Análisis de media de diferencia entre imágenes y su posterior implementación usando Matlab

Proyecto Final de Carrera

Roberto de la Ossa Pérez y Raúl Notario Vaquero

Mayo 2015

Gracias a todas aquellas personas que han soportados nuestros lamentos estos años, especialmente a nuestros familiares.

RESUMEN DEL PROYECTO.

El proyecto consta de dos partes principales y dos anexos.

La primera es teórica, en ella realizamos; a modo de introducción, un estudio sobre el tratamiento digital de la imagen, desarrollando las principales técnicas de tratamiento y análisis de imágenes que pudimos estudiar durante la carrera. Una vez desgranado el análisis nos centraremos en la correlación digital de imagen, su evolución y distintas técnicas, donde nos centramos en la correlación cruzada normalizada que usamos posteriormente para la correlación de imágenes con Matlab.

La segunda parte consiste en la implementación de un sencillo programa mediante Matlab en el que podremos evaluar y analizar las diferencias entre dos o más imágenes, pudiendo observar gráficamente la desviación en milímetros entre varias imágenes y su dirección con vectores.

Posteriormente analizamos los resultados obtenidos y proponemos posibles mejoras para futuros proyectos de correlación de imágenes digitales.

Por último, incluimos un par de anexos en los que incluimos un tutorial para automatizar acciones con Adobe Photoshop para facilitar el pretratamiento de fotografías antes de analizarlas con el script y una posible práctica de laboratorio para futuros alumnos de la escuela utilizando nuestro script de Matlab.

PROJECT ABSTRACT.

The project involves two main parts and two annexes.

The first is theoretical, it performed; by way of introduction, a study on digital image processing, developing the main treatment techniques and image analysis we were able to study along our career. Once shelled analysis we will focus on digital image correlation, evolution and different techniques, where we focus on normalized cross-correlation which we use later for the correlation of images with Matlab.

The second part is the implementation of a simple program using Matlab where we can evaluate and analyze the differences between two or more images and can graphically see the deviation in millimeters between various images and their direction vectors.

Then we analyze the results and propose possible improvements for future projects correlation of digital images.

Finally, we have a couple of annexes in which we include a tutorial to automate actions with Adobe Photoshop to facilitate pretreatment photographs before analyzing the script and a possible lab for future school students using our Matlab script.

Índice

1. Planteamiento del proyecto

El proyecto consta de dos partes principales.

La primera es teórica, en ella realizamos; a modo de introducción, un estudio sobre el tratamiento digital de la imagen. Una vez desgranado el análisis nos centraremos en la correlación digital de imagen mediante su evolución y distintas técnicas.

La segunda parte consiste en la implementación de un sencillo programa mediante Matlab en el que podremos evaluar y analizar las diferencias entre dos o más imágenes apoyándonos en la parte teórica del proyecto.

Posteriormente analizamos los resultados obtenidos y proponemos posibles mejoras para futuros proyectos de correlación de imágenes digitales.

Por último, incluimos un par de anexos en los que incluimos un tutorial para automatizar acciones con Adobe Photoshop y una posible práctica de laboratorio utilizando nuestro script de Matlab.

2. Estudios previos

2.1 Teoría de tratamiento digital de la imagen (TDI)

Realizaremos una introducción al proyecto basándonos en la asignatura de TDI para introducir los conceptos básicos.

2.1.1 Fundamentos de imágenes digitales

Podemos atacar el concepto de imagen según múltiples criterios: etimológicos, fotográficos, artísticos… En nuestro caso una definición simple sería que la imagen es la representación visual de un objeto. Por otro lado tenemos la palabra digital (relativo a los dedos) que hace referencia a la información en modo binario.

Estas definiciones nos permiten indicar que una imagen digital es una representación bidimensional construida a partir de una matriz binaria.

2.1.2. Tratamiento digital de imágenes

MUESTREO Y CUANTIFICACIÓN

La imagen monocroma continua. Es una imagen en escala de grises, no en blanco y negro. En la parte práctica de nuestro proyecto usaremos este tipo de imágenes, sobre ellas llevaremos a cabo las técnicas y procesos que describiremos posteriormente.
Definimos la imagen matemáticamente con la siguiente función:

$$z = f(x, y) \qquad \text{(Ecuación 1)}$$

Siendo x e y las coordenadas espaciales en el plano y z el brillo o nivel de gris.

Las tres variables (x, y, z) son continuas, pero (x, y) se discretizan y pasan a ser (m, n) con lo que ya no es una función sino una tabla.

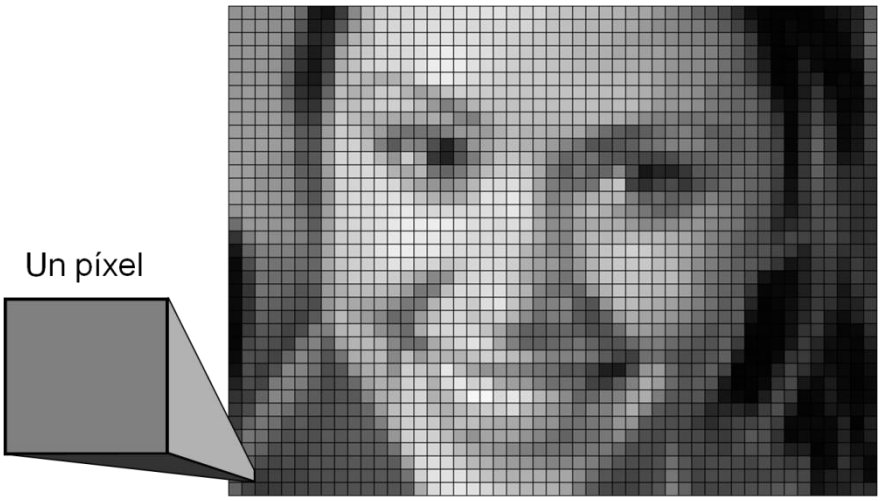


Imagen 1. Imagen con sus píxeles

Realmente es una matriz de m filas y n columnas, por lo que las imágenes se consideran matrices. El valor del brillo; z, también se discretiza.

Una imagen es una matriz en la que cada elemento es un píxel (elemento de imagen). Al codificar los niveles, al negro le corresponde el 0 mientras que el blanco tiene el valor 255 (hexadecimalmente).

IMÁGENES CROMÁTICAS

Estas se descomponen en 3 imágenes monocromáticas, correspondientes a los componentes R, G, B. Por lo tanto; cada imagen, serán 3 matrices y cada componente se suele representar como escala de grises (no como imagen roja, verde y azul).

$$f(x,y) \sim \begin{bmatrix} f(0,0)\ f(0,1) \dots\ f(0,M-1) \\ f(1,0)\ f(1,1) \dots\ f(1,M-1) \\ \vdots \\ f(N-1,0)\ f(N-1,1) \dots\ f(N-1,M-1) \end{bmatrix} \qquad \text{(Ecuación 2)}$$

Se suelen usar matrices cuadradas de 256*256 ó 512*512 (según empleemos 8 ó 9 bits).

MUESTREO

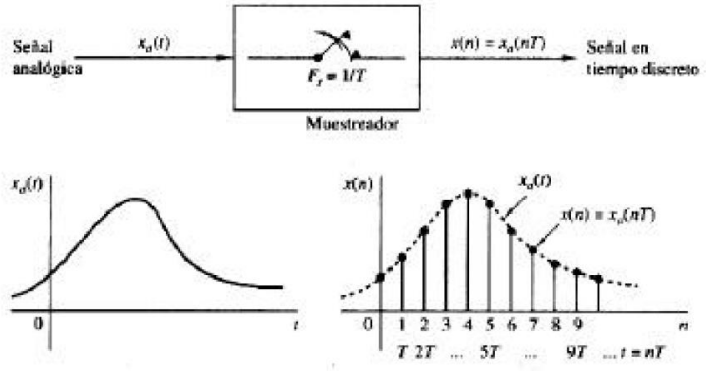
Consiste en la digitalización o discretización de la posición de cada píxel.

$$(x,y)\ valores\ continuos \rightarrow (m,n)\ valores\ discretos \qquad \text{(Ecuación 3)}$$

Tenemos que tener una relación de compromiso en la cantidad de muestras que tomemos ya que si cogemos muchas ocuparemos demasiada memoria, por el contrario si tomamos pocas tendremos efectos perjudiciales en la imagen a tratar (efecto mosaico).

La resolución establece el número de píxeles que tiene la imagen tanto en horizontal como en vertical, podemos disminuir la resolución manteniendo la sensación de calidad si disminuimos también la superficie de la imagen en la misma proporción.

PRINCIPIOS DEL MUESTREO



Muestreo de una señal analógica

Imagen 2

En frecuencia:

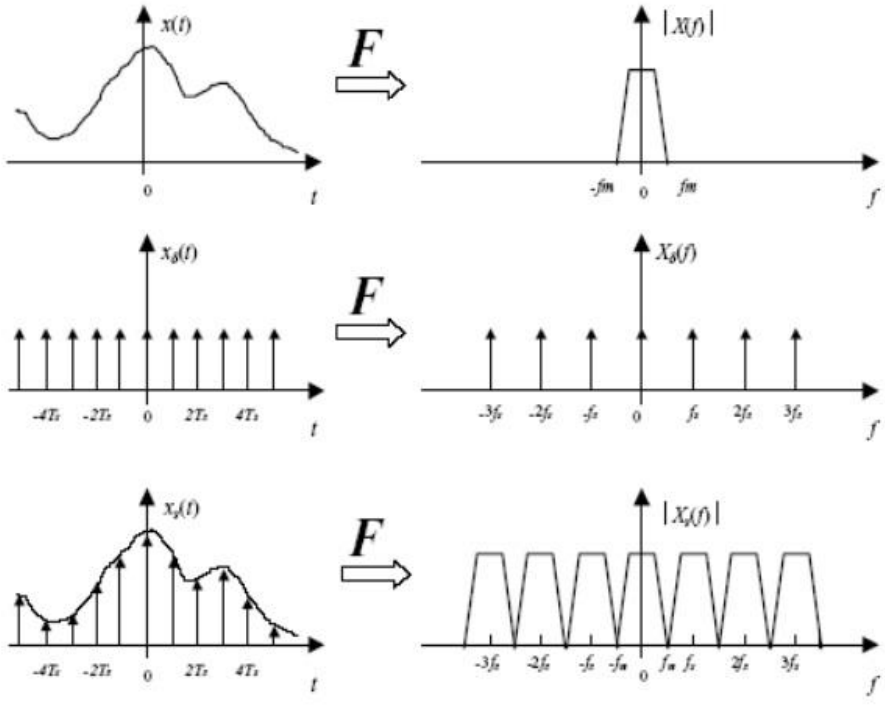


Imagen 3. Muestreo en frecuencia

Para recuperar la señal se usa un filtro paso bajo antes de entrar al muestreador debido a que se tiene que cumplir el teorema de Nyquist para que la señal no sufra aliasing. Sino la señal original sería irrecuperable.

$$f_m > 2f_{máx} \quad \text{(Ecuación 4)}$$

SEÑALES BIDIMENSIONALES

Pasamos al dominio de la frecuencia (espectro) a través de la transformada de Fourier Bidimensional, de forma que aparecen dos frecuencias: una horizontal y otra vertical. La imagen también debe ser filtrada y debe cumplir:

$$F(w_x, w_y) = 0 \begin{cases} w_x > w_{ox} \\ w_y > w_{oy} \end{cases} \quad \text{(Ecuación 5)}$$

Definimos la Región de Soporte como el margen de frecuencias utilizado por el espectro de la señal (viéndolo desde arriba).

