Cuestionario de teoría
1

Alejandro García Montoro
agarciamontoro@correo.ugr.es

21 de octubre de 2015

1. Cuestiones

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

Solución. El objetivo último de la visión por computador es el de extraer *significado* de forma automatizada de las imágenes que recibe un ordenador. Qué es el *significado* y cómo hacemos esa automatización son preguntas cuyas respuestas intenta abordar la visión por computador.

Así, el tipo de significado —esto es, de información— que se intenta extraer es amplio y depende del uso que le demos a esta técnica; por ejemplo, podemos nombrar algunos tipos de información que se puede extraer de una imagen:

- Información semántica: determinar qué objetos aparecen en una imagen, qué papel juegan en la escena retratada, deducir sentimientos por las expresiones faciales...
- Información geométrica: determinar cómo está formada geométricamente la escena retratada, medir distancias, determinar la profundidad relativa de cada objeto que aparece, extraer la perspectiva...

La automatización de esta extracción de significado es la parte técnica de la visión por computador; requiere del desarrollo de modelos matemáticos, de algoritmos y del estudio de la ejecución de estos últimos para que sean viables.

Cuestión 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.*

Solución. La definición de operador de convolución que se ha dado en teoría no sólo restringe las máscaras a ser de dos dimensiones sino a que sean cuadradas,

$$\star_{H,F} : I \times J \longrightarrow \mathbb{R}$$

$$(i,j) \longmapsto (H \star F)(i,j) = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H(u,v)F(i-u, j-v)$$

pues los índices de las sumatorias van de $-k$ a k, donde k es el lado en píxeles de la máscara.

Sin embargo, matemáticamente, y también en el ámbito de la visión por computador, las máscaras de convolución pueden ser, por ejemplo, de una sola dimensión. Esto además puede encajar en la definición vista en teoría si la máscara que conceptualmente es de una dimensión la vemos como de dos dimensiones asignando un peso nulo a los píxeles que no nos interesan.

De hecho, cuando las máscaras de convolución son separables, como la Gaussiana, la convolución se hace en dos pasos: el primero con una máscara de una dimensión horizontal y el segundo con una máscara de una dimensión vertical.

Con respecto a máscaras de convolución tridimensionales, habría que estudiar qué filtro se quiere realizar. La convolución pretende hacer una transformación continua de los píxeles de la imagen, teniendo en cuenta para el valor del nuevo píxel un entorno del píxel original; si nuestro filtro depende, por ejemplo, de los colores del píxel original y de los píxeles adyacentes al original, entonces no habría problema en considerar máscaras tridimensionales que recogieran los colores de cada píxel.

Sin embargo, lo habitual para los filtros que se usan regularmente es usar matrices de dos dimensiones que, en el caso de tratar con imágenes con más de un canal, operan sobre cada uno de los canales por separado y hacen después la reconstrucción.

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

Solución. El concepto de correlación y convolución es esencialmente el mismo: se trata de una operación local sobre los píxeles de una imagen tal que para cada uno de ellos se tienen en cuenta los píxeles adyacentes ponderados de alguna manera. Cómo se elige el entorno y la ponderación de los píxeles son los factores claves a la hora de desarrollar un filtro con significado visual diferente a otro. Y hasta aquí las dos operaciones son iguales.

Sin embargo, ténicamente son diferentes: si bien la correlación tiene en cuenta la máscara —que define el entorno y la ponderación— de forma directa, la convolución refleja la máscara en horizontal y vertical antes de aplicarla.

Por tanto, es directo el cómo hay que redefinir una máscara inicialmente diseñada para correlación para usarla en convolución: basta hacer el reflejo inverso tanto horizontal como vertical.

Así, aunque la definición es diferente, sus posibles usos en visión por computador son esencialmente iguales.

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

Solución. Los filtros de convolución definen funciones lineales. Para demostrarlo, tomemos la definición de convolución anterior y sean $\lambda \in \mathbb{R}$ una constante y F_1, F_2 dos imágenes. Entonces:

$$\begin{aligned}
H \star (\lambda(F_1 + F_2)) &= \sum_{u=-k}^{k} \sum_{v=-k}^{k} H(u,v)\lambda\Big(F_1(i-u, j-v) + F_2(i-u, j-v)\Big) = \\
&= \lambda\Big(\sum_{u=-k}^{k} \sum_{v=-k}^{k} H(u,v)(F_1(i-u, j-v) + F_2(i-u, j-v))\Big) = \\
&= \lambda\Big(\sum_{u=-k}^{k} \sum_{v=-k}^{k} H(u,v)F_1(i-u, j-v) + \\
&+ \sum_{u=-k}^{k} \sum_{v=-k}^{k} H(u,v)F_2(i-u, j-v)\Big) = \\
&= \lambda(H \star F_1 + H \star F_2)
\end{aligned}$$

