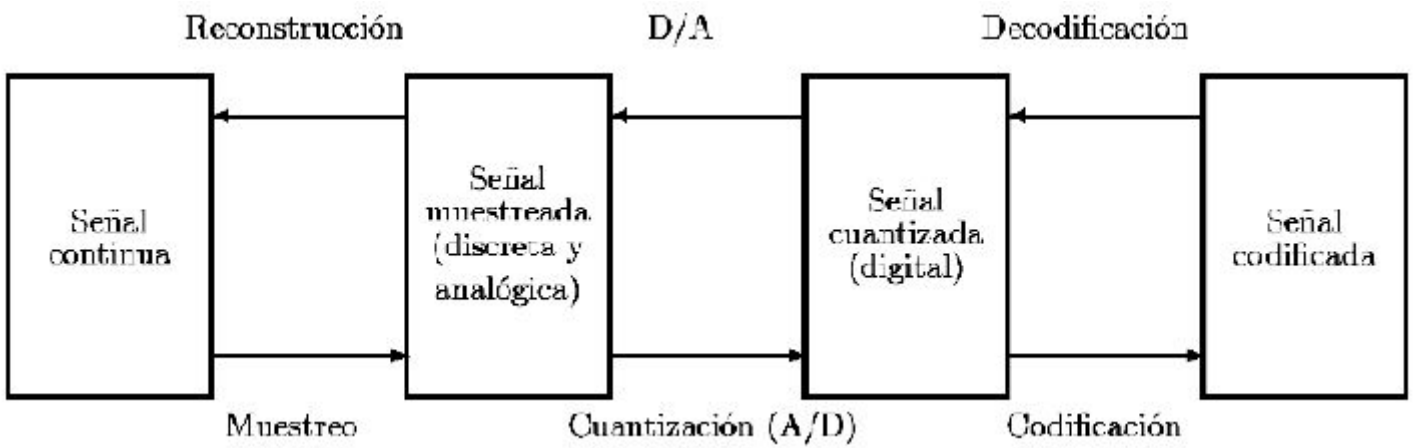
# Procesamiento Digital de Imágenes

Procesamiento por convolución.

Noelia Revollo
Facultad de Ingeniería– UNJU – CONICET
grevollo@fi.unju.edu.ar

# PDI – Señales (imágenes) discretas

Si recordamos el pasaje de las señales continuas a discretas, el primer paso consistía en el muestreo de la señal para transformarla en una señal discreta, es decir, que asume valores solo en determinados instantes. vecinos.



# PDI – Señales

El **procesamiento de señales digitales** es la manipulación matemática de una señal de información para modificarla o mejorarla en algún sentido. Este está caracterizado por la representación en el dominio del tiempo discreto, en el dominio frecuencia discreta, u otro dominio discreto de señales por medio de una secuencia de números o símbolos y el procesado de esas señales.

Esto se puede conseguir mediante un sistema basado en un procesador o microprocesador, tiene un *hardware* y un *software*

Debido a esto es especialmente útil para el procesado y representación de señales analógicas en tiempo real: en un sistema que trabaje de esta forma (tiempo real) se reciben muestras, normalmente provenientes de un conversor analógico/digital y digital/analógico

# PDI – Señales (imágenes) discretas

En el caso de las imágenes, los procesos de muestreo y cuantización se realiza en el sensor de la cámara, el cual integra el valor de la luz en una determinada área y lo transforma en un valor digital.

El arreglo del sensor típicamente sigue el esquema de Bayer que aquí se muestra, y los canales que faltan en cada lugar se obtienen promediando los vecinos.

https://es.wikipedia.org/wiki/Mosaico_de_Bayer

# PDI – Señales (imágenes) discretas

Varios aspectos esenciales de las señales (imágenes) discretas requiere un fundamento matemático bastante elaborado. La hoja de ruta pasa por los siguientes destinos esenciales:

*Modelo matemático de una señal discreta
*Modelo frecuencial de señales
*Pasaje del modelo tradicional (“temporal”) al frecuencial por medio de la Transformada de Fourier
*Transformada de Fourier discreta

# PDI – Filtrado espacial -Procesamiento por convolución

El filtrado espacial se utiliza en un amplio espectro de aplicaciones de procesamiento de imágenes.

