VISIÓN POR COMPUTADOR:

PRACTICA 1:
Filtrado y Muestreo

Jesús Moyano Doña - 76626194-S
jesusmoyano97@correo.ugr.es
Grupo: 4ºA

Ejercicio 1

1.- USANDO LAS FUNCIONES DE OPENCV : escribir funciones que implementen los siguientes puntos: (2 puntos)

Apartado A

El cálculo de la convolución de una imagen con una máscara Gaussiana 2D (Usar GaussianBlur). Mostrar ejemplos con distintos tamaños de máscara y valores de sigma. Valorar los resultados.

Para calcular la convolución de la imagen he aplicado a la misma un GausianBlur con un tamaño de la mascara de 5 y he ido variando los sigmas para ver como cuanto mayor sea esté parámetro mas se acentúa el filtro. Esto se debe a que cuanto más grande sea el valor del sigma mayor sera la campana de la distribución normal de la que depende el filtro, haciendo así que los pixeles usados para generar la gaussiana serán más distantes. Si aumentamos el sigma, también debemos aumentar el tamaño del kernel para que pueda capturar esos pixeles.
La primera imagen es la original y las siguientes toman los valores de sigma 3, 6 y 9 consecutivamente.

Apartado B

Usar getDerivKernels para obtener las máscaras 1D que permiten calcular al convolución 2D con máscaras de derivadas. Representar e interpretar dichas máscaras 1D para distintos valores de sigma.

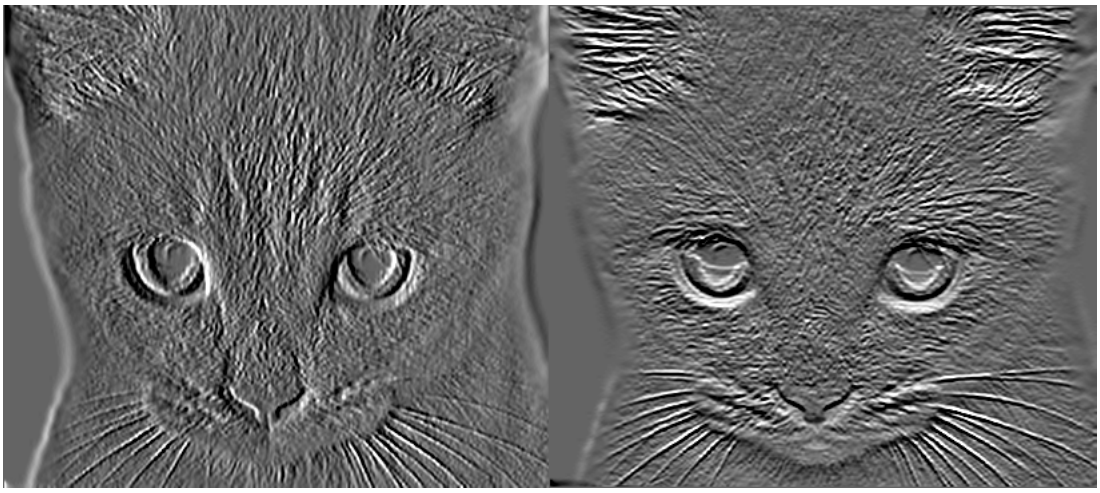
La función getDerivKernels(dx,dy,ksize), nos proporciona los kernels de las mascaras derivadas, podemos establecer que derivada queremos mediante los parámetros dx,dy (los cuales se identifican con la derivada en X y en Y), la mascara sera del tamaño establecido en el parámetro ksize.
En el caso de obtener la derivada de los kernels de y solamente la matriz que obtenemos es la siguiente:

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

Interpretando esa matriz nos damos cuenta de que al derivar en Y, lo que vamos a aplicar el filtro a los ejes horizontales.
En el caso contrario, al derivar sobre X, lo que remarcará son los ejes verticales. Esta es la matriz de los kernels de la derivada de X:

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

En la foto de la izquierda la derivada es en X, en caso contrario en la foto de la derecha, la derivada es en Y.



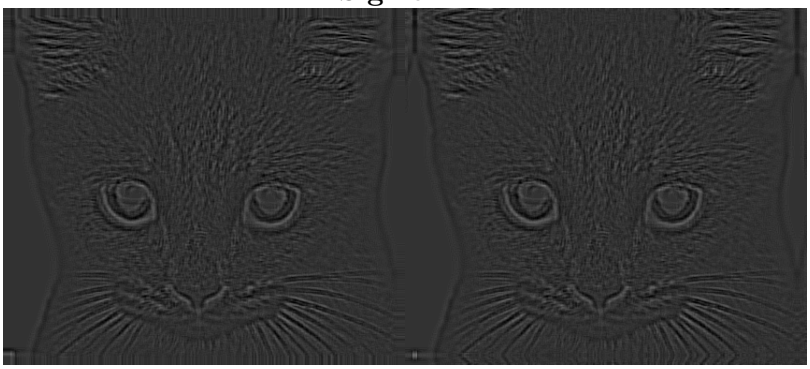
Se observan diferencias claramente en las orejas, donde en la derivada de y se ven mejor ya que las rayas son horizontales, mientras que en la de X son más lisas. Las marcas verticales de la frente se ven mejor en la derivada de X, por el contrario en la imagen derivada de Y no se aprecian.

