Bloque 2: Operadores lineales

MATERIAL EXPERTOS	Experto 3: Restauración lineal

Aproximaciones a la realización de operadores locales lineales

La aplicación de un operador sobre una imagen que denominaremos original ($\psi[n,m]$), da lugar a una nueva imagen que denominaremos procesada ($\theta[n,m]$), Un operador local toma como entrada el valor de un entorno de píxeles en el entorno de uno dado y genera un valor resultante en el píxel homólogo de la imagen resultante. Por simplicidad asumiremos que el entorno es rectangular, de dimensiones M', N', según indica la Figura 1.

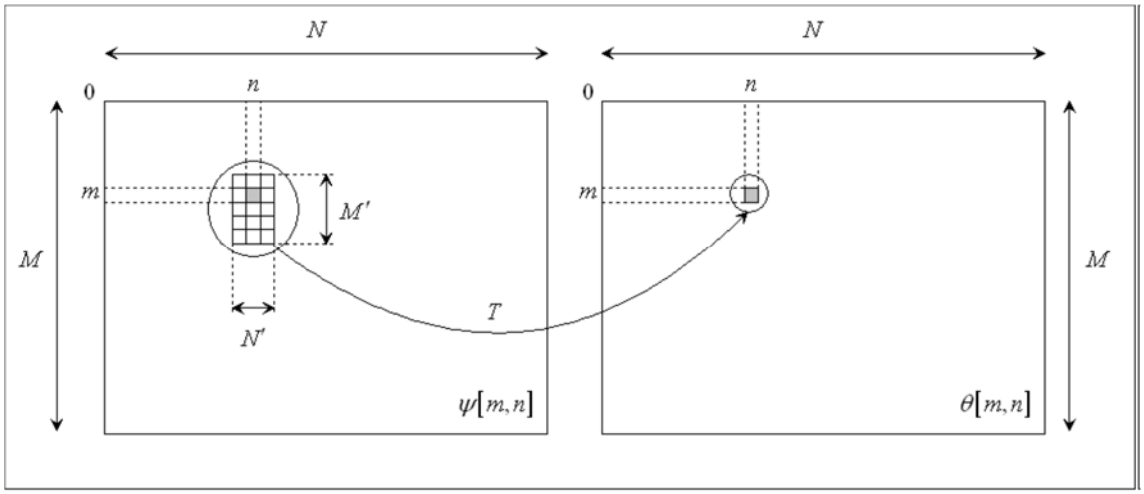

Figura 1: Esquema genérico de aplicación de un operador local

Según se ha visto en la parte teórica de la asignatura, si la operación que se efectúa sobre los píxeles del entorno es lineal, aplica la Teoría de Sistemas Lineales multidimensionales. MatLab ofrece herramientas específicas para esta situación.

Restauración lineal

Como han visto en teoría, el objetivo de la restauración lineal es recuperar una imagen que ha sido degradada de una manera conocida. El método consiste en modelar la degradación y aplicar el proceso inverso.

a. Restauración lineal en ausencia de ruido

Asuma que una imagen original $\psi(x,y)$ ha sido filtrada mediante un filtro $h(x,y)$ de manera que la imagen resultante $\theta(x,y)$ queda:

$$\theta(x,y) = \psi(x,y) * h(x,y), \quad \eta(x,y) = 0$$

, donde $\eta(x,y)$ representa el ruido de la señal.

En este caso, la señal original puede recuperarse prácticamente sin alteraciones si conocemos el filtro aplicado. Para ello, lo más cómodo es trabajar en el domino frecuencial:

$$\Psi(u,v)=\frac{\Theta(u,v)}{H(u,v)}, \Psi(u,v)\xrightarrow{IFT}\psi(x,y)$$

b. Restauración de una imagen afectada por ruido de cuantificación.

Dada una imagen filtrada, `Ifilt,` la existencia de ruido de cuantificación puede simularse mediante la expresión

```
Ifilt_Q = double(uint8(255.*Ifilt./max(Ifilt(:))));
```

, note que este ruido es casi inherente a todos los procedimientos que hemos realizado durante las prácticas de lo que se deriva que es fácil encontrar imágenes afectadas por este tipo de ruido.

En este caso, el ruido de la imagen no es nulo, si no que se produce al cuantificar los **más de 256** niveles de `Ifilt` en los **256** niveles de `Ifilt_Q.`

$$\theta(x,y)=\psi(x,y)*h(x,y)+\eta(x,y),\quad \eta(x,y)\neq 0$$

Para eliminar la influencia de este ruido en la restauración, pueden utilizarse estimadores del ruido y compensarlo en el dominio frecuencial.

$$\Psi^{\sim}(u,v)=\frac{\Theta(u,v)}{H(u,v)}=\Psi(u,v)+\frac{N(u,v)}{H(u,v)}$$

Uno de los más usados es el filtrado por Wiener o MMSE, que en la práctica se aplica en el dominio frecuencial mediante la expresión:

$$\Psi^{\sim}(u,v)=\left[\frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2+\kappa}\right]\Theta(u,v)$$

, donde κ es el factor de estimador de ruido. En MatLab, este filtro puede definirse mediante la expresión:

```
Hc=H.*conj(H);
w_filter=(1./H).*(Hc./(Hc+k));
```

, donde `H` es el filtro en frecuencia utilizado para la transformación local lineal.

c. Restauración de una imagen afectada por ruido blanco Gaussiano.

El ruido blanco Gaussiano es el más usado para la verificación de algoritmos de estimación de ruido, dada su naturaleza aditiva, se entiende que, si no es muy intenso, puede detectarse y compensarse.

Dada una imagen filtrada, `Ifilt,` la existencia de ruido de blanco Gaussiano puede simularse mediante la expresión

```
Ifilt_G=imnoise(Ifilt,'gaussian',0,0.001);
```

, documéntese sobre la función `imnoise` mediante la ayuda de MatLab para conocer los parámetros que modelan el ruido.

El filtrado por Wiener descrito en el apartado anterior tiene como principal limitación que requiere conocer los espectros de potencia de señal y ruido como datos para la estimación del parámetro κ.

Esto es raramente posible en la práctica. El filtrado CLS ofrece una alternativa cuyo parámetro modelador del ruido γ (`gamma`) es un escalar, y además requiere exclusivamente del conocimiento del ruido, en este caso de la media y la desviación típica del ruido gaussiano.

Su aplicación en frecuencia viene dado por la ecuación:

$$\tilde{\Psi}(u,v) = \left[\frac{H^*(u,v)}{|H(u,v)|^2 + \gamma |L(u,v)|^2} \right] \Theta(u,v)$$

, donde $L(u,v)$ es un filtro Laplaciano. En MatLab, esta operación puede realizarse mediante la expresión:

```
f_lapc= laplacian_filter.*conj(laplacian_filter);
cls_filter=conj(H)./(Hc+gamma*f_lapc);
```

, donde `laplacian_filter` es un filtro Laplaciano definido sobre el mismo retículo utilizado para la definición de `H`.