Es evidente, por otro lado, que los filtros de mediana no son lineales, al no serlo la operación *mediana de un conjunto de píxeles*. Sean por ejemplo F_1 y F_2 las dos imágenes siguientes, cuya suma también inidicamos:

$$F_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 2 & 5 \\ 7 & 6 & 9 \end{pmatrix}; F_2 = \begin{pmatrix} 0 & 0 & 0 \\ 4 & 5 & 0 \\ 9 & 6 & 5 \end{pmatrix}; F_1 + F_2 = \begin{pmatrix} 0 & 0 & 0 \\ 4 & 7 & 5 \\ 16 & 12 & 14 \end{pmatrix}$$

El filtro de mediana con una máscara 3×3 sobre los píxeles centrales evidencia que no es una operación lineal, pues en F_1 vale 2, en F_2 vale 4 y en $F_1 + F_2$ vale $5 \neq 2 + 4$.

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

Solución. Los filtros de alisamiento suavizan los bordes, los grandes contrastes, los cambios bruscos de iluminación..., en definitiva, trabajan siempre sobre los lugares donde hay un cambio rápido en la intensidad lumínica; esto es, sobre los extremos de la derivada de la función intensidad.

Como la derivada es una operación local, es natural que la construcción de filtros de alisamiento sea con operaciones locales.

No tendría sentido aplicar una operación globalmente sobre la imagen cuando lo que buscamos es aplicar una modificación a los entornos donde la derivada cambia brusamente.

Cuestión 6. *Para implementar una función que calcule la imagen gradiente de una imagen dada pueden plantearse dos alternativas:*

1. *Primero alisar la imagen y después calcular las derivadas sobre la imagen alisada.*
2. *Primero calcular las imágenes derivadas y después alisar dichas imágenes.*

Discutir y decir qué estrategia es la más adecuada, si alguna lo es. Justificar la decisión.

Solución. Las dos alternativas que se nos presentan son las siguientes:

1. $\nabla f = [\frac{\partial}{\partial x}(h \star f), \frac{\partial}{\partial y}(h \star f)]$
2. $\nabla f = [h \star \frac{\partial}{\partial x} f, h \star \frac{\partial}{\partial y} f]$

Fijémonos en el número de operaciones que hace cada una: en el primer caso se hacen tres —una convolución y dos derivadas— y, en el segundo, cuatro —dos convoluciones y dos derivadas—. Por tanto, parece que computacionalmente la primera opción es la más adecuada por ser la más eficiente.

El resultado de ambas alternativas va a ser exactamente igual ya que, en virtud del teorema de derivación de la convolución, tenemos que:

$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x} h) \star f = h \star (\frac{\partial}{\partial x} f)$$

Como el resultado final no depende de la alternativa y, computacionalmente, la primera de ellas es más eficiente, podemos concluir que la primera es la más adecuada.

Cuestión 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.*

Solución. La Gaussiana en dos dimensiones es:

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Sus derivadas parciales son simétricas por serlo la Gaussiana:

$$\frac{\partial}{\partial x}G_\sigma(x,y) = -\frac{1}{2\pi\sigma^4}xe^{-\frac{x^2+y^2}{2\sigma^2}}$$
$$\frac{\partial}{\partial y}G_\sigma(x,y) = -\frac{1}{2\pi\sigma^4}ye^{-\frac{x^2+y^2}{2\sigma^2}}$$

Ambas son separables, en el sentido de que se pueden expresar como una función en x por una función en y:

$$\frac{\partial}{\partial x}G_\sigma(x,y) = \left(-\frac{1}{2\pi\sigma^4}xe^{-\frac{x^2}{2\sigma^2}}\right)\left(e^{-\frac{y^2}{2\sigma^2}}\right)$$
$$\frac{\partial}{\partial y}G_\sigma(x,y) = \left(-\frac{1}{2\pi\sigma^4}e^{-\frac{x^2}{2\sigma^2}}\right)\left(ye^{-\frac{y^2}{2\sigma^2}}\right)$$

En el proceso de convolución, el papel de dichos núcleos es simétrico: en el caso de $\frac{\partial}{\partial x}G_\sigma(x,y)$, la convolución dará como resultado una imagen en la que se resalten los cambios de intensidad en el eje X, esto es, se resaltarán los bordes verticales; en el caso de $\frac{\partial}{\partial y}G_\sigma(x,y)$ se resaltarán los cambios de intensidad en el eje Y, es decir, los bordes horizontales.