En el muestreo usamos una señal muestreadora que es una parrilla de deltas.

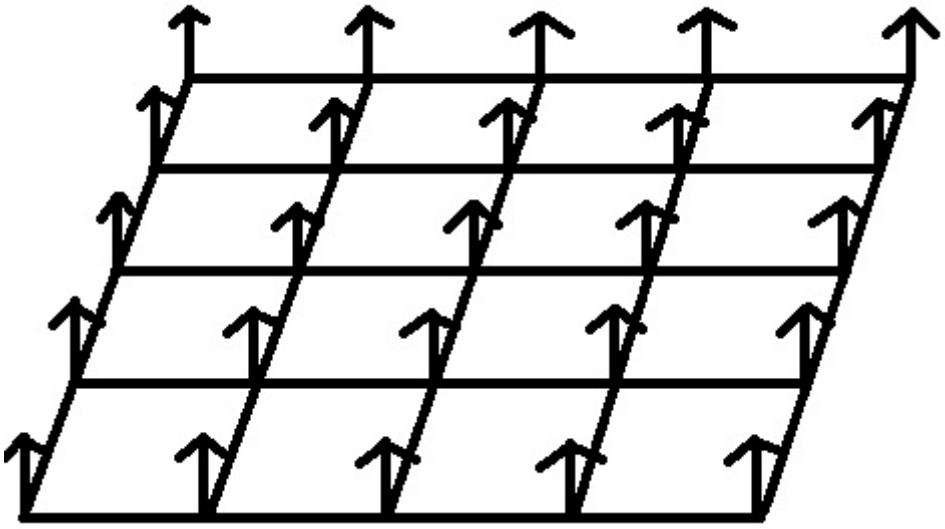


Imagen 4. Parrilla de deltas

Señal a muestrear:

$$f(x, y) \quad \text{(Ecuación 6)}$$

Señal muestreadora:

$$\sum\sum [\delta(x - j\Delta x), \delta(y - \Delta y)] \quad \text{(Ecuación 7)}$$

Señal muestreada:

$$\sum\sum \delta(j\Delta x, k\Delta y)[\delta(x - j\Delta x), \delta(y - \Delta y)] \quad \text{(Ecuación 8)}$$

El espectro de la señal muestreada es convolucionar esta con el espectro de la señal (un espectro en cada delta).

Para recuperar la señal filtramos en las deltas antes de muestrear para evitar el aliasing antes mencionado.

CUANTIFICACIÓN

Se realiza mediante un conversor analógico/digital. A la entrada tenemos infinitos brillos pero en la salida tenemos un número finito de valores en función de los bits que utilicemos para cuantificar la señal (256 valores si usamos 8 bits).

La cuantificación introduce un error que no puede recuperarse. El conversor A/D consta de dos pasos: cuantificador y codificador, el error se produce en el cuantificador y dependerá del escalón de cuantificación usado, cuanto más pequeño sea este, más pequeño será el error. El error no será nunca mayor que la unidad del escalón cuántico ya que a la salida se le asigna el valor de la mitad de dicho escalón.

El error también depende de la señal de entrada, es aleatorio. Con este error aparece lo que llamamos **ruido de cuantificación.**

$$-\frac{a}{2} < e < \frac{a}{2} \;\; ; \;\; a = \frac{V_{máx}}{2^n} = \frac{V_{máx}}{N} \quad \text{(Ecuación 9)}$$

Definimos la potencia de la señal: $S = x^2$.

Definimos la potencia del ruido: $N = \frac{a^2}{12}$ (valor esperado del error).

Relación Señal/Ruido: $\frac{S}{N} = \frac{12x^2}{a^2}$.

$$\frac{S}{N}(dB) = 10{,}8 + 6n + 20log\left(\frac{x}{V_{máx}}\right) \quad \text{(Ecuación 10)}$$

Por cada incremento de 1 bit, mejoramos 6 dB la relación S/N.

De manera práctica tenemos dos algoritmos de cuantificación: por truncamiento y por redondeo.

1. Cuantificación por truncamiento:

   Valor cuantificado = parte entera [Valor/escalón]

   Al coger la parte entera de la división lo que hacemos es eliminar los decimales.

2. Cuantificación por redondeo:

   Valor cuantificado = redondeo [Valor/escalón]

   Técnicamente es igual que sumar la mitad del escalón cuántico y tomar la parte entera.

2.1.3 Transformaciones

2.2.1.1 Diezmado

Consiste en eliminar píxeles de acuerdo a un ritmo prefijado. Reduce el tamaño de la imagen resultante. **Se elimina información que no es posible recuperar.** Puede generar aliasing por lo que es necesario un filtrado previo.

2.2.1.2 Interpolación

Consiste en generar más píxeles de los existentes ayudándonos de los propios píxeles de la imagen por lo que no se puede añadir información pero incrementamos el tamaño de la imagen. Existen varias técnicas de interpolación.

1. La interpolación lineal es una interpolación unidimensional en la que cada píxel es repetido tantas veces como indiquemos, con lo que la imagen definitiva será x veces superior a la original. Puede ser de orden cero en la que los píxeles interpolados se repiten con el mismo valor que el píxel original o de primer orden en la que los píxeles interpolados varían según la media de los píxeles originales vecinos.

   Pese a su sencillez el uso de esta técnica es desaconsejable en el suavizado de imágenes por sus efectos indeseables.

2. La interpolación bilineal se trata de una técnica más sofisticada que la anterior en la que interpolamos a partir de los cuatro píxeles adyacentes que rodean al punto deseado. Se implementa a partir de la adquisición de los coeficientes de la siguiente ecuación para los cuatro píxeles vecinos a interpolar:

   $$u(x, y) = c_1 + c_2 y + c_3 xy + c_4 \qquad \text{(Ecuación 11)}$$

   Donde u(x, y) representa la intensidad del píxel y (x, y) la coordenada donde se redefinirá la intensidad del píxel.

3. La interpolación cúbica se traza una cúbica entre los cuatro puntos más próximos (dos a cada lado).

4. La interpolación bicúbica se realiza interpolando cúbicamente en horizontal y en vertical de forma que intervienen los dieciséis puntos que rodean al que se

quiere localizar. Es muy útil para realizar zoom en la imagen y reducir el sobremuestreo. El valor de la intensidad aplicado al píxel responde a:

$$u(x,y) = \sum_{j=0}^{3}\sum_{i=0}^{3} a_{ji} x^i y^i \qquad \text{(Ecuación 12)}$$

Donde los dieciséis coeficientes a_{ji} se obtienen de los dos sumatorios de la ecuación anterior.

Debe quedar claro que cada píxel de salida depende de uno o varios píxeles de entrada y se calcula de acuerdo a determinadas funciones (como no serán posiciones enteras, habrá que interpolar) por lo que no se puede inventar información, lo más que se puede aspirar es a perder la mínima información y no provocar aliasing.

2.2.1.3 Tratamiento estadístico de imágenes:

El valor de un píxel puede considerarse como una variable aleatoria con valores comprendidos entre 0 y 255. Su probabilidad se obtiene al dividir el número de veces que ha aparecido en la imagen, entre el número total de píxeles.

2.2.1.3.1 Brillo medio

En una imagen es la suma de todos los brillos dividido por el número total de píxeles de la imagen.

$$\mu = x = \frac{1}{M \cdot N}\sum_{u=0}^{255} u_i \, n_i \qquad \text{(Ecuación 13)}$$

Siendo M y N las dimensiones de la imagen, u_i el valor de brillo de 0 a 255 y n_i es el número de veces que se aparece ese valor de brillo.

2.2.1.3.2 Varianza de una imagen

Es la valoración existente entre los brillos de los píxeles y el valor medio, elevado al cuadrado. Valora el contraste de la imagen, mide la separación respecto de la media.

$$\sigma^2 = \frac{1}{M \cdot N}\sum_{i=1}^{M}\sum_{j=1}^{N}[x(i,j) - \mu]^2 \qquad \text{(Ecuación 14)}$$

2.2.1.3.3 Desviación típica

Es la raíz cuadrada de la varianza. σ.

2.2.1.3.4 Histograma

El histograma de una imagen representa de forma gráfica el número de veces que aparece cada brillo. En el eje de abscisas tendremos los distintos brillos de la imagen y en el eje de ordenadas la frecuencia con que aparecen dichos brillos.

Mediante la ecualización del histograma podemos ajustar los parámetros de brillo y contraste de la imagen, para ello primero debemos convertir el histograma en una función de densidad de probabilidad normalizando los brillos entre 0 y 1, lo que conseguimos dividiendo las ordenadas entre el número total de píxeles.

