VC: Trabajo 1

Gustavo Rivas Gervilla

1. ¿Cuáles son los objetivos principales de las técnicas de visión por computador? Poner algún ejemplo si lo necesita.

Como se dice en *Szel* y lo que diferencia a la Visión por Computador del procesado digital de imágenes es que lo que pretende la Visión por Computador es reconstruir una escena 3D y comprenderla completamente, extrayendo toda la información.

Entonces podríamos decir que el objetivo principal de la visión por computador es comprender una escena emulando lo mejor posible los procesos de la visión humana, algo muy difícil.

Por otro lado las aplicaciones más típica de la Visión por computador son:

1. Reconocimiento: de objetos, identificación de personas...
2. Análisis de movimiento: obtener información del movimiento a partir de una secuencia de imágenes.
3. Reconstrucción de escenas. Obtener un modelo 3D a partir de imágenes de una escena desde distintos puntos de vista.
4. Restauración de imágenes.

`https://en.wikipedia.org/wiki/Computer_vision`

2. ¿Una máscara de convolución para imágenes debe ser siempre una matriz 2D? ¿Tiene sentido considerar máscaras definidas a partir de matrices de varios canales como p.e. el tipo de OpenCV CV_8UC3? Discutir y justificar la respuesta.

No, de hecho hemos visto cómo hay núcleos de convolución separables en dos núcleos 1D, uno para las filas y otro para las columnas, con lo cual podemos tener una máscara 1D para convolucionar (p.e. en el caso de suavizado Gaussiano nos basta con una sola para las filas y las columnas al ser simétrica) y pasarla primero por las filas y después por las columnas de la imagen. De hecho hemos visto que esto tiene ventajas desde el punto de vista de la complejidad computacional.

Es claro que nosotros podemos definir una máscara de convolución cómo queramos, p.e. obtener una máscara CV_32F haciendo operaciones sobre los canales de una CV_8UC3 (teniendo en cuenta que sumen 1 los elemento para hacer un buen alisamiento si es lo que deseamos). Por otro lado nosotros podemos usar cualquier matriz como máscara de convolución, siempre y cuando lo que hamos sea eso, una convolución (y los tipos sean compatibles para las operaciones necesarias). Pero sí es posible usar una máscara de 3 canales, podemos hacer una convolución distinta en cada uno de los canales de la imagen (con cada una de las máscara 1C que contendría nuestra máscara de tres canales), obteniendo distintos efectos (por ejemplo usar un alisamiento más acusado en el canal de rojos eliminando las altas frecuencias de este canal).

3. Expresar y justificar las diferencias y semejanzas entre correlación y convolución. Justificar la respuesta.

La primera semejanza es que ambas siguen un mecanismo similar, tomamos una máscara y a la vamos pasando por cada píxel de la imagen, operando sobre una región del mismo tamaño que la máscara y almacenando el resultado en el píxel central de dicha región (suponemos las máscaras de orden impar).

Una de las diferencias es que para realizar la convolución se realiza un paso previo a la correlación y es voltear la máscara en sus dos direcciones (izquierda a derecha y arriba a abajo).

Podemos encontrar otras diferencias y semenjanzas en *Szel*:

Es que si nosotros convolucionamos una máscara con una señal que vale cero en todo los puntos excepto en el origen el resultado es la misma señal, en cambio si lo que aplicamos es correlación entonces puede comprobarse que lo que obtenemos es la señal reflejada.

Además la convolución sabemos que tiene algunas propiedades adicionales como son la conmutatividad o la asociatividad. Y la transformada de Fourier de dos imágenes convolucionadas es la convolución de la transformada de cada imágen.

Pero también hay algunas semejanzas:

Ambos operadores siguen el principio de superposición, estos es:

$$h \circ (f + g) = h \circ f + h \circ g$$

además de que ambos operadores se comportan igual en cualquier sitios, lo que formalmente se expresa como sigue:

$$g(i, j) = f(i + k, j + l) \Leftrightarrow (h \circ g)(i, j) = (h \circ f)(i + k, j + l)$$

Y otra diferencia es el uso que se le suele dar a cada una de estas operaciones: la correlación (cruzada) nos da una medida de la similitud entre dos señales, mientras que la convolución se usa para ver la respuesta de un sistema ante un impulso y una entrada dada.