Apartado C

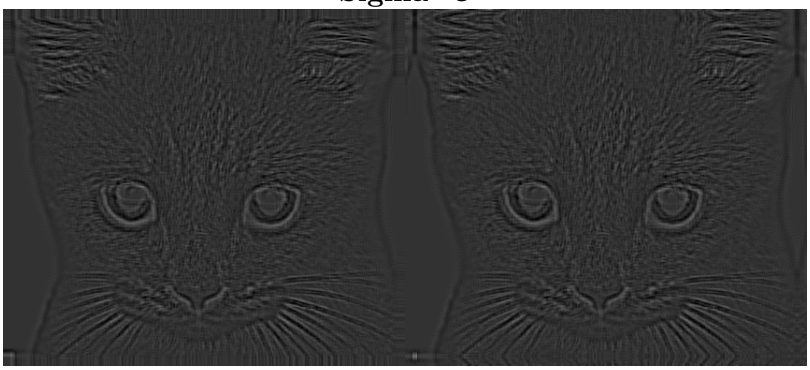
Usar la función Laplacian para el cálculo de la convolución 2D con una máscara de Laplaciana-de-Gaussiana de tamaño variable. Mostrar ejemplos de funcionamiento usando dos tipos de bordes y dos valores de sigma: 1 y 3.

En este apartado, lo primero ha sido añadir bordes a las imágenes con la función copyMakeBorder(). A continuación he realizado el blur a la imagen tanto con sigma = 1 como con sigma = 3 y con una mascara (5,5). Para finalizar a cada imagen le aplicamos la función Laplacian. El filtro laplaciano, es un filtro derivado, nos sirve para identificar las arenas con cambios grandes, por asi decirlo los bordes.
Esta función tiene un parámetro Delta que lo que hace es sumar a la imagen el valor que pasas por parámetro, he usado un delta de 50 para ayudar en la visualización de la imagen.

Sigma=1



Sigma=3



Ejercicio 2

IMPLEMENTAR apoyándose en las funciones getDerivKernels, getGaussianKernel, pyrUp(), pyrDown(), escribir funciones los siguientes (3 puntos)

Apartado A

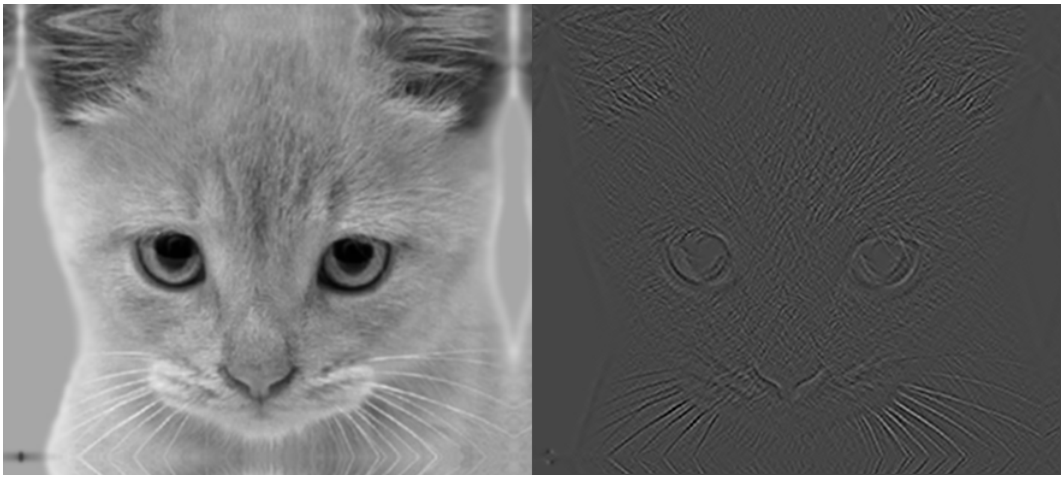
El cálculo de la convolución 2D con una máscara separable de tamaño variable. Usar bordes reflejados. Mostrar resultados

Para este ejercicio he implementado una función que recibe una imagen y dos kernels. A la imagen le añade un borde de reflejo y le aplica un filtrado con esos kernels. Esta función la usaremos en los siguientes apartados de este ejercicio.