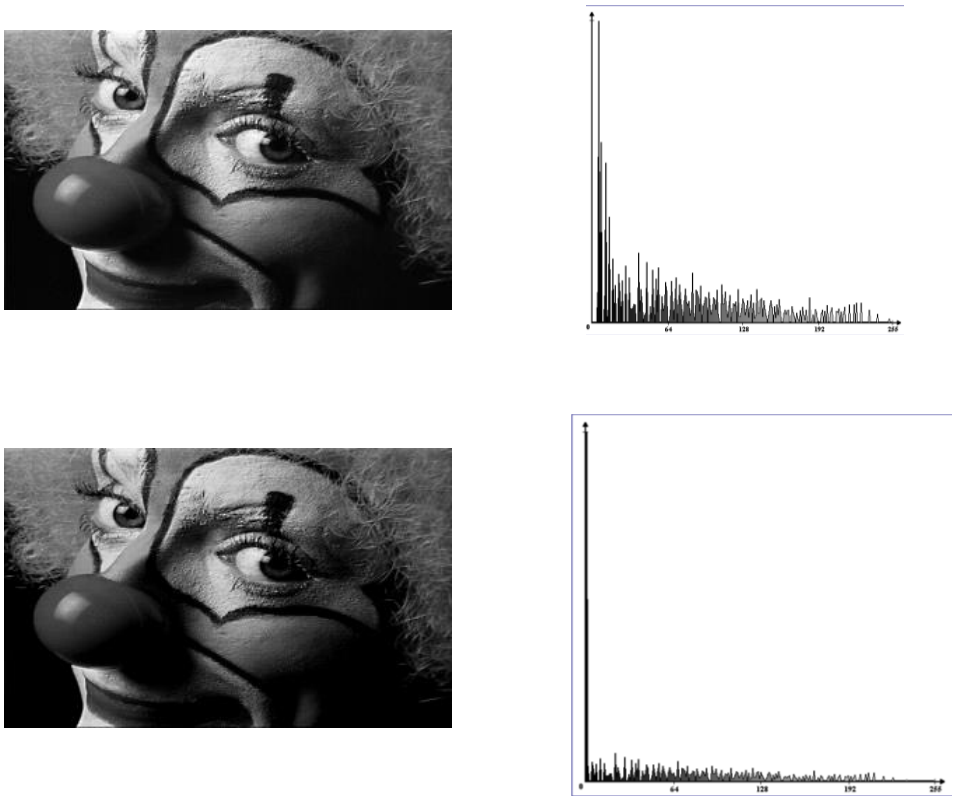


Imagen 5. Misma imagen con diferentes brillos

2.2.1.4 Operaciones de punto

A partir de una imagen $u[m,n]$, creamos otra $v[m,n]$ en la que el brillo depende únicamente del brillo del píxel homólogo. La transformación sólo depende del brillo del punto transformado, no de los adyacentes, también se denomina **transformación de intensidad.**

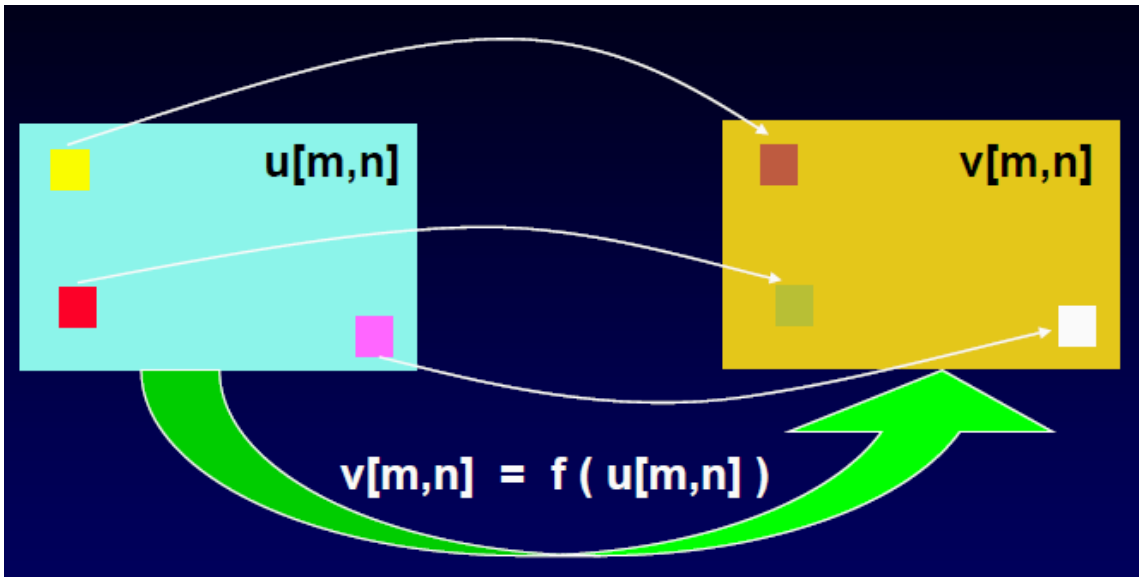


Imagen 6. Operaciones de punto. Las posiciones de los píxeles no se modifican, sólo cambia su valor.

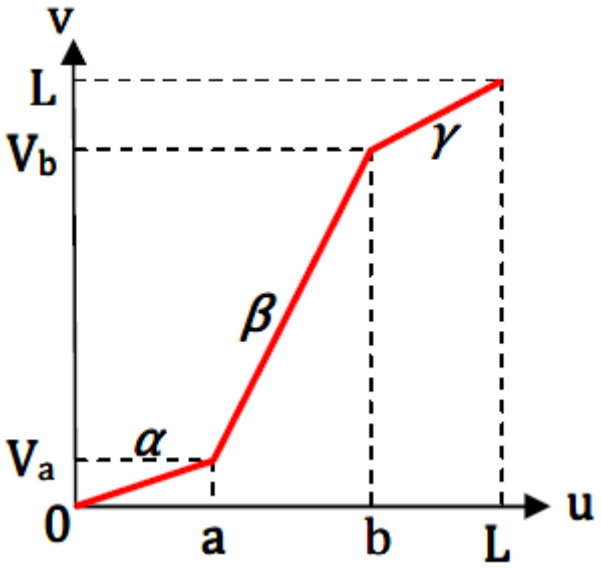
Llamando "u" a la variable aleatoria correspondiente al brillo de la imagen original y "v" al de la imagen modificada, se cumple que:

$$v = f(u) \qquad \text{(Ecuación 15)}$$

1. Stretching

   Se denomina estiramiento o ampliación de contraste. En las imágenes con poco contraste los valores de los niveles de gris de los píxeles que la componen toman valores similares, en el histograma se apreciaría como en una pequeña zona de niveles de píxeles habría mucha frecuencia de aparición.

$$v = f(u) = \begin{cases} \alpha u + V_0 & 0 \le u < a \\ \beta(u-a) + V_a & a \le u < b \\ \gamma(u-b) + V_b & b \le u \le L \end{cases} \qquad \text{(Ecuación 16)}$$



Gráfica 1

Se pueden observar tres tramos, cuando la pendiente es mayor que la unidad se produce estiramiento, si es menor que la unidad se efectúa una compresión.

La mayor parte de las operaciones siguientes puede considerarse un caso particular de esta para distintos valores de:

$$\alpha\ \beta\ \gamma\ a\ b\ v_0\ v_a\ v_b\ v_{255} \qquad \text{(Ecuación 17)}$$

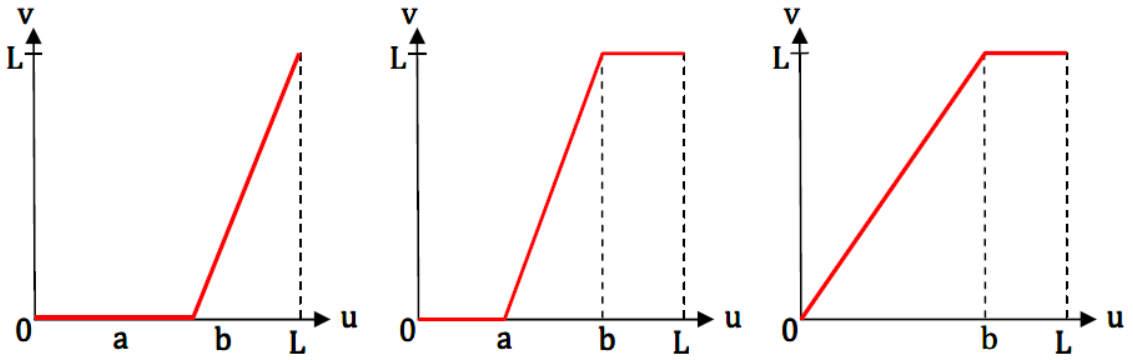
2. Clipping

Tenemos tres casos: recorte por la parte inferior, recorte por la parte superior y recorte por ambos lados.

$$v = f(u) = \begin{cases} v = 0 & si\ \ u < a \\ v = L\dfrac{(u-a)}{(L-a)} & si\ \ u > a \end{cases} \qquad \text{(Ecuación 18)}$$

$$v = f(u) = \begin{cases} v = u(L/a) & si\ \ u < a \\ v = L & si\ \ u > a \end{cases} \qquad \text{(Ecuación 19)}$$

$$v = f(u) = \begin{cases} v = 0 & si\ \ u < a \\ v = L\dfrac{(u-a)}{(b-a)} & si\ \ a < u < b \\ v = L & si\ \ u > b \end{cases} \qquad \text{(Ecuación 20)}$$



Gráfica 2. Ecuaciones 18, 19 y 20 respectivamente

Con esta operación podemos ampliar de forma drástica el contraste de un número reducido de niveles de gris, el rango de niveles se sitúa donde se desea resaltar el contraste, permitiendo diferenciar los niveles que se encuentren por encima y por debajo del rango en cuestión. En este caso, los niveles que están por debajo del umbral "a", pasan a valer un único valor (0), mientras que los niveles que se encuentren por encima de "b" tendrán un valor máximo (L). Por lo que la operación clipping corresponde a un stretching donde los valores de niveles fuera del intervalo [a, b] sufren la máxima compresión. Usamos esta técnica para reducir el ruido cuando se sabe que los valores de brillo se encuentran en un cierto rango de valores.

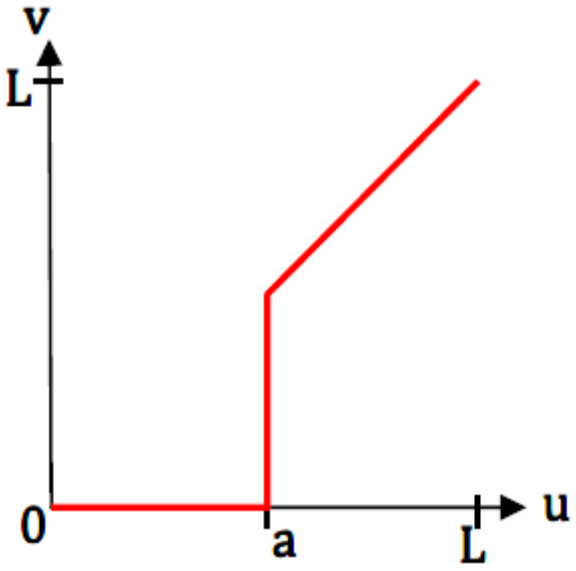
3. Umbralización

En este caso particular dividimos el histograma forzando los píxeles mediante la umbralización inferior o superior.