El término filtrado **se toma prestado del procesamiento en el dominio de la frecuencia**, donde **"filtrar" se refiere a pasar, modificar, o rechazar** componentes de **frecuencia** específicos de una imagen.

Por ejemplo, un filtro que deja pasar frecuencias bajas se llama **filtro de paso bajo**. El efecto neto producido por un filtro de paso bajo es *suavizar una imagen desenfocándola.*

# PDI – Filtrado espacial -Procesamiento por convolución

El filtrado espacial *modifica una imagen reemplazando el valor de cada píxel por una función de los valores del píxel y sus vecinos.*

Si la **operación** realizada en los píxeles de la imagen es **lineal**, entonces el filtro se denomina **filtro espacial lineal**. De lo contrario, el filtro es espacial no lineal.

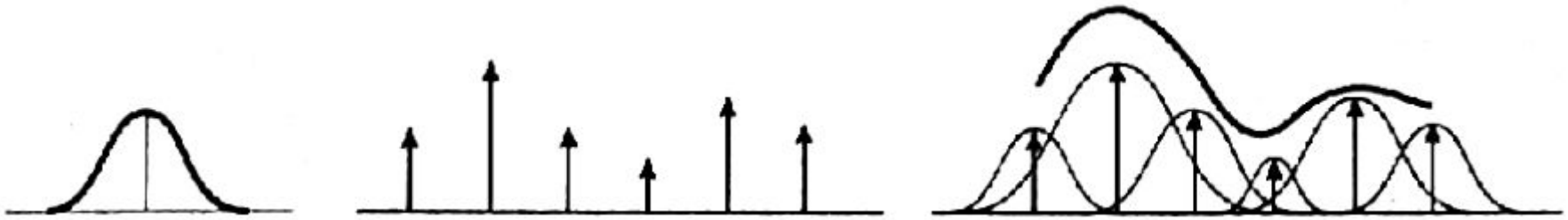
A su vez, los filtros lineales pueden clasificarse según las frecuencias que dejen pasar: los filtros **paso bajo**; los filtros **paso alto**; los filtros **pasa banda**.

# PDI – Convolución

Este caso puede ser ilustrado en la figura de abajo: **f** es la función de la izquierda (continua y acotada) y **g** es la función del centro (discreta).

En este caso su **convolución será la sumatoria que se obtiene al trasladar y escalar f en cada uno de los lugares donde g ocurre.**

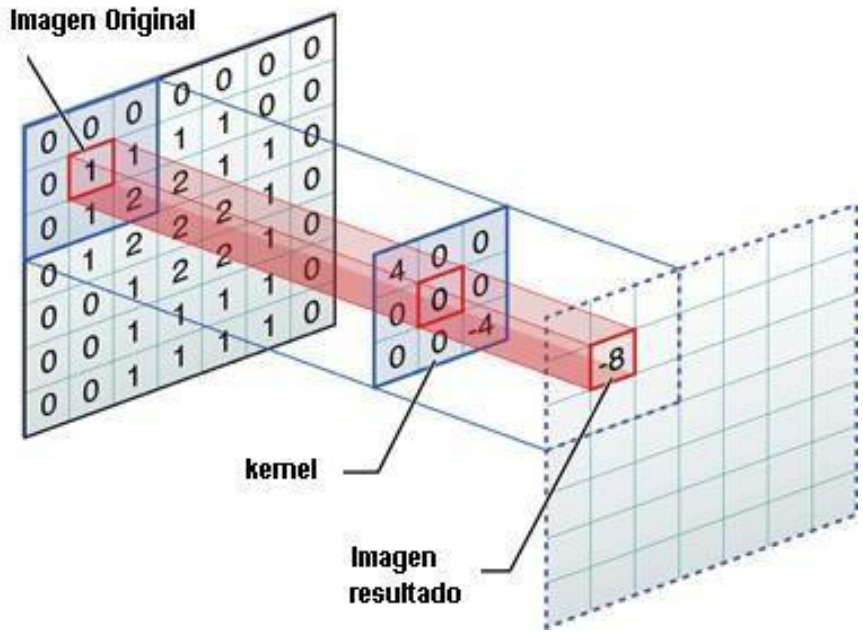
En https://es.wikipedia.org/wiki/Convolución pueden encontrar algunas animaciones bastante ilustrativas.