Cuestión 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.*

Solución. La Laplaciana de la Gaussiana está definida como sigue:

$$\nabla^2 G_\sigma(x,y) = \frac{\partial^2}{\partial x^2}G_\sigma(x,y) + \frac{\partial^2}{\partial y^2}G_\sigma(x,y)$$

Calculamos por tanto las derivadas que necesitamos:

$$\frac{\partial^2}{\partial x^2}G_\sigma(x,y) = -\frac{1}{2\pi\sigma^4}e^{-\frac{x^2+y^2}{2\sigma^2}}\left(1-\frac{x^2}{\sigma^2}\right)$$
$$\frac{\partial^2}{\partial y^2}G_\sigma(x,y) = -\frac{1}{2\pi\sigma^4}e^{-\frac{x^2+y^2}{2\sigma^2}}\left(1-\frac{y^2}{\sigma^2}\right)$$

Así, podemos escribir la Laplaciana de la Gaussiana como una suma de productos de funciones en x por funciones en y; esto es, podemos implementar la Laplaciada de la Gaussiana como una sucesión de núcleos de convolución tales que todos ellos son separables por filas y columnas:

$$
\begin{aligned}
\nabla^2 G_\sigma(x,y) &= \frac{\partial^2}{\partial x^2} G_\sigma(x,y) + \frac{\partial^2}{\partial y^2} G_\sigma(x,y) = \\
&= -\frac{1}{2\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}} \left(1 - \frac{x^2}{\sigma^2}\right) - \frac{1}{2\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}} \left(1 - \frac{y^2}{\sigma^2}\right) = \\
&= -\frac{1}{2\pi\sigma^4} e^{-\frac{x^2}{2\sigma^2}} e^{-\frac{y^2}{2\sigma^2}} \left(2 - \frac{x^2}{\sigma^2} - \frac{y^2}{\sigma^2}\right) = \\
&= -\frac{2}{2\pi\sigma^4} e^{-\frac{x^2}{2\sigma^2}} e^{-\frac{y^2}{2\sigma^2}} + \frac{x^2}{2\pi\sigma^6} e^{-\frac{x^2}{2\sigma^2}} e^{-\frac{y^2}{2\sigma^2}} + \frac{y^2}{2\pi\sigma^6} e^{-\frac{x^2}{2\sigma^2}} e^{-\frac{y^2}{2\sigma^2}} \\
&= \left(-\frac{2}{2\pi\sigma^4} e^{-\frac{x^2}{2\sigma^2}}\right) \left(e^{-\frac{y^2}{2\sigma^2}}\right) + \\
&+ \left(\frac{x^2}{2\pi\sigma^6} e^{-\frac{x^2}{2\sigma^2}}\right) \left(e^{-\frac{y^2}{2\sigma^2}}\right) + \\
&+ \left(e^{-\frac{x^2}{2\sigma^2}}\right) \left(\frac{y^2}{2\pi\sigma^6} e^{-\frac{y^2}{2\sigma^2}}\right)
\end{aligned}
$$

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

Solución. La primera aproximación a la reducción del tamaño de una imagen es evidente: si queremos reducir la imagen a la mitad, eliminamos una de cada dos filas y una de cada dos columnas. Esto es lo que se conoce como submuestreo. Sin embargo, haciendo sólo esto se consiguen resultados muy pobres: aparece el *aliasing*, que ocurre cuando el muestreo que se hace de una señal no es lo suficientemente grande como para capturar las frecuencias más altas. En el caso de las imágenes, lo que ocurre es que perdemos mucho detalle.

Sin embargo, si queremos reducir una imagen a la mitad, la solución no puede pasar por tomar más muestras de las que consideramos al principio. Por tanto, la solución tiene que pasar por eliminar las frecuencias más altas para que el muestreo original no sufra de *aliasing*; esto es, debemos suavizar la imagen. Esto se consigue, como hemos visto en teoría, con un filtro Gaussiano previo al submuestreo.