`https://es.wikipedia.org/wiki/Correlaci%C3%B3n_cruzada`

`http://dsp.stackexchange.com/questions/12684/difference-between-correlation-and-convolution-on-an-image`

`http://www.researchgate.net/post/Difference_between_convolution_and_correlation`

4. ¿Los filtros de convolución definen funciones lineales sobre las imágenes? ¿y los de mediana? Justificar la respuesta.

Por teoría sabemos que la convolución tiene las propiedades de superposición y de sacar fuera los escalares, con lo que efectivamente es una aplicación lineal.

Para ver que los filtros de mediana no son lineales basta con encontrar un contraejemplo para dos matrices (imágenes) dadas:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 6 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

$$B = \begin{bmatrix} ? & ? & ? \\ ? & 5 & ? \\ ? & ? & ? \end{bmatrix}$$

B es el resultado de aplicar a cada matriz la mediana y luego sumarlas, y como vemos en el pixel central estaría el valor 5. Por otro lado al aplicar el filtro de mediana sobre el píxel central de A, el valor obtenido sería un 6. Con lo cual hemos demostrado que los filtros de mediana no son lineales.

5. ¿La aplicación de un filtro de alisamiento debe ser una operación local o global sobre la imagen? Justificar la respuesta.

Debe de ser local ya que lo que se pretende es sustituir en cada píxel un cierto valor en base a los píxeles que lo rodean, con una máscara más o menos grande. De este modo se consigue el efecto deseado. En cambio si fuese una operación global se le haría a todos los píxeles lo mismo sin tener en cuenta su entorno, no obteniendo un alisamiento.

6. Para implementar una función que calcule la imagen gradiente de una imagen dada pueden platearse dos alternativas:
a) Primero alisar la imagen y después calcular las derivadas sobre la imagen alisada.
b) Primero calcular las imágenes derivadas y después alisar dichas imágenes.
Discutir y elegir qué estrategia es la más adecuada, si alguna lo es. Justificar la decisión.

Como se recomienda en *Szel* lo mejor es la primera opción, puesto que la imagen puede tener mucho ruido con lo cual el gradiente que obtendríamos no sería del todo adecuado en caso de no alisarla.

Además computacionalmente es mejor puesto que, si por ejemplo vamos a usar el mismo filtro de alisamiento, podemos tener precargada la derivada de ese filtro y usarla varias veces, aprovechando la siguiente propiedad de la convolución:

$$D(f \bigstar h) = D(f) \bigstar h = f \bigstar D(h)$$

Con lo cual la a es mejor que la b, pero puede mejorarse computacionalmente (pues ahorramos una discretización, como dijimos en clase) al aplicar la propiedad anterior.

7. Verificar matemáticamente que las primeras derivadas (respecto de x e y) de la Gaussiana 2D se puede expresar como núcleos de convolución separables por filas y columnas. Interpretar el papel de dichos núcleos en el proceso de convolución.

Llamemos $h_\sigma(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$ entonces sus parciales, donde hemos agrupado las dos partes en las que descomponen, son:

$$\frac{\partial}{\partial x} h_\sigma = \begin{pmatrix} \frac{1}{2\pi\sigma^2} \frac{-x}{\sigma^2} e^{\frac{-x^2}{2\sigma^2}} \end{pmatrix} \begin{pmatrix} e^{\frac{-y^2}{2\sigma^2}} \end{pmatrix}$$
$$\frac{\partial}{\partial y} h_\sigma = \begin{pmatrix} \frac{1}{2\pi\sigma^2} \frac{-y}{\sigma^2} e^{\frac{-y^2}{2\sigma^2}} \end{pmatrix} \begin{pmatrix} e^{\frac{-x^2}{2\sigma^2}} \end{pmatrix}$$

Como sabemos estos núcleos, al emplear la propiedad de la convolución que hemos citado en la pregunta anterior, intervienen en el cálculo del vector gradiente. Cada uno de ellos nos da una de las componentes del vector gradiente (la correspondiente a con respecto qué variable estamos derivando), ya que aplica los mayores pesos a la derecha e izquierda del píxel o arriba y abajo (muy parecido a los filtros del tipo [-1,1] que se usan para aproximar la derivada de una imagen.

8. Verificar matemáticamente que la Laplaciana de la Gaussiana se puede implementar a partir de núcleos de convolución separables por filas y columnas. Interpretar el papel de dichos núcleos en el proceso de convolución.