# PDI – Procesamiento por convolución

La convolución entre imágenes (o una imagen y un pequeño *kernel*) puede entenderse con esta imagen

1 Para cada pixel (i, j) de la imagen original depositamos el centro del *kernel* en dicho pixel.
2 Se multiplica cada pixel de la imagen original por cada celda correlativa del *kernel*.
3 Dichos productos se suman, y el valor final es el pixel correlativo (i, j) en la imagen resultado.



# PDI – Procesamiento por convolución

Si bien hay casos excepcionales, en la gran mayoría de los casos los **kernel** que se utilizan son **cuadrados y de tamaño impar,** y por lo tanto se **“insertan” en su celda central.**

El procesamiento por convolución, por lo tanto, desde el punto de vista computacional es relativamente sencillo: se **implementa** por medio de un doble ciclo for doble: recorremos la imagen original por (i, j), y para cada pixel (i, j) recorremos el kernel por fila y columna para calcular la **suma de productos.**

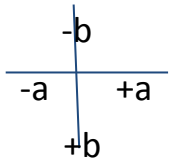
Es, por lo tanto, menos complejo y más sencillo que el procesamiento espectral. Al igual que en dicho caso, **veremos aquí convolución de imágenes en niveles de gris**, aunque es sencillo llevar los conceptos a YIQ o RGB.