Así, los dos pasos que se siguen en la reducción del tamaño de una imagen son:

1. Filtro Gaussiano de la imagen original.
2. Submuestreo de la imagen tras el filtro.

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

Solución. Las pirámides Gaussianas no son más que sucesiones de filtros gaussianos y submuestreos, como hemos visto en la anterior pregunta, para reducir el tamaño de la imagen original. Por tanto, sabemos que las altas frecuencias se pierden, o al menos se atenúan o se suavizan.

Por tanto, es claro que la información visual que se conserva en los niveles superiores de una pirámide Gaussiana es *aquella relacionada con las frecuencias bajas*; es decir, aquellas zonas sin cambios bruscos de intensidad. En definitiva, se consigue conservar la *estructura* general de la imagen —formas generales y zonas grandes sin cambios— sacrificando los pequeños detalles, que se pierden con el filtro Gaussiano.

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

Solución. Sean G_i y G_{i+1} dos niveles sucesivos de una pirámide gaussiana, que ya hemos visto cómo se construye. Sea G'_{i+1} el nivel G_{i+1} pero expandido al tamaño del nivel G_i. Entonces, la pirámide laplaciana asociada se construye de forma que el nivel L_i es la diferencia siguiente:

$$L_i = G_i - G'_{i+1}$$

Si la pirámide Gaussiana tiene N niveles, convenimos que $L_N = G_N$, por no existir G_{N+1}.

La diferencia más clara es entonces la siguiente: mientras las pirámides Gaussianas conservan las frecuencias bajas de la imagen, los sucesivos niveles de las pirámides laplaciana conservan las frecuencias medias y altas, pues a la imagen original le vamos sustrayendo las frecuencias bajas.

Las aplicaciones de las pirámides laplacianas son varias, pero una de los más útiles es la codificación eficiente de imágenes, como se describe en el artículo *The Laplacian Pyramid as a Compact Image Code*, de Peter J. Burt y Edward H. Adelson. Allí se pone de manifiesto cómo la eficiencia de la compresión es mayor en aquellas imágenes con frecuencias más altas, como las que tenemos en los diferentes niveles de las pirámides laplacianas. La causa de esto es que las imágenes de la pirámide laplaciana tienen una entropía y varianza muy pequeña; así, la codificación se puede hacer de forma más basta sin salirse de los límites de distorsión impuestos por el sistema visual humano.

Cuestión 12. *¿Cuál es la aportación del filtro de Canny al cálculo de fronteras frente a filtros como Sobel o Roberts? Justificar detalladamente la respuesta.*

Solución. Los filtros Sobel o Roberts simplemente calculan una aproximación al gradiente de la imagen con dos máscaras de convolución consecutivas

que, en el caso de Sobel detectan bordes horizontales y verticales y, en el caso de Roberts, funcionan de forma óptima con los bordes a $\pm 45^{o}$.

El filtro de Canny hace un filtro con la derivada de la Gaussiana y encuentra igualmente el gradiente de la imagen. Hasta aquí la naturaleza de los tres filtros es la misma.

Sin embargo, Canny añade dos pasos más:

1. *Supresión de los no-máximos*: Para cada píxel de la imagen gradiente, se compara su intensidad con la de los píxeles adyacentes que están en la misma dirección del gradiente en ese punto. Si su intensidad es máxima en comparación con los otros, se conserva el píxel del borde; si no, se elimina.

2. *Histéresis y enlazado*: Se definen dos umbrales: alto y bajo. Los píxeles cuya intensidad supera el umbral alto definirán un borde, que se puede *unir* con píxeles entre los dos umbrales si son adyacentes. Por otro lado, todos los píxeles cuya intensidad es inferior al umbral bajo se eliminan.

¿Qué característica visual añade entonces este filtro? Si bien los filtros Sobel o Roberts daban imágenes gradiente donde los bordes se ven resaltados pero quedan difusos —muy anchos—, Canny garantiza una imagen donde cada borde tiene un píxel de ancho. La importancia de esto es clara cuando se necesita saber con precisión dónde se encuentra exactamente el borde, pregunta que no podemos responder con bordes anchos y borrosos como los que devuelven los otros dos filtros.

Además, la salida del filtro Canny es una imagen binaria, con píxeles cuyo valor es 0 o 255. En los otros dos casos, el paso de *thresholding* no existe y se devuelve una imagen en escala de grises.