- Umbralización inferior

Establecemos un umbral mínimo (a) respetando el brillo de los píxeles que se encuentren por encima y forzando a blanco o a negro los que se encuentren por debajo del umbral.

$$v = f(u) = \begin{cases} v = 0 & si\ u < a \\ v = u & si\ u > a \end{cases} \qquad \text{(Ecuación 21)}$$

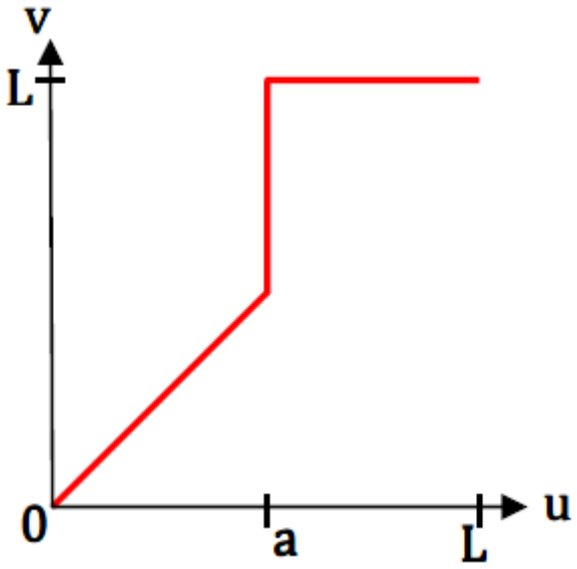


Gráfica 3

- Umbralización superior

Establecemos un umbral máximo (a) respetando el brillo de los píxeles que se encuentren por debajo y forzando a blanco o a negro los que se encuentren por encima del umbral.

$$v = f(u) = \begin{cases} v = u & si\ \ u < a \\ v = L & si\ \ u > a \end{cases} \qquad \text{(Ecuación 22)}$$



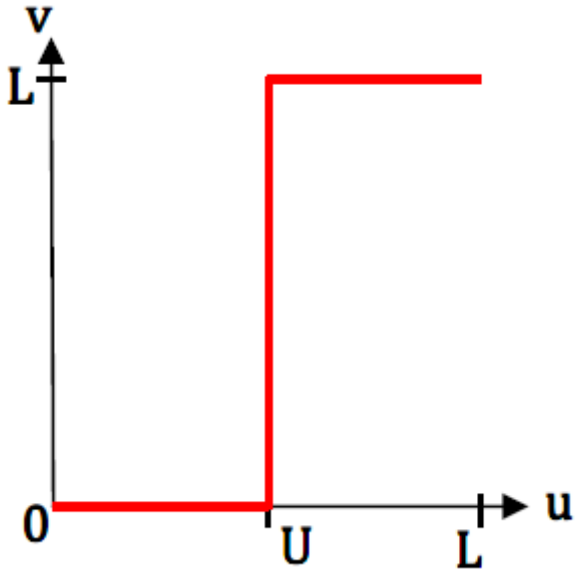
Gráfica 4

4. Binarización

No hay que confundirlo con umbralización, ya que aunque usemos el concepto de umbral (a) son dos ideas diferentes. Esta operación genera una

imagen binaria (a dos niveles). Es usada en ciertas aplicaciones de reconocimiento óptico de caracteres (OCR), elevando el umbral mejoramos la visualización del texto al ser letras negras sobre fondo blanco.

$$v = f(u) = \begin{cases} v = 0 & si\ u < a \\ v = L & si\ u > a \end{cases} \qquad \text{(Ecuación 23)}$$



Gráfica 5

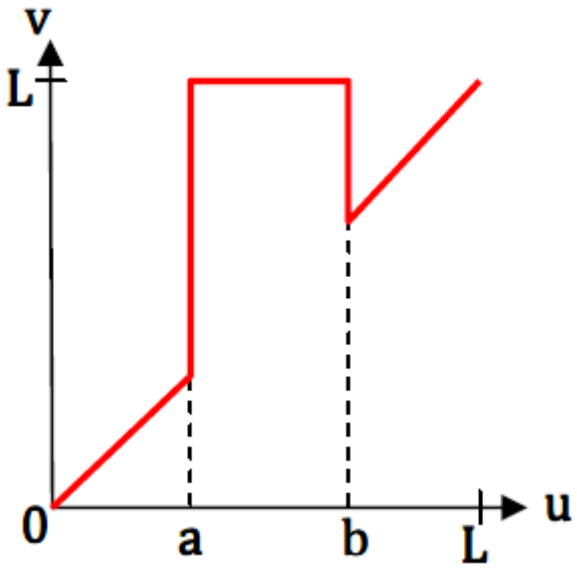
5. Slicing

Puede traducirse por franja, rodaja, sector… Mediante esta técnica resaltamos un determinado rango de niveles de brillo según sea la importancia de la información que contengan dentro de la imagen. Existen varios tipos posibles de slicing.

- Conservando el fondo

Se resaltan zonas de la imagen con un brillo determinado, mientras que el resto se mantiene.

$$v = f(u) = \begin{cases} v = L & a < u < b \\ v = u & otro\ caso \end{cases} \qquad \text{(Ecuación 24)}$$

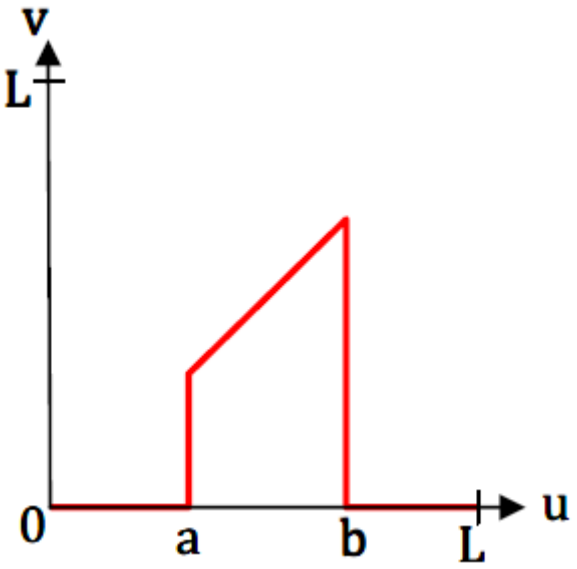


Gráfica 6

- Eliminando el fondo

Se mantienen zonas de la imagen con un brillo determinado, el resto se fuerza.

$$v = f(u) = \begin{cases} v = u & a < u < b \\ v = L & otro\ caso \end{cases} \qquad \text{(Ecuación 25)}$$

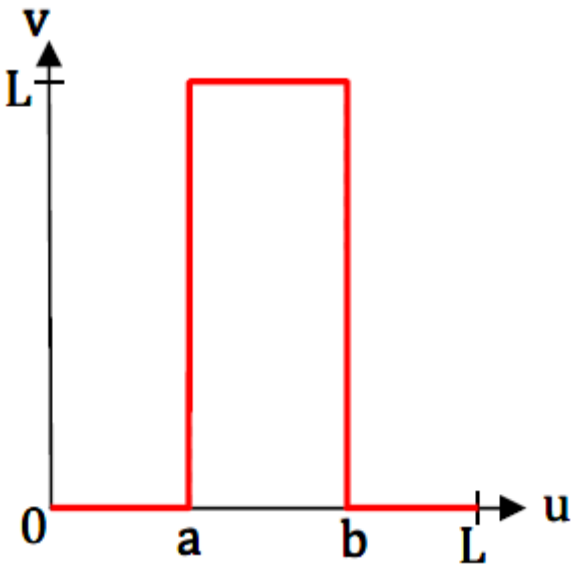


Gráfica 7

- Separando franja y resto

Se mantienen zonas de la imagen con un brillo determinado, el resto se fuerza.

$$v = f(u) = \begin{cases} v = 0 & a < u < b \\ v = L & otro\ caso \end{cases}$$ (Ecuación 26)

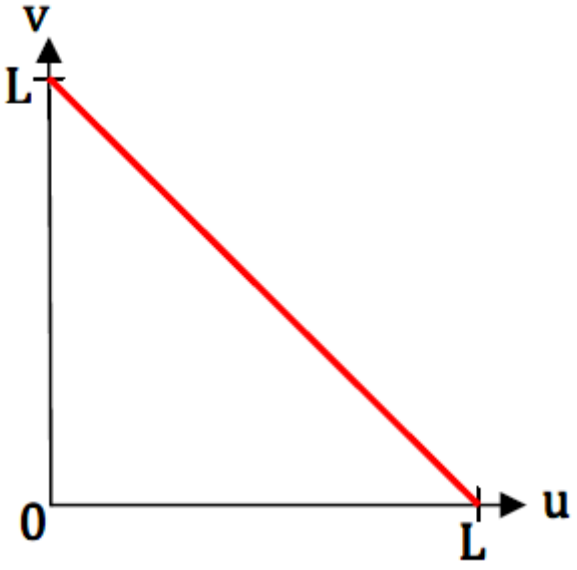


Grafica 8

6. Complementación

   A veces se le llama inversión de forma incorrecta. Consiste en sustituir cada píxel de la imagen por su complementario.

$$v = f(u) = L - u$$ (Ecuación 27)



Grafica 9

2.2.1.5 Operaciones aritméticas

Son muy útiles en vídeo, modelos acumulados, detección de movimiento, transparencias difusas, etc.

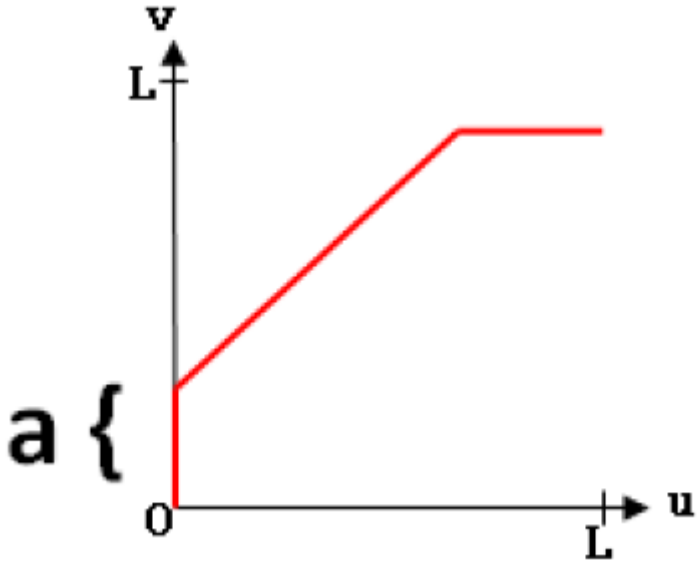
1. Operaciones unarias

   • Sumar una constante

   Sumamos al brillo de cada píxel un valor constante (a), incrementando el brillo de la imagen en este valor. En el histograma veríamos como se desplaza a la derecha "a" píxeles.

   $$R(x,y) = A(x,y) + a \qquad \text{(Ecuación 28)}$$

   Debido a este desplazamiento pueden existir píxeles cuyo brillo supere el máximo soportado por el dispositivo de salida lo que produce un error llamado overflow, debemos comprobar que ningún píxel supere el brillo 255 o sea inferior al 0. Estos píxeles; coloquialmente, decimos que están saturados, lo que supone una pérdida de información, por lo que deberemos recortar estos valores. En imágenes en color, la suma se realiza sobre los tres canales (R, G, B) y siempre con el mismo valor.