# PDI

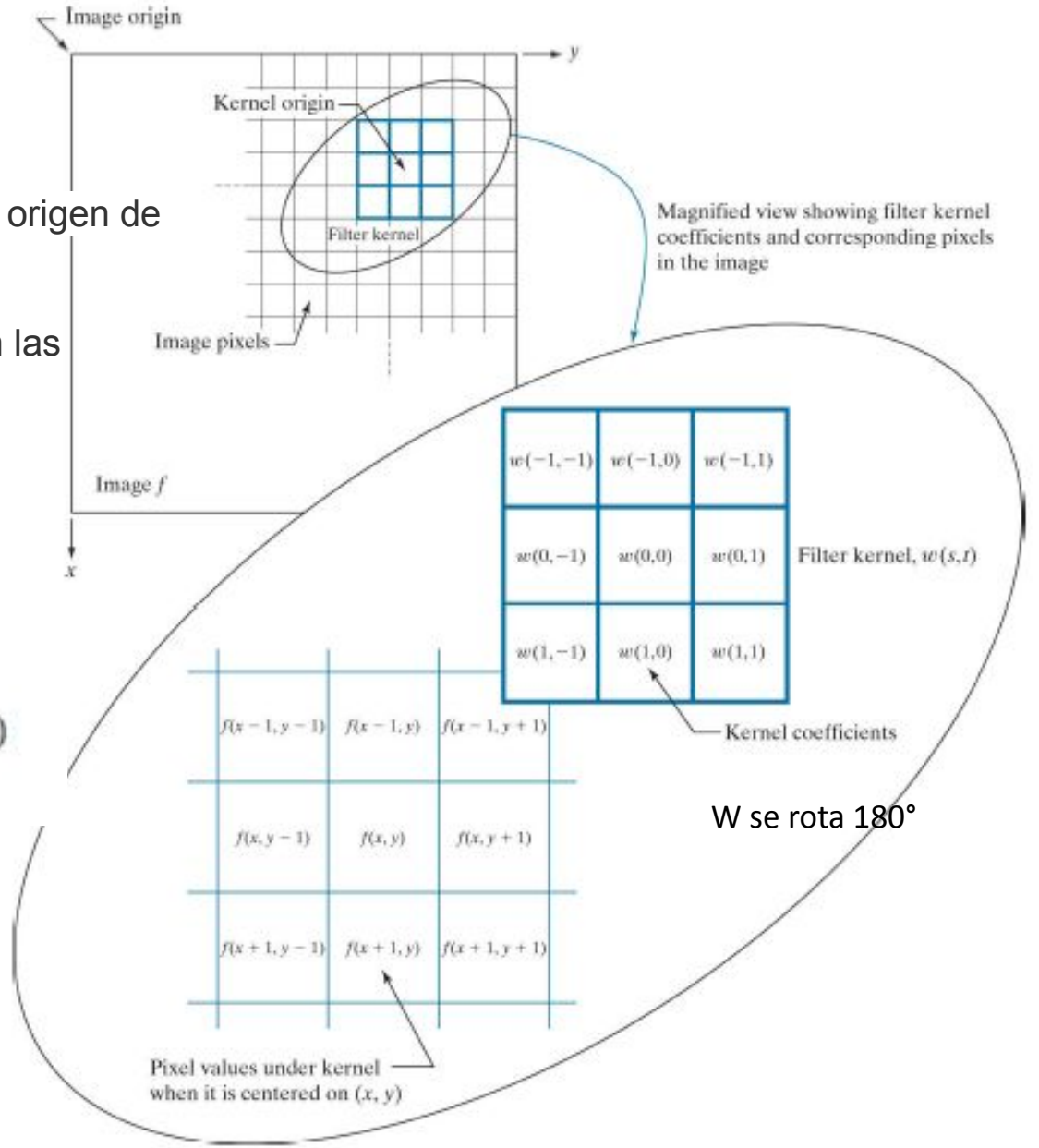
Kernel **w** son definidos de tamaño *impar*, debido a que el origen de estas matrices se encuentra al *centro*.

Para determinar el tamaño **(m×n)** de los filtros, se utilizan las formulas: ***m=2a+1***

***n=2b+1***




$$(w \star f)(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x - s, y - t)$$

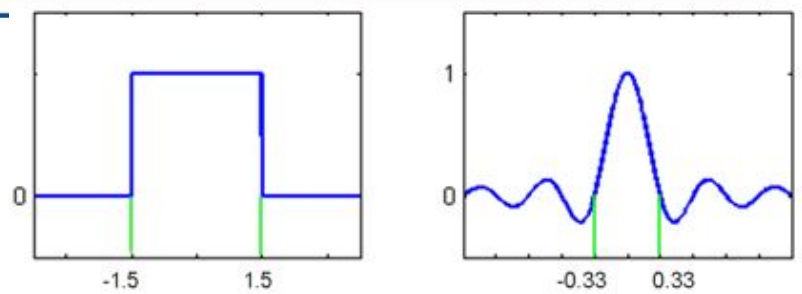



W se rota 180°

# PDI – Procesamiento por convolución

Tenemos un pulso de ancho 3.

La teoría predice que la TF de una sinc de ancho 3 tendrá los cruces por cero a frecuencias 1/3.




Es decir, nuestro filtrado dejará pasar mayoritariamente las frecuencias menores a 1/3 y filtrará mayoritariamente las mayores. Frecuencia 1/3 representa una oscilación cada tres pixels. Es decir, el resultado será un flitro *pasa bajo*.

Y esto es así con nuestro kernel con todos unos, salvo por un pequeño detalle.