Cuestión 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.*

Solución. En general, entornos industriales donde se tiene total control sobre la iluminación, la imagen capturada y los objetos que se quieren analizar son los mejores escenarios para trabajar con comodidad en problemas de visión por computador.

Así, una posible aplicación de este filtro puede ser la siguiente: imaginemos una fábrica donde se realizan chips con las últimas técnicas de miniaturización. En el proceso de la construcción de uno de estos chips se requiere soldar componentes de manera que el error tolerable que se puede cometer en la alineación del componente es mínimo.

Si queremos automatizar el proceso de análisis de la soldadura podemos usar técnicas de visión por computador: en particular, si se requiere, por ejemplo, que el componente sea completamente paralelo a uno de los lados

del chip, podemos fotografiar el chip de manera que se garantice que un componente correctamente soldado tendrá sus lados completamente paralelos a los lados de la imagen.

De este modo, al aplicar un filtro Canny con sus parámetros fijados experimentalmente, con la iluminación controlada de forma que se realce el lado del componente y con total conocimiento de la posición del borde del componente, podemos garantizar que todos los casos en los que la soldadura sea incorrecta serán devueltos por el algoritmo que se use.

Las razones son las siguientes:

- La verticalidad —u horizontalidad— del componente es crítica: es en estos casos donde el filtro Canny optimiza su salida, ya que al utilizar las derivadas en x e y los cambios que mejor se detectan son los que se tienen a través las direcciones de los ejes. Así, Canny garantiza encontrar el borde.

- La iluminación controlada es esencial: un estudio previo de la iluminación y su repercusión en el chip nos permitirá fijar experimentalmente los parámetros de forma óptima. Esto permite que tanto los umbrales del filtro Canny como el sigma de la convolución con la máscara Gaussiana sean los precisos para controlar el error de la soldadura a la precisión necesaria.

- La característica principal del filtro Canny —que los bordes que devuelve sean de un píxel de ancho— permitirá analizar de forma muy precisa y realmente sencilla si el lado del componente es paralelo al lado de la imagen: esto es, si los píxeles del borde están en una sola columna de la imagen o se mueven a lo largo de varias columnas. Esta precisión en las columnas de píxeles dependerá de la resolución de la cámara, pero podemos hacer un estudio previo como con la iluminación, de manera que el error tolerado devuelva siempre un borde cuyos píxeles estén en una sola columna, y se dispersen en varias si el error es mayor que el umbral.

La aplicación anterior es por tanto una en la que el filtro Canny garantiza unas fronteras interpretables que sirven como solución a un problema de visión por computador: automatizar la precisión de una soldadura.

2. Bonus

Bonus 1. *Usando la descomposición SVD (Singular Value Decomposition) de una matriz, deducir la complejidad computacional que es posible alcanzar en la implementación de la convolución 2D de una imagen con una máscara 2D de valores y tamaño cualesquiera (suponer la máscara de tamaño inferior a la imagen).*

Solución. Sea F una imagen cualquiera de $M \times N$ píxeles y H una máscara de convolución cualquiera de $m \times n$ píxeles, con la única restricción de que $m < M$ y $n < N$.

Sabemos que para realizar la convolución tenemos que calcular, para cada uno de los MN píxeles de la imagen F, mn operaciones. Por tanto, el coste computacional de la convolución de la máscara H sobre la imagen F es:

$$\mathcal{O}(MNmn)$$

Sabemos, por otro lado, que hay máscaras de convolución separables; esto es, que se pueden escribir como el producto de dos vectores. Si la máscara H es separable, entonces existen un vector columna h_1 de orden $m \times 1$ y un vector fila h_2 de orden $1 \times n$ tales que

$$H = h_1 \cdot h_2$$

donde $\cdot$ es el producto matricial.

En este caso, la convolución se puede *descomponer* de forma sucesiva con las máscaras h_1 y h_2 por separado. En este caso, por cada píxel tendríamos que hacer m operaciones en la convolución con h_1 y, después, n operaciones en la convolución con h_1. Por tanto, el coste computacional se puede reducir a:

$$\mathcal{O}(MNm + MNn) = \mathcal{O}(MN(m+n))$$

Si por ejemplo tomamos una máscara cuadrada de lado k, el coste computacional de la convolución es:

$$\mathcal{O}(MN(k+k)) = \mathcal{O}(2MNk) = \mathcal{O}(MNk)$$

Así que, en general, $\mathcal{O}(MN(m+n))$ es el mínimo coste computacional que vamos a poder tener en una convolución. La respuesta de cuándo lo alcanzaremos pasa necesariamente por ver cuándo una máscara de convolución es separable. Hagamos entonces ese estudio:

Si vemos la máscara de convolución H en el espacio $\mathcal{M}_{m\times n}(\mathbb{R})$ de las matrices de orden $m \times n$ valuadas en los reales, sabemos que existe su descomposición en valores singulares —SVD por sus siglas en inglés—; esto es, existen matrices $U \in \mathcal{M}_{m\times p}(\mathbb{R})$ y $V \in \mathcal{M}_{p\times n}(\mathbb{R})$ ortonormales —es decir,

que cumplen $U^T \cdot U = V^T \cdot V = I$—, con $p = mín(m, n)$, y una matriz diagonal $S \in \mathcal{M}_{p\times p}(\mathbb{R})$ tales que

$$H = USV^T \tag{1}$$

Si notamos por $u_i \in \mathcal{M}_{m\times 1}$, $i = 0, 1, \ldots, p-1$, los vectores columna que forman U, por $v_j^T \in \mathcal{M}_{1\times n}$, $j = 0, 1, \ldots, p-1$, los vectores fila que forman V^T, y por $\sigma_k \in \mathbb{R}$, $k = 0, 1, \ldots, p-1$, los valores que forman la diagonal de S —que llamamos *valores singulares*—, entonces tenemos que la expresión (1) se puede escribir como sigue:

$$H = \sum_{i=0}^{p-1} \sigma_i (u_i \cdot v_i^T) \tag{2}$$

Los σ_i son siempre no negativos y podemos ordenarlos de forma que $\sigma_0 \geq \sigma_1 \geq \cdots \geq \sigma_{p-1} \geq 0$. Entonces, si existe un único valor singular no nulo, este será σ_0 y, de (2) podemos deducir la siguiente expresión de H:

$$H = \sigma_0 (u_0 \cdot v_0^T) \tag{3}$$

De la expresión anterior es evidente que H se puede escribir como el producto de un vector fila por un vector columna:

$$H = (\sqrt{\sigma_0})^2 (u_0 \cdot v_0^T) = \sqrt{\sigma_0} u_0 \cdot \sqrt{\sigma_0} v_0^T =$$

$$= \begin{bmatrix} \sqrt{\sigma_0} u_{0,0} & \sqrt{\sigma_0} u_{0,1} & \cdots & \sqrt{\sigma_0} u_{0,m-1} \end{bmatrix} \cdot \begin{bmatrix} \sqrt{\sigma_0} v_{0,0} \\ \sqrt{\sigma_0} v_{0,1} \\ \vdots \\ \sqrt{\sigma_0} v_{0,n-1} \end{bmatrix}$$

Hemos demostrado, por tanto, el siguiente resultado:

Teorema 1. *Sea F una imagen de orden $M \times N$. Sea H una máscara de convolución de orden $m \times n$ tal que tiene un único valor singular no-nulo en su descomposición SVD. Entonces, el mínimo coste computacional que se puede alcanzar haciendo la convolución de la imagen F con la máscara H es*

$$\mathcal{O}(MN(m+n))$$

Restaría estudiar los demás casos, en los que la eficiencia de la convolución será menor: aquellos en los que la diagonal de S tiene más de un valor singular. El coste computacional en estos casos estará directamente relacionado con el número de $\sigma_i \neq 0$.

Sea $t \leq p-1$ el último σ_i no-nulo. Entonces, por (2), H se puede escribir como sigue:

$$H = \sum_{i=0}^{t} \sigma_i (u_i \cdot v_i^T)$$

Cada uno de los $t+1$ sumandos son, por el Teorema 1, separables, así que el coste computacional de la convolución en el caso anterior es entonces:

$$\mathcal{O}(tMN(m+n)) = \mathcal{O}(MN(m+n))$$

La notación $\mathcal{O}$ afirma que el coste computacional de la convolución tiene el mismo orden independientemente del número de valores singulares no-nulos. Esto es así en el marco teórico, pues las constantes se desprecian. Sin embargo, en estas operaciones la constante es muy importante: si la máscara es, por ejemplo, 7×7 y todos sus valores singulares son estrictamente positivos, el tiempo de computación se multiplicará por 7, una cantidad bastante significativa.

El estudio hecho hasta ahora es *muy* simplificado, pues estamos despreciando el coste que supone hacer la descomposición SVD y encontrar los vectores fila y columna que hacen la máscara separable.