Esto es claro una vez hemos demostrado que las primeras derivadas (respecto x e y) son separables. Veámoslos:

$$\frac{\partial^2}{\partial x^2} h_\sigma(x,y) = \frac{1}{2\pi\sigma^2} \left[\frac{-1}{\sigma^2} e^{\frac{-x^2}{2\sigma^2}} + \left(\frac{-x}{\sigma^2}\right)^2 e^{\frac{-x^2}{2\sigma^2}} \right] \begin{pmatrix} e^{\frac{-y^2}{2\sigma^2}} \end{pmatrix}$$
$$\frac{\partial^2}{\partial y^2} h_\sigma(x,y) = \frac{1}{2\pi\sigma^2} \left[\frac{-1}{\sigma^2} e^{\frac{-y^2}{2\sigma^2}} + \left(\frac{-y}{\sigma^2}\right)^2 e^{\frac{-y^2}{2\sigma^2}} \right] \begin{pmatrix} e^{\frac{-x^2}{2\sigma^2}} \end{pmatrix}$$

Con lo que sumando podemos ver la laplaciana como la suma de dos núcleos de convolución descomponibles cada uno de ellos en filas y columnas, que aprovechando la propiedad de superposición de la convolución podemos aplicar uno tras el otro.

Como sabemos la laplaciana se usa para detectar bordes, en concreto remarca zonas de máximo cambio (corte con el eje), cada uno de los núcleos sirve para distinguir cambios en cada una de las direcciones, y en conjunto detectamos dónde se ha producido el mayor salto, dónde puede haber un borde.

http://homepages.inf.ed.ac.uk/rbf/HIPR2/log.htm

9. ¿Cuáles son las operaciones básicas en la reducción del tamaño de una imagen? Justificar el papel de cada una de ellas.

Las dos operaciones básicas son el submuestreo y el alisamiento.

El papel del submuestreo es obvio, buscamos reducir el tamaño (resolución) de una imagen con lo que tenemos que reducir el número de píxeles que la forman. Este submuestreo lo podemos hacer cómo queramos, se trata de tomar un subconjunto de píxeles. Ahora bien, si queremos obtener un buen resultado tendremos que tomar un número de muestras adecuado, el número de muestras a tomar nos lo da el ratio de Nyquist.

Por otro lado, pese a tomar un número de muestra adecuado podemos seguir obteniendo imágenes con el indeseado efecto de aliasing. Esto se debe a que la imagen de partida tiene frecuencias muy altas y no es suficiente con tomar un número de muestras considerable. Aquí es donde juega su papel el alisamiento, eliminando estas altas frecuencias, reduciendo el indeseado efecto.

Con lo cual en primer lugar aplicaremos un filtro de paso bajo y a continuación realizaremos el submuestreo.

10. ¿Qué información de la imagen original se conserva cuando vamos subiendo niveles en una pirámide Gausssiana? Justificar la respuesta.

De un nivel a otro de la gaussiana realizamos un submuestreo de la imagen anterior suavizada, en consecuencia en cada píxel se almacena una media de los píxeles que tenía alrededor en la imagen previa. Además como lo que estamos haciendo es alisar lo que conservamos son las bajas frecuencias, las altas como sabemos se perderán.

11. ¿Cuál es la diferencia entre una pirámide Gaussiana y una Piramide Lapalaciana? ¿Qué nos aporta cada una de ellas? Justificar la respuesta. (Mirar en el artículo de Burt-Adelson)

La primera diferencia que observamos es lo que se almacena en cada una de ellas, la pirámide Gaussiana es una especie de paso previo para calcular la Laplaciana. Si g_0 es el primer nivel de la pirámide Gaussiana el siguiente nivel será g_0 suavizada (mediante un filtro gaussiano) y submuestreada, y así sucesivamente.

Por otro lado lo que almacenamos en la pirámdie Laplaciana es lo que se llama el error de predicción, así el primer nivel de esta piramide será $L_0 = g_0 - EXPAND(g_1)$ como podemos ver tenemos que hacer un aumento de tamaño de g_1 (que se realiza mediante interpolación) para que las dos matrices que se restan tengan el mismo tamaño. Y es evidente cómo se calculan los siguientes niveles de la pirámide en base a los sucesivos niveles de la pirámide Gaussiana asociada.

Con lo cual estas pirámides difieren en la información que almacenan con

respecto a la imagen original y evidentemente, en el proceso de cálculo.