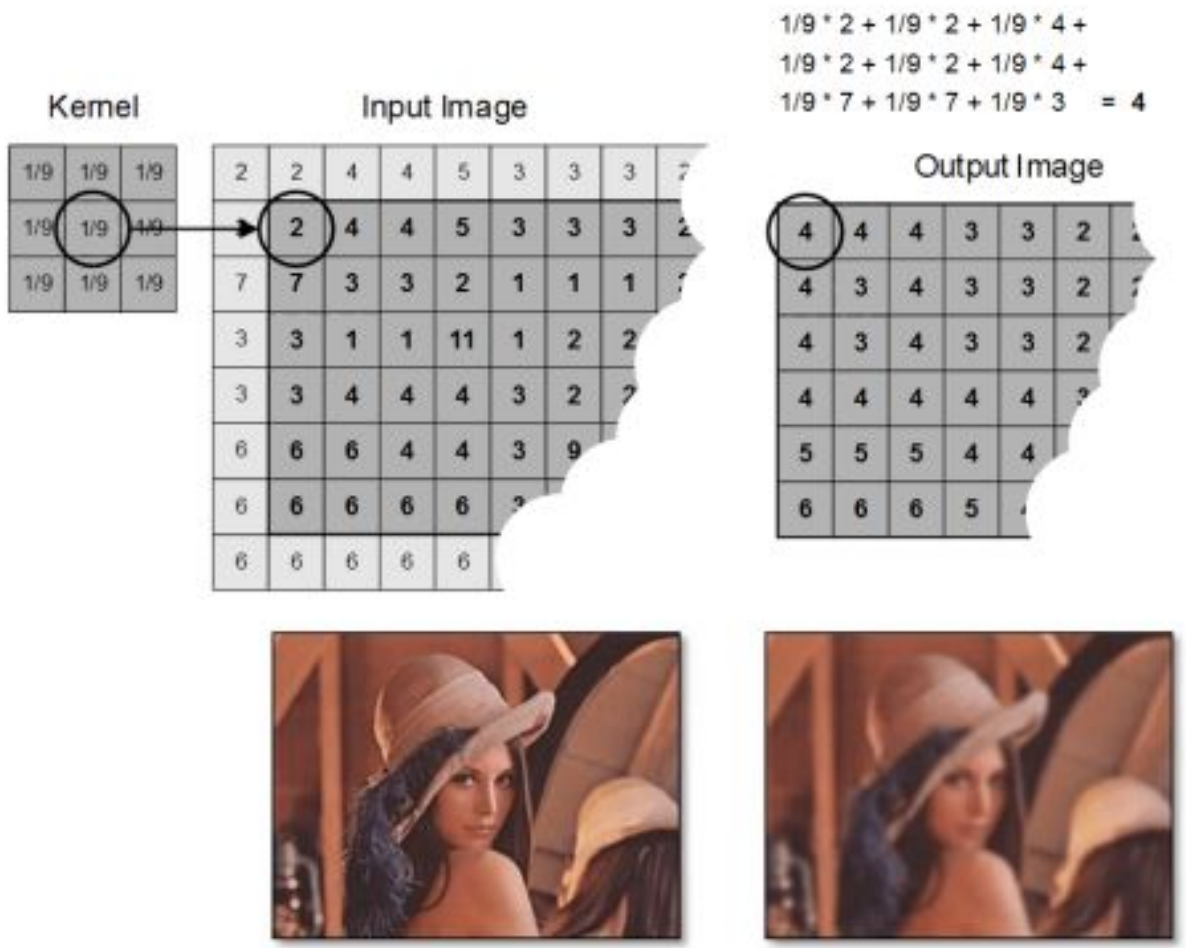
# PDI – Procesamiento por convolución

El problema que tenemos ahora es que un kernel con todos unos actualmente multipli-cará la luminancia por tantos valores como tenga el kernel. En nuestro caso, si un pixel tiene un valor de luminancia 1 y así ocurre con sus vecinos, el resultado del procesa-miento por convolución producirá en la imagen resultado un pixel de luminancia 9.