Apartado B

El cálculo de la convolución 2D con una máscara 2D de 1a derivada de tamaño variable. Mostrar ejemplos de funcionamiento usando bordes a cero.

Se pide el calculo de la convolución 2D de una mascara con la 1ª derivada, por lo que convolucionamos la imagen con la misma función usada en el ejercicio 1A, que aplica un GausianBlur a la imagen y terminamos aplicando el apartado 2A con los kernels de la primera derivada. La primera derivada nos da como resultado una imagen donde se ven los bordes de la imagen coloreados en blanco, el resto en gris es donde no existen cambios.



Apartado C

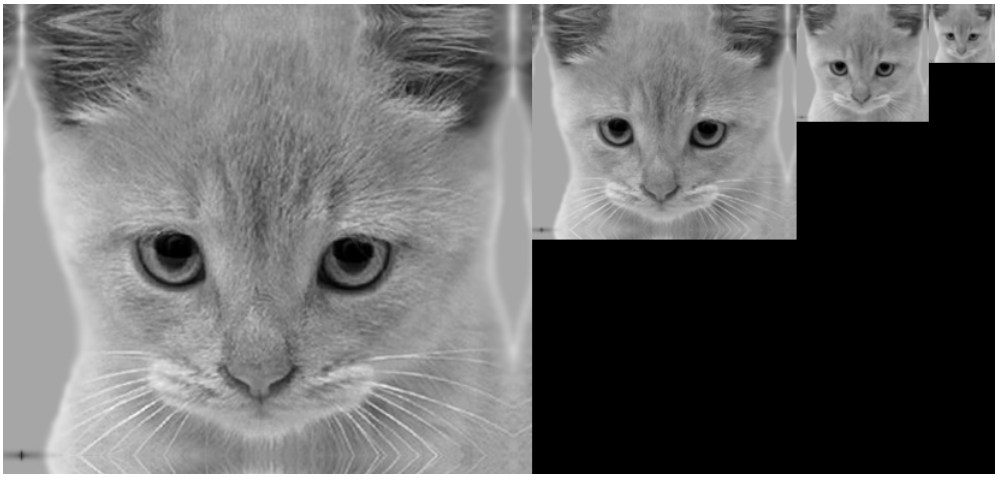
El cálculo de la convolución 2D con una máscara 2D de 2a derivada de tamaño variable.

Este ejercicio se realiza de la misma forma que el apartado B, la diferencia es que la mascara se le hace la segunda derivada. La segunda derivada marca los bordes que más cambio tienen.

Apartado D

Una función que genere una representación en pirámide Gaussiana de 4 niveles de una imagen. Mostrar ejemplos de funcionamiento usando bordes

Para generar la pirámide Gaussiana de 4 niveles, se ha usado la función pyrDown(). Esta función lo que hace es ir disminuyendo el tamaño de la imagen, por lo que, a medida que se disminuya se irán juntando los pixeles, "se ira perdiendo información", dando la sensación de que se esta aplicando un filtro gaussiano. La piramide de resultado es la siguiente:



Y estas son las imágenes de la piramide escaladas a las misma dimensión que la original para observar mejor el filtro.

Apartado E

Una función que genere una representación en pirámide Laplaciana de 4 niveles de una imagen. Mostrar ejemplos de funcionamiento usando bordes.

Para generar la pirámide Laplaciana, partimos de la ultima imagen de la pirámide Gaussiana, ya que necesitamos una imagen con filtro gaussiano. Lo siguiente sería usar la función pyrUp() sobre la imagen para obtener una imagen de mayor tamaño y un filtro gaussiano mas grande. La Laplaciana de cada imagen la generamos restando a la imagen original su gaussiana. Este proceso lo hacemos 3 veces para obtener una piramide Laplaciana.
A cada imagen le he sumado una variable delta de valor 40, para ayudar a la visualización.

Ejercicio 3

Imágenes Híbridas: (SIGGRAPH 2006 paper by Oliva, Torralba, and Schyns). (3 puntos)

Para crear una imagen híbrida lo que necesitamos es mezclar dos imágenes tratadas de la siguiente forma. A una debemos obtenerle las frecuencias bajas, que será que esté de fondo (se visualizará cuando se haga pequeña) y a otra obtenerle las frecuencias altas, para que predomine cuando el tamaño sea mayor.

Las frecuencias bajas las obtendremos aplicándole un filtro gaussiano a una imagen. Para las altas lo que debemos hacer es una aproximación de la laplaciana, que se hace aplicando a una imagen un filtro gaussiano y luego restándoselo a la original. Es una forma de obtener una aproximación de un filtro laplaciano.

A continuación se muestran varias imágenes en las que observamos que a mayor tamaño más se marcan las imágenes con frecuencias altas, sin embargo cuando disminuimos, se pierden.