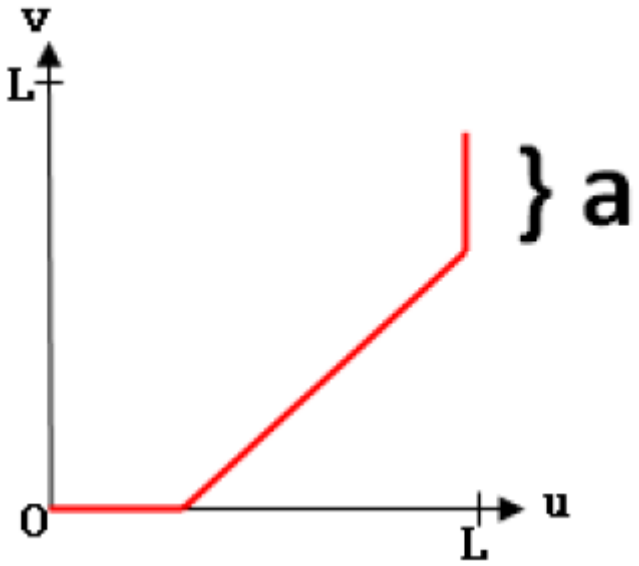


Gráfica 10

   • Restar una constante

   Restamos al brillo de cada píxel un valor constante, decrementando el brillo de la imagen este valor (a). En el histograma veríamos como se desplaza a la izquierda "a" píxeles.

   $$R(x,y) = A(x,y) - a \qquad \text{(Ecuación 29)}$$

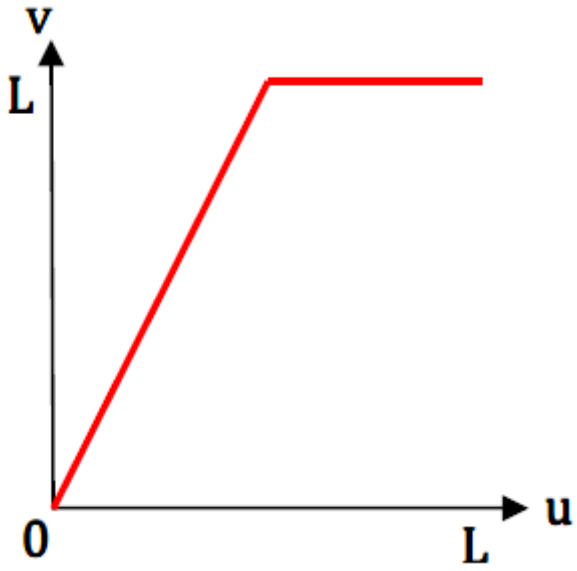


Gráfica 11

- Multiplicar por una constante

Aumenta la intensidad de la imagen en un valor (b). El histograma de la imagen se "estira" hacia la izquierda.

$$R(x,y) = b * A(x,y) \qquad \text{(Ecuación 30)}$$



Gráfica 12

Tanto en la suma como en la multiplicación, se aumenta al nivel de gris de los píxeles pero de forma distinta.

En la suma, el parámetro "a" (número entero) indica el número de niveles de gris a incrementar: de -255 a 255.

En el producto, el parámetro "b" (número real) indica el factor a incrementar:

- b = 1 indica ningún cambio.
- b = 2 indica que se duplica el valor de gris. Los píxeles con brillo mayor a 127 se saturarán.
- b = 0,5 indica que el histograma se encoge a la mitad.
- b < 1 indica que estamos dividiendo por una constante. Lo que obviamente supone que el histograma se encoja.

2. Combinación de imágenes

Se trata de utilizar dos o más imágenes de entrada (A y B) para producir una imagen de salida (R).

$$R(x,y) = f\langle A(x,y), B(x,y)\rangle \qquad \text{(Ecuación 31)}$$

El valor del píxel resultante es función de los píxeles de A y B en la misma posición. En principio, todas las imágenes deben ser del mismo tamaño.

• Suma

Al sumar dos imágenes obtenemos como resultado una tercera imagen que es la mezcla de ambas.

$$R(x,y) = A(x,y) + B(x,y) \qquad \text{(Ecuación 32)}$$

Tenemos que reducir previamente el rango de las imágenes a sumar debido al problema de la saturación, para ello se dividen los valores de los píxeles entre el número de imágenes que participan en la suma, obteniendo una imagen media de las originales con lo que será semitransparente.

$$R(x,y) = \frac{A(x,y) + B(x,y)}{2} \qquad \text{(Ecuación 33)}$$

Las imágenes tendrán diferentes patrones de ruido ya que cada imagen se produce en instantes diferentes de tiempo. Al hacer la media de todas ellas, obtenemos como resultado una imagen donde el ruido aleatorio ha sido atenuado porque la media refuerza los valores de los píxeles que no varían en ambas imágenes.

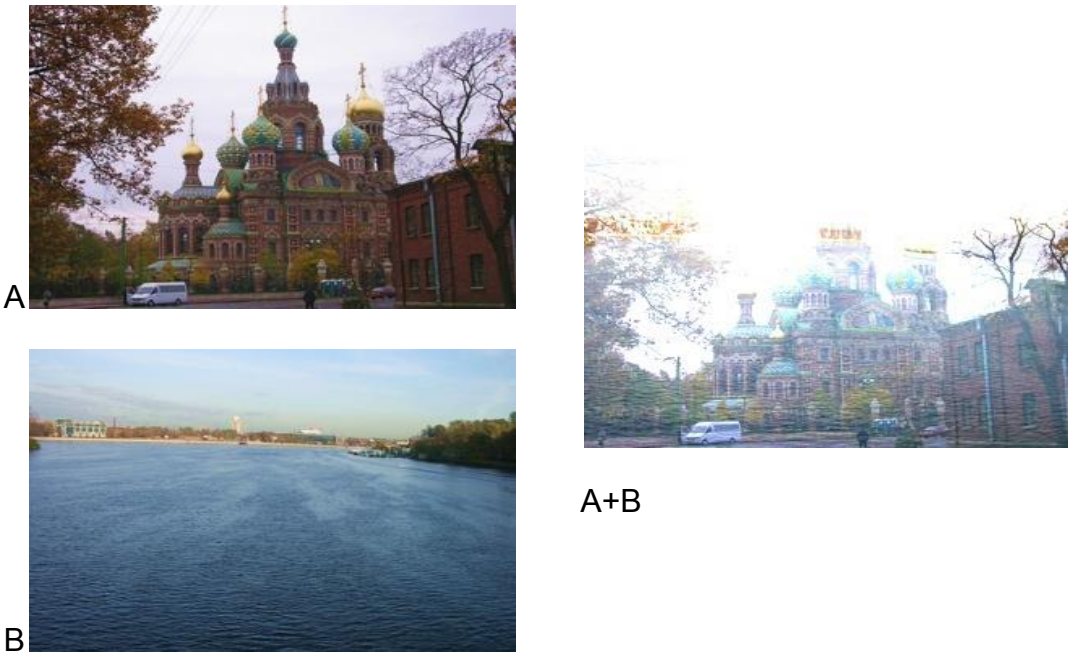


Imagen 6. Suma de imágenes

- Resta

Significa que obtenemos como salida la diferencia de las imágenes de entrada. Es una técnica de gran aplicación en segmentación y realce debido a que es muy útil para detectar el cambio producido en dos imágenes que han sido captadas en dos instantes de tiempo diferentes.

$$R(x,y) = A(x,y) - B(x,y) \qquad \text{(Ecuación 34)}$$

También se puede producir saturación, en este caso los brillos se irán a cero, para restar dos imágenes manteniendo el rango de salida aplicamos la siguiente fórmula:

$$R(x,y) = \frac{A(x,y) - B(x,y)}{2} + 128 \qquad \text{(Ecuación 35)}$$

Por otra parte, la mayoría de las veces lo que nos interesa es conocer la diferencia entre las imágenes, por lo que la solución es tomar el valor absoluto de la resta.

$$R(x,y) = abs\langle A(x,y) - B(x,y)\rangle \qquad \text{(Ecuación 36)}$$

Los píxeles negros que obtengamos indican que las dos imágenes son iguales en esos píxeles; por el contrario, cuanto más clara es una zona, más se diferencian las imágenes.

A

A-B

B

B-A

Imagen 7. Resta de imágenes

- Producto

El producto de dos imágenes da como resultado una totalmente saturada por lo que debemos escalar la imagen de salida (dividirla por 255).

$$R(x,y) = \frac{A(x,y) * B(x,y)}{255} \qquad \text{(Ecuación 37)}$$

Esta técnica puede ser usada para realizar una transformación de intensidad diferente para cada píxel.

A

A*B

B

Imagen 8. Multiplicación de imágenes

- División

Es la operación contraria al producto por lo que debemos realizar otro escalado, pero esta vez multiplicando por 255.

$$R(x,y) = \frac{A(x,y)}{B(x,y)} * 255 \qquad \text{(Ecuación 38)}$$

3. Operaciones lógicas

Estas operaciones tienen sentido cuando al menos una de las imágenes es binaria y actúa como máscara de la otra, de manera que el nivel de negro es FALSE y el de blanco en TRUE.