Si deseamos que las luminancias no se multipliquen, tenemos que ponderar los valores dentro del kernel por un factor inversamente proporcional a la cantidad de celdas del kernel. En nuestro caso, el kernel tendrá todos valores 1/9.

Ver que de esa manera, el filtrado que estamos haciendo es el *promedio* de un pixel con sus vecinos.

# PDI – Procesamiento por convolución



# PDI – Procesamiento por convolución

Vemos en otra aplicación de referencia el efecto de aplicar este procesamiento sobre una imagen fotográfica. Se observa que el efecto se puede caracterizar como un borroneo o difuminado (blur). Los detalles finos se pierden, las aristas dejan de ser tan claras, etc.

# PDI – Procesamiento por convolución

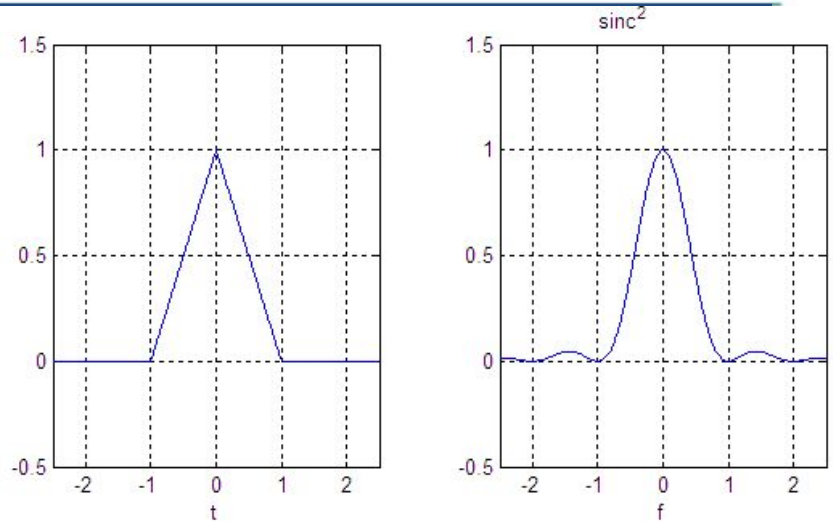
El objetivo de este tipo de operadores es ***suavizar*** las señales, esto es logrado al convolucionar la señal/imagen con una ventana que promediará el pixel de salida con su vecindad, también es conocido como ***filtro pasa bajos*.**

Entre las ventajas encontramos que logra ***reducir el ruido***, sin embargo, citando al tío Ben: "***un gran poder conlleva una gran responsabilidad***" y uno de los problemas es que también se da una *perdida de detalle*.

# PDI – Procesamiento por convolución

Si por ejemplo convolucionamos el pulso consigo mismo, el resultado es una función lineal a trozos o función triangular (recordar la animación en el artículo de wikipedia:

https://en.wikipedia.org/wiki/Convolution#/media/File:Convolution_of_box_signal_with_itself2.gif ).



# PDI – Pasabajos Bartlett

La forma más sencilla de implementar filtros de Bartlett consiste en armar una secuencia creciente hasta el centro y luego decreciente, y luego decreciente (1-2-1 para kernel de tamaño 3, 1-2-3-2-1 para kernel de tamaño 5, etc.) y luego armar el kernel multiplicando por fila y por columna dicha secuencia (se muestra el caso 3x3). Luego, para tener ganancia 1 como corresponde a un pasabajos, hay que dividir por la sumatoria de los coeficientes (en este caso 16).

| | | |
|---|---|---|
| 1 | 2 | 1 |
| 2 | 4 | 2 |
| 1 | 2 | 1 |

Luego, el kernel Bartlett 3x3 queda 1/16, 2/16, 1/16, etc.