Veamos ahora qué aportan cada una de ellas:

La pirámide Gaussiana nos ayuda a almacenar información sobre los valores predecidos para la imagen original a distintas escalas, de modo que podemos tener una imagen algo más pequeña y que ocupe menos que la original. Por otro lado en la Laplaciana se almacena el posible error que se da al pasar de un nivel a otro de la Gaussiana, de modo que podamos recuperar la original, con una representación mucho más comprimida.

En definitiva la Gaussiana nos da la base para elaborar la Laplaciana fácilmente, y esta nos ayuda a almacenar imágenes con un coste menor en memoria.

`http://persci.mit.edu/pub_pdfs/pyramid83.pdf`

12. Cual es la aportación del filtro de Canny al cálculo de fronteras frente a filtros como Sobel o Robert. Justificar detalladamente la respuesta.

Va más allá del Sobel y Robert al aplicar un filtro para eliminar ruido y además después hace el non-maximun supression para seleccionar de entre todos los píxeles candidatos a formar parte de un eje, los mejores.

Los filtros de Sobel y Robert son mucho más simples, simplemente aplican una máscara 3x3 y 2x2 (respec.) para calcular el gradiente. El filtro de Sobel es aplicado como un paso del filtro de Canny, el cual busca mejorar los filtros de detección de bordes previos en base a tres criterios básicamente:

1. Obtener una cota de error baja.
2. Que la distancia entre el eje detectado y el real fuese lo más pequeña posible.
3. Que sólo se produjese una respuesta por cada uno de los bordes reales.

Para conseguir tales objetivos se usa el alisamiento, además se realiza la supresión de píxeles por debajo de un determinado valor poniéndolos a cero (non-maximum supression). Y por último este filtro incorpora dos valores límites, el de eliminación que acabamos de comentar y el que sirve para determinar si un píxel puede formar parte de un borde: si está por encima del nivel máximo se marca directamente, si está entre este nivel y el otro entonces formará parte de un borde si está conectado con un píxel marcado previamente (histeresis).

Un estudio más detallado de la comparativa entre distintos filtros de detección de bordes puede verse en el siguiente artículo, de donde hemos extraido la información (junto con algunos enlaces visitados más):

`http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.301.927&rep=rep1&type=pdf`

https://en.wikipedia.org/wiki/Canny_edge_detector
https://en.wikipedia.org/wiki/Roberts_cross
https://en.wikipedia.org/wiki/Sobel_operator
https://www.quora.com/
What-is-the-difference-between-edge-detection-Sobel-detection-and-Canny-detection

13. Buscar e identificar una aplicación real en la que el filtro de Canny garantice unas fronteras que sean interpretables y por tanto sirvan para solucionar un problema de visión por computador. Justificar con todo detalle la bondad de la elección.

El cemento es un material muy heterogéneo y sus propiedades depende de cómo se distribuyan los distintos materiales (además de los huecos que suele presentar) que lo forman.

Lo que se suele usar a nivel indutrial para estudiar la calidad y propiedades del cemento son técnicas de visión por computador, entre ellas, después de un preprocesamiento de las imágenes tomadas para hacerlas más interpretables, se usa la detección de bordes. Como sabemos hay multitud de filtros para tal propósito, pero, por las propiedades que hemos comentado en la pregunta anterior el más utilizado es el de Canny.

Ahora bien este filtro también tiene sus defectos que pasamos a describir a continuación:

El filtro sigue siendo vulnerable a ruido y en consecuencia puede detectar algunos falsos bordes o perder información de bordes presentes en la imagen. Como sabemos este algoritmo compara píxeles adyacentes en la dirección del gradiente para saber si el píxel que se está procesando presenta un máximo local, lo que puede dar lugar a imprecisiones.

Por otro lado el algoritmo de Canny tampoco es capaz de detectar bordes ramificados. Además pueden producirse rugosidades que por tener valores muy bajos en los niveles máximos sean difíciles de detectar y en algunas ocasiones hay que controlar muy bien los niveles para realizar la histeresis puesto que pueden perderse bordes que han sido suavizados por la Gaussiana aplicada previamente.

Por todas estas razones en el siguiente artículo se presenta una mejora del fitro de Canny, como por ejemplo cambiar el filtro de alisamiento:

http://www.researchgate.net/publication/220753963_An_Improved_
Canny_Edge_Detection_Application_for_Asphalt_Concrete