Los principales operadores son:

- AND
- OR
- XOR

- NOT

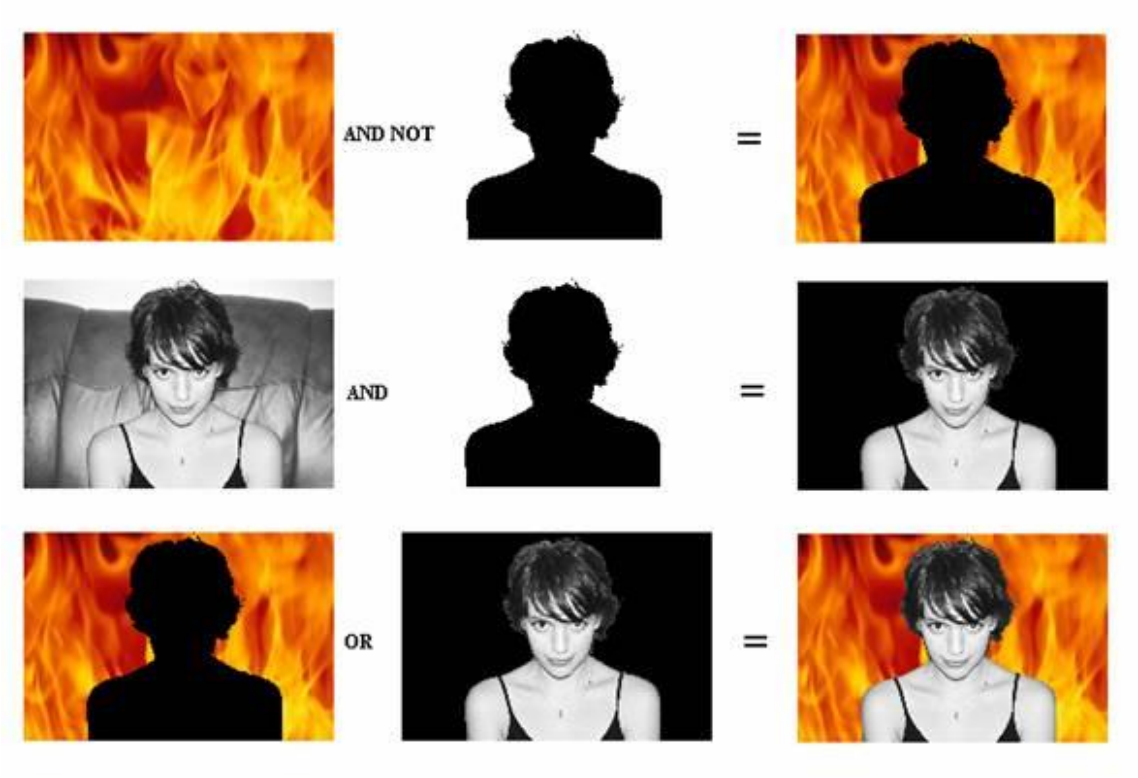


Imagen 9

2.2.1.6 Introducción de ruido

El ruido en una imagen digital se puede explicar como cualquier valor de un píxel que no se corresponde con la realidad, es la suma de la imagen original con otra imagen formada por ruido. Al adquirir las imágenes, estas están contaminadas con ruido debido a los equipos de captación utilizados (ruido de cuantificación de la imagen, efecto niebla…) y al ruido añadido en los tramos de transmisión (posibles interferencias o errores al transmitir los bits de información).

1. Ruido uniforme

   La función densidad de probabilidad es tal que todos los valores de ruido tienen la misma probabilidad.

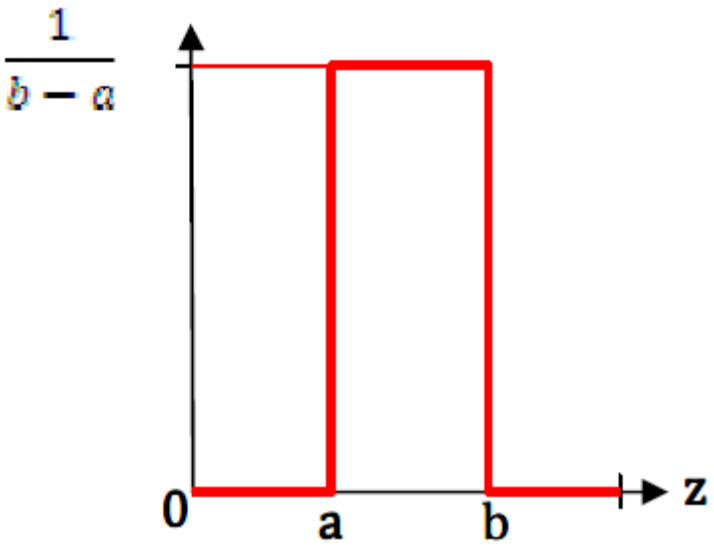
$$p(z) = \begin{cases} \frac{1}{b-a} & si\ \ a \le z \le b \\ 0 & en\ otro\ caso \end{cases} \qquad \text{(Ecuación 39)}$$

   Con valores de media y desviación típica:

$$\mu = \frac{a+b}{2} \qquad \text{(Ecuación 40)}$$

$$\sigma^2 = \frac{(b-a)^2}{12} \qquad \text{(Ecuación 41)}$$

No es un ruido muy frecuente, pero se utiliza mucho en simulaciones.

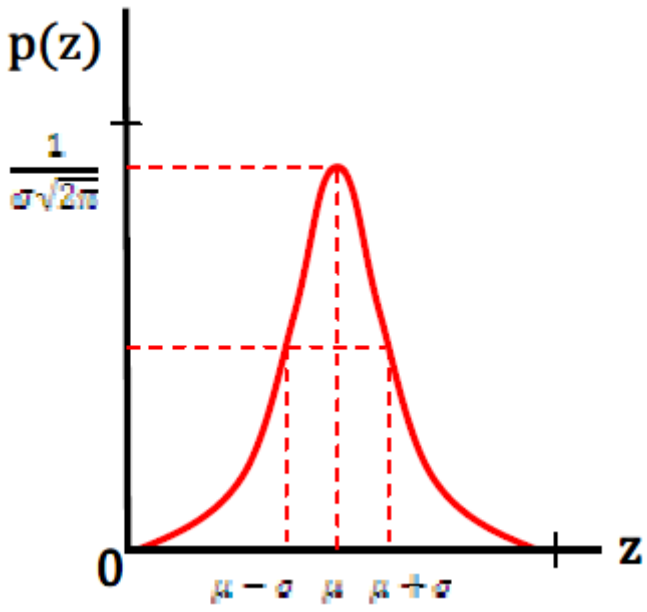


Gráfica 13

2. Ruido gaussiano

La función densidad de probabilidad es:

$$p(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}} \qquad \text{(Ecuación 42)}$$



Gráfica 14

Los parámetros que lo definen son la media (μ) y la desviación típica (σ).

El 70% de las veces el ruido se encuentra comprendido entre los valores de $(\mu - \sigma)$ y $(\mu + \sigma)$. El 95% de las veces el ruido se encuentra comprendido entre los valores de $(\mu - 2\sigma)$ y $(\mu + 2\sigma)$.

Se genera debido a problemas en circuitos o temperaturas altas.

3. Ruido exponencial

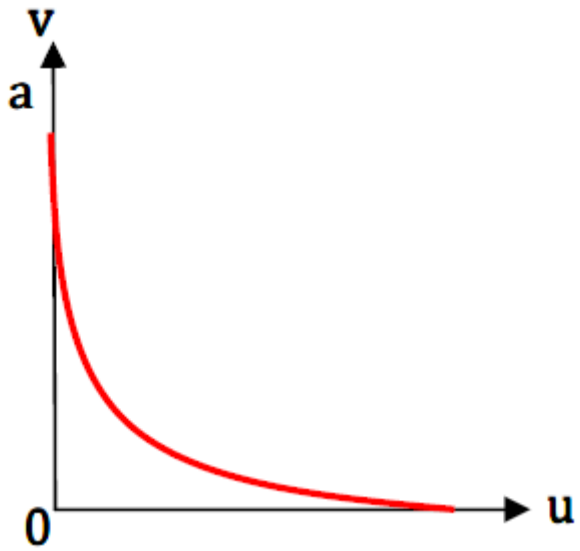
Su función densidad de probabilidad es de la forma:

$$p(z) = \begin{cases} ae^{-az} & para\ z \geq a \\ 0 & para\ z < a \end{cases} \quad \text{(Ecuación 43)}$$

Con valores de media y desviación típica:

$$\mu = \frac{1}{a} \quad \text{(Ecuación 44)}$$

$$\sigma^2 = \frac{1}{a^2} \quad \text{(Ecuación 45)}$$



Gráfica 15

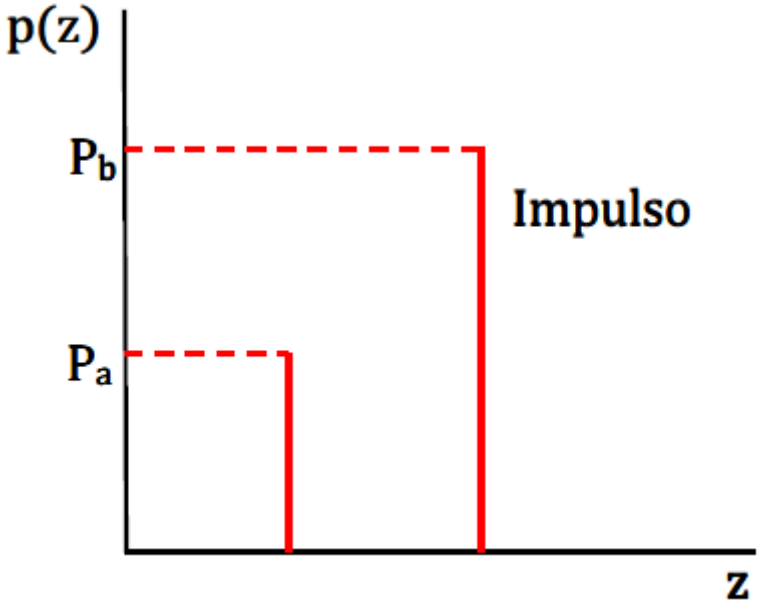
4. Ruido sal y pimienta

Es causado principalmente por fallos en el funcionamiento de los sensores encargados de capturar una imagen o por errores de tiempo cuando se produce el proceso de digitalización de la misma.

En el momento de digitalizar la imagen este tipo de ruido suele tomar valores extremos en la imagen (cercanos al brillo 0 y al brillo 255), esto es debido a que los impulsos de ruido pueden ser negativos o positivos. Generalmente se supone que los valores de a y b se encuentran saturados, ya sea en sus valores máximos o en sus valores mínimos cuando se digitaliza la imagen.

Su función densidad de probabilidad es:

$$p(z) = \begin{cases} P_a & para\ z = a \\ P_b & para\ z = b \\ 0 & en\ otro\ caso \end{cases} \qquad \text{(Ecuación 46)}$$



Gráfica 16

Cuando las dos probabilidades son parecidas, el efecto es similar al de "sal y pimienta"; puntos blancos y puntos negros y de ahí su nombre.