# PDI – Filtro Gaussiano

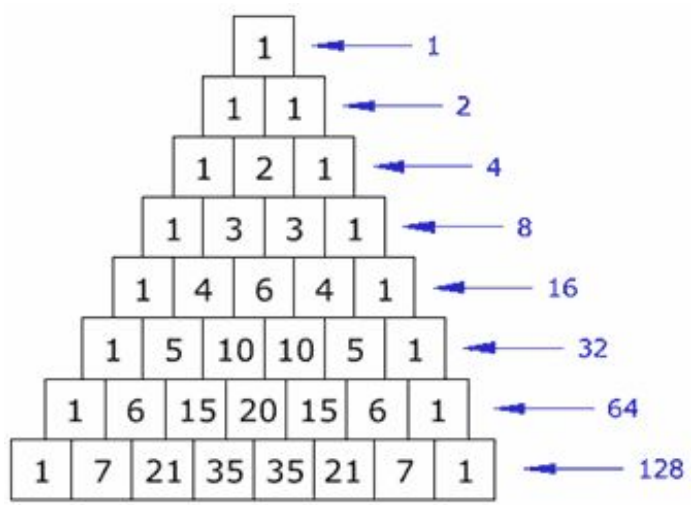
Por qué detenernos en una sola convolución del pulso consigo mismo? Por qué no hacerlo 10, 20 veces e ir mejorando cada vez más la calidad del filtro pasabajos?

Un corolario del teorema del límite central es que la convolución repetida del pulso consigo mismo tiende a la función Gaussiana. Por dicha razón, el kernel Gaussiano es el estándar de facto en filtrado pasabajos.

Una manera práctica de implementar kernels Gaussianos es aprovechar la relación que tiene el “triángulo de Pascal” con la función Gaussiana.

# PDI – Filtro Gaussiano

En la figura vemos algunas filas de dicho triángulo y la suma total de sus coeficientes (ya veremos por qué).




Si vemos las filas con cantidad impar de elementos, vemos que la tercera coincide con Bartlett, así que veamos la quinta: 1-4-6-4-1.

Aplicando la misma idea que con Bartlett, armamos una matriz multiplicando esos coeficientes por fila y por columna.

# PDI – Filtro Gaussiano

Tenemos luego que normalizar para obtener ganancia 1, por lo que hay que dividir dichos coeficientes por su sumatoria. Como la suma en la fila era igual a 16, y la matriz tiene el producto de dos de dichos vectores, el coeficiente de normalización es 256.

$$\frac{1}{256}\cdot\begin{bmatrix}1 & 4 & 6 & 4 & 1\\ 4 & 16 & 24 & 16 & 4\\ 6 & 24 & 36 & 24 & 6\\ 4 & 16 & 24 & 16 & 4\\ 1 & 4 & 6 & 4 & 1\end{bmatrix} = \frac{1}{256}\cdot\begin{bmatrix}1\\4\\6\\4\\1\end{bmatrix}\cdot\begin{bmatrix}1 & 4 & 6 & 4 & 1\end{bmatrix}$$

Variantes de este filtro son utilizadas multitudinariamente por todas las aplicaciones de procesamiento de imágenes y video.