5. Ruido de Rayleigh

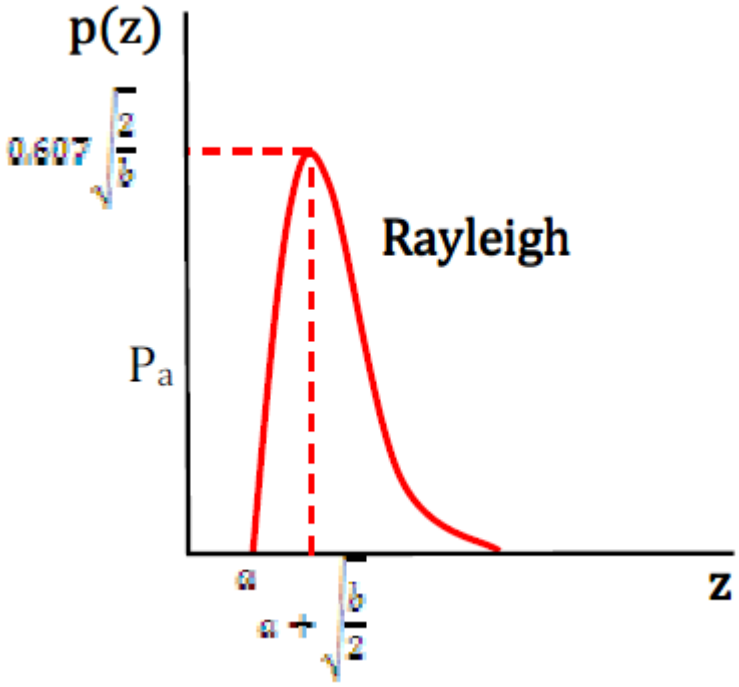
Su función densidad de probabilidad es:

$$p(z) = \begin{cases} \frac{2}{b}(z-a)e^{-\frac{(z-a)^2}{b}} & para\ z \geq a \\ 0 & para\ z < a \end{cases} \qquad \text{(Ecuación 47)}$$

Siendo su valor de media y desviación típica:

$$\mu = a + \sqrt{\frac{\pi b}{4}} \qquad \text{(Ecuación 48)}$$

$$\sigma^2 = \frac{b(4-\pi)}{4} \qquad \text{(Ecuación 49)}$$



Gráfica 17

Este ruido suele generares en el proceso de obtención de imágenes.

6. Ruido gamma

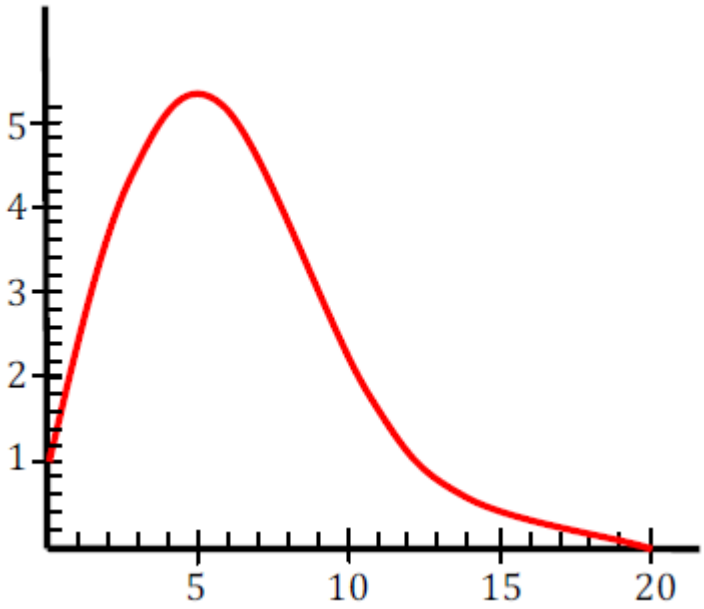
Su función densidad de probabilidad es:

$$p(z) = \begin{cases} \frac{a^b z^{b-1}}{(b-1)!} e^{-az} & para\ z \geq a \\ 0 & para\ z < a \end{cases} \qquad \text{(Ecuación 50)}$$

Con valores de media y desviación típica definidos por:

$$\mu = \frac{b}{a} \qquad \text{(Ecuación 51)}$$

$$\sigma^2 = \frac{b}{a^2} \qquad \text{(Ecuación 52)}$$



Gráfica 18

7. Ruido periódico

   Este tipo de ruido suele producirse por interferencias eléctricas o electromagnéticas durante la adquisición de la imagen.

   Este ruido no es aleatorio, sino completamente determinista, tiene una amplitud, una frecuencia y una posición relativa sobre la imagen.

   Si la imagen tiene tres brillos, tendremos tres rayos debido a este ruido, para deshacernos de él basta con realizar un filtrado en el dominio de la frecuencia ya que el ruido periódico produce picos de frecuencia en el espectro de Fourier que son fácilmente identificables.

2.2.1.7 Operaciones geométricas

A diferencia de las operaciones aritméticas o de la introducción al ruido, en las operaciones geométricas no se modifican los brillos de los píxeles que componen las imágenes sino que se modifican las posiciones donde se van a ubicar.

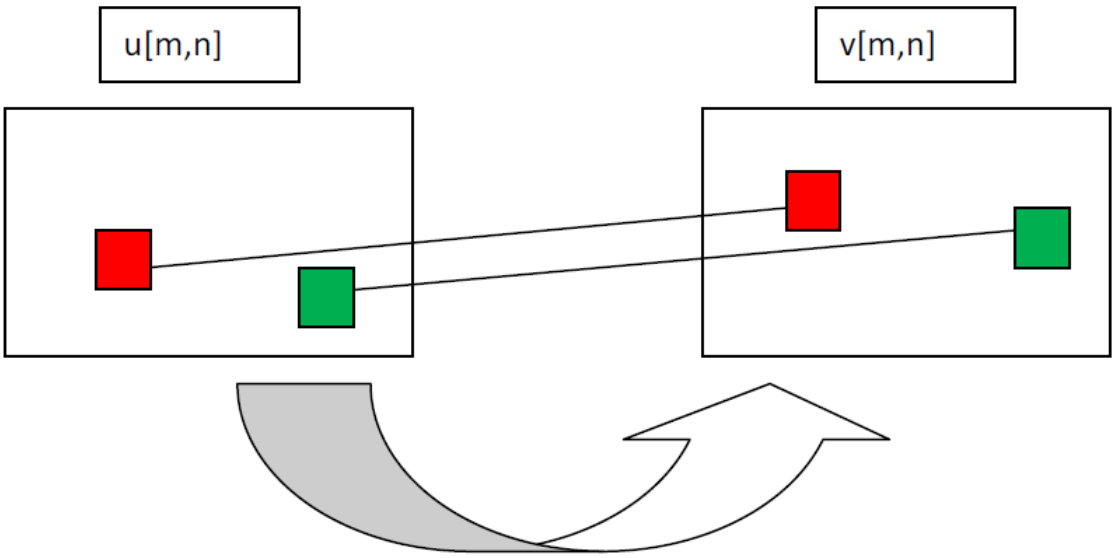


Imagen 10

$$v[m', n'] = u[f1(m,n), f2(m,n)] \quad \text{(Ecuación 53)}$$

El píxel de llegada se busca en algún punto de la imagen de entrada.

Tenemos varios tipos de operaciones geométricas:

1. Simetría:

   Con la simetría, la imagen mantiene la colocación de los píxeles en un eje mientras los invierte en el otro. Si la inversión se realiza en el eje x, la simetría será horizontal mientras que si se realiza en el eje y será vertical.

   • Simetría horizontal

   Consiste en intercambiar las columnas de la imagen.

$$x' = x;\ y' = Y_0 - y \quad \text{(Ecuación 54)}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \left[\begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & Y_0 \\ 0 & 0 & 1 \end{pmatrix} \right] \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \text{(Ecuación 55)}$$

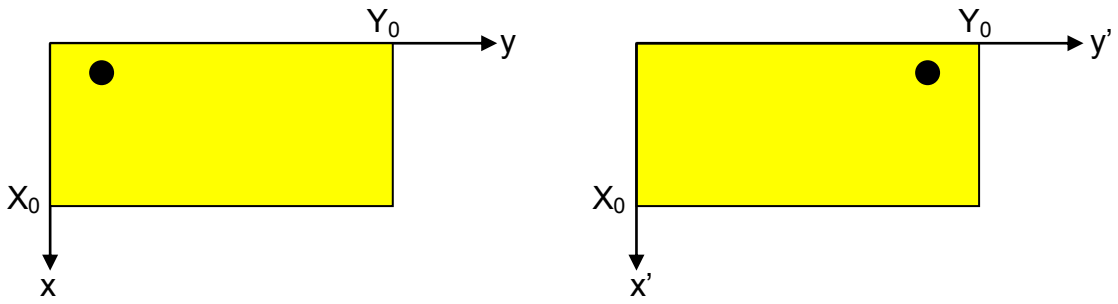


Imagen 11

Tomando (x, y) como coordenadas en la imagen original y (x´, y´) como las coordenadas en la imagen simétrica.

- Simetría vertical (flip)

Consiste en intercambiar las filas de una imagen.

$$x' = X_0 - x;\ y' = y \qquad \text{(Ecuación 56)}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \left[\begin{pmatrix} -1 & 0 & X_0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}\right] \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad \text{(Ecuación 57)}$$

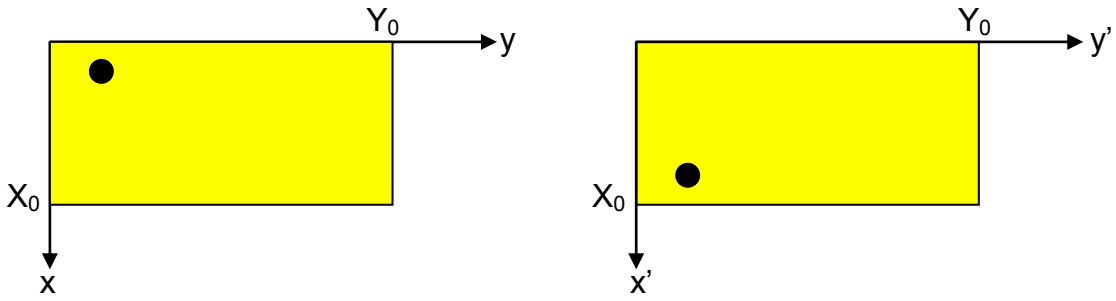


Imagen 12

Tomando (x, y) como coordenadas en la imagen original y (x´, y´) como las coordenadas en la imagen simétrica.