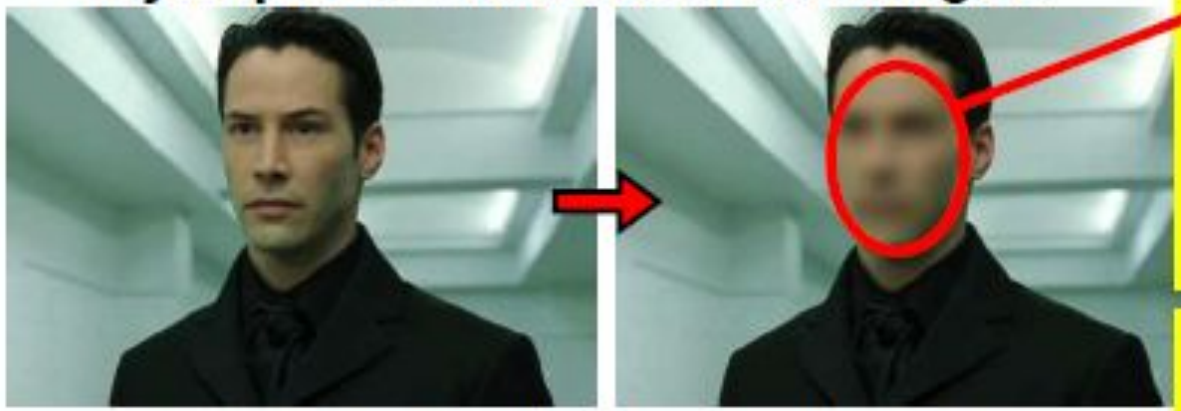
# PDI – Filtro Gaussiano

También para comparar, vemos en esta imagen el resultado de aplicar nuestro filtro Gaussiano a la imagen fotográfica. Si bien a través de este documento es difícil que se aprecien las sutiles diferencias con los primeros filtrados, el hecho es que si bien la imagen filtrada no posee alta frecuencia, igual sigue siendo mucho más inteligible y menos deteriorada que con el filtrado promedio o el Bartlett.

Ejem

# PDI – Ejemplos

- **Ejemplo 1.** Protección de testigos.



Se aplica un suavizado pero sólo en cierta región de interés (**ROI**), en este caso elíptica.

¿Cómo encontrar la posición de la cara automáticamente?

- **Ejemplo 2.** Resaltar objetos de interés.

Se suaviza el fondo para destacar al personaje, simulando un desenfoque.

# PDI – Filtros Pasa-altos y detección de bordes

Si existe un pasabajos, entonces seguramente existirá y será útil el pasa*altos*. Este será un filtro que eliminará en la imagen toda la energía excepto aquella que esté en frecuencias altas (es decir, cambios de luminancia muy locales).

Como sabemos, la mayor parte de la energía de las imágenes está en la zona baja del espectro, así que el filtrado pasaaltos retiene muy poca cantidad de energía.

El filtro más utilizado para estos fines es el Laplaciano.

https://en.wikipedia.org/wiki/Discrete_Laplace_operator.

# PDI – Filtrado Laplaciano

El Laplaciano (discreto) en 1D queda expresada como la diferencia finita entre dos diferencias finitas (o también, *segundas diferencias*). En 2D queda según esta ecuación:

$$\Delta f(x,y) \approx \frac{f(x-h,y) + f(x+h,y) + f(x,y-h) + f(x,y+h) - 4f(x,y)}{h^2}$$

En pocas palabras, utilizamos el pixel central y sus 4 vecinos por fila y columna, o bien sus 8 vecinos. Veamos que la ganancia de un pasaaltos debe ser cero, por lo que la suma de todos los coeficientes del kernel debe ser cero.

| | | |
|---|---|---|
| 0 | −1 | 0 |
| −1 | 4 | −1 |
| 0 | −1 | 0 |

| | | |
|---|---|---|
| −1 | −1 | −1 |
| −1 | 8 | −1 |
| −1 | −1 | −1 |

# PDI – Filtrado Laplaciano

Uno de los problemas con los que nos enfrentamos al implementar el Laplaciano es que en el pixel resultado podemos toparnos con valores mayores que 1 o menores que 0, al aplicar Laplaciano a un pixel de luminancia alta rodeado de pixels de luminancia baja o a la inversa, respectivamente.

Para ello tenemos que recurrir a nuestras funciones de coerción que vimos en el segundo tema de la primera clase, es decir, luego de aplicar la suma de productos, si la luminancia es menor que cero, se coerciona a cero, o si es mayor que uno se coerciona a uno.

# PDI – Filtrado Laplaciano

Volvemos a nuestra anterior imagen fotográfica y vemos que el resultado de aplicar el Laplaciano (en este caso con 4 vecinos) selecciona la parte de la imagen donde se concentra el mayor detalle que era borroneado por el filtro pasabajos. Estos detalles están en general en las aristas o bordes de los objetos.