2. Translaciones

Es una operación geométrica donde una posición determinada por un píxel (x, y) se desplaza a una nueva posición empleando unos desplazamientos $(\Delta x, \Delta y)$, según las siguientes ecuaciones. Para ello el fondo debe ser de un tamaño tal que aloje a las imágenes.

$$x' = x + \Delta x;\ y' = y + \Delta y \qquad \text{(Ecuación 58)}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \left[\begin{pmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{pmatrix}\right] \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad \text{(Ecuación 59)}$$

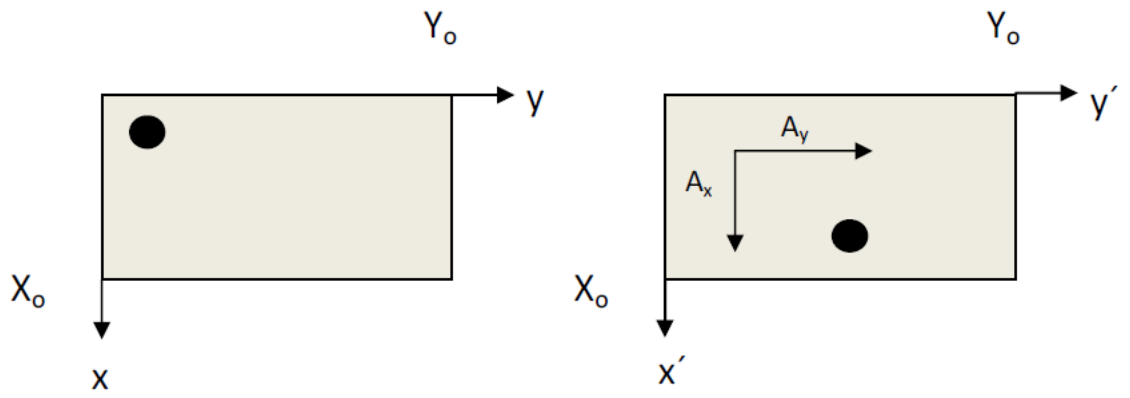


Imagen 13

3. Escalado

Es una operación geométrica que permite cambiar la escala de la imagen, según las siguientes ecuaciones:

$$x' = k_1 x;\ y' = k_2 y \qquad \text{(Ecuación 58)}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \left[\begin{pmatrix} k_1 & 0 & 0 \\ 0 & k_2 & 0 \\ 0 & 0 & 1 \end{pmatrix}\right] \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad \text{(Ecuación 58)}$$

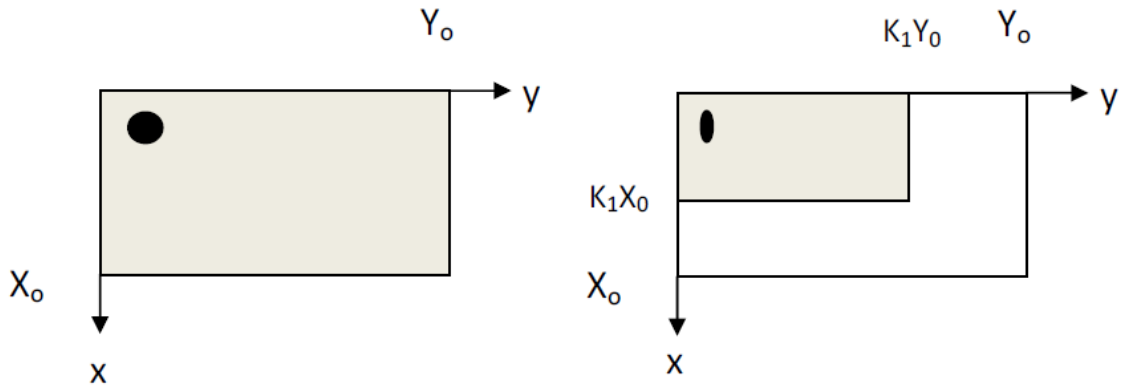


Imagen 14

4. Rotación

Consiste en girar la imagen un ángulo definido. Debemos tener en cuenta que con este proceso la imagen puede cambiar de tamaño respecto a la original.

Primero deducimos la relación entre las posiciones iniciales y finales:

$$\begin{aligned} y'_0 &= y_0 \cos\alpha + x_0 \sin\alpha \\ x'_0 &= x_0 \cos\alpha + y_0 \sin\alpha \end{aligned} \qquad \text{(Ecuación 59)}$$

A partir de estas ecuaciones obtenemos la matriz de rotación:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \qquad \text{(Ecuación 60)}$$

Si los nuevos ejes giran en sentido horario, la imagen gira en sentido contrario a las agujas del reloj y viceversa. Para girar una imagen en sentido horario α debe ser negativo.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \qquad \text{(Ecuación 61)}$$

- Rotación simple 90º

Al rotar una imagen 90º, se produce un reacomodo de píxeles. Por este motivo si tenemos una imagen de M filas por N columnas, al rotar en sentido contrario a las agujas del reloj respecto al centro de la imagen, se producirá una transposición y la imagen pasara a ser de N filas por M columnas.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \left[\begin{pmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}\right] \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \xrightarrow{\alpha = +90º} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad \text{(Ecuación 62)}$$

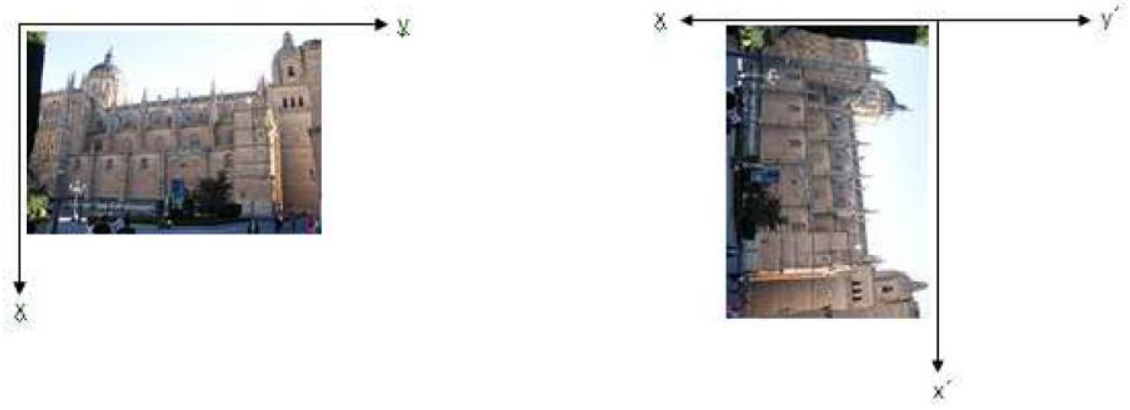


Imagen 15

- Rotación 180º

Al rotar una imagen 180º obtenemos una imagen con las mismas dimensiones que la original pero en sentido contrario. Usamos la siguiente matriz de transformación:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \left[\begin{pmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}\right] \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \xrightarrow{\alpha = +180º} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad \text{(Ecuación 63)}$$

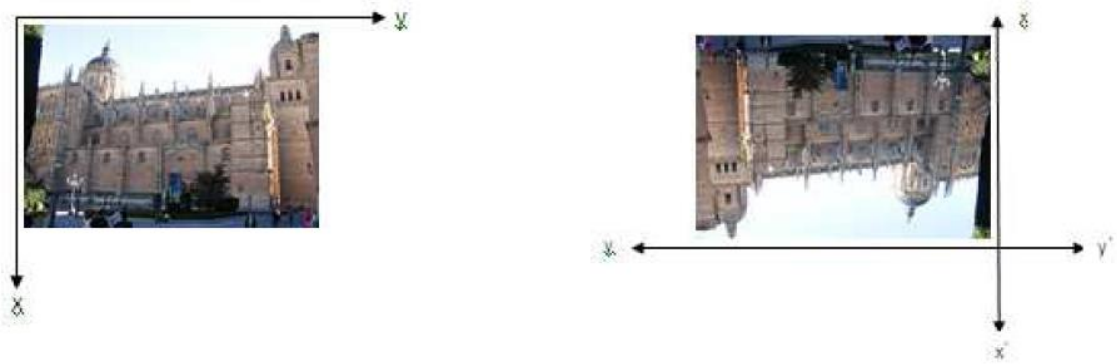


Imagen 16

Nótese que no es igual que la simetría espejo vertical.

• Rotación libre directa

La usamos para girar una imagen un ángulo α determinado. Pueden quedarse píxeles sin asignar un valor.

• Rotación libre inversa

En vez de buscar la posición más cercana (inventarse la posición) es posible inventarse el valor del píxel haciéndolo al revés. Las posiciones son exactas pero los brillos son interpolados.

5. Inclinación

La inclinación transforma una región rectangular en un romboide. Sirve para simular una perspectiva.

2.2.1.8 Convolución

1. Unidimensional

La convolución de dos secuencias x[n] e y[n], es otra secuencia z[n] dada por la expresión:

$$z[n] = x[n] * y[n] = \sum_{\substack{n'=-\infty \\ -\infty < n < +\infty}}^{\infty} x[n']\, y[n-n'] \qquad \text{(Ecuación 64)}$$

Si el tamaño de x[n] es N_1 y el de y[n] es N_2, el tamaño de z[n] es $N_1 + N_2 + 1$

$$z[n] = x[n] * y[n] = y[n] * x[n] \qquad \text{(Ecuación 65)}$$

Para el desplazamiento se suele elegir la secuencia más corta.

2. Bidimensional

La convolución de dos secuencias bidimensionales x[m, n] e y[m, n] es otra secuencia bidimensional z[m, n] dada por la siguiente expresión:

$$z[m,n] = x[m,n] * y[m,n] = \sum_{m'=-\infty}^{\infty} \sum_{n'=-\infty}^{\infty} x[m',n']y[(m-m'),(n-n')] \quad -\infty < m < +\infty \quad -\infty < n < +\infty \qquad \text{(Ecuación 66)}$$

Si el tamaño de x[m, n] es $[M1, N_1]$ y el de y[m, n] es $[M2, N_2]$, el tamaño de z[m, n] es $[(M1 + M2 - 1), (N_1 + N_2 - 1)]$.

2.2.1.9 Transformadas de la imagen

1. Transformada de Fourier unidimensional

La transformada de una secuencia x[n] es:

$$X(\Omega) = \sum_{-\infty}^{\infty} x[n]\, e^{-j\Omega n} \qquad \text{(Ecuación 67)}$$

Donde $\Omega = 2\pi f/N$, siendo N el número de valores de la secuencia x[n] para la variable Ω de periodo N.

La transformada inversa será:

$$x[n] = \frac{1}{2\pi} \int_{2\pi}^{0} X(\Omega)\, e^{j\Omega n}\, d\Omega \qquad \text{(Ecuación 68)}$$

2. Transformada de Fourier bidimensional

La transformada de Fourier de una imagen x[m, n] es una función continua:

$$X(\Omega_1, \Omega_2) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} x[m,n] e^{-j\Omega_1 m} e^{-j\Omega_2 n} \qquad \text{(Ecuación 69)}$$

Siendo $\Omega_1 = 2\pi F_1/N$ y $\Omega_2 = 2\pi F_2/N$.

La transformada inversa será:

$$X(\Omega_1, \Omega_2) = \frac{1}{(2\pi)^2} \int_{2\pi}^{0} \int_{2\pi}^{0} X(\Omega_1, \Omega_2) e^{j\Omega n} \, d\Omega_1 d\Omega_2 \qquad \text{(Ecuación 70)}$$

3. Transformada discreta de Fourier unidimensional

La transformada de Fourier de una secuencia discreta genera una función continua, la transformada discreta de Fourier genera una secuencia:

$$X[K] = \sum_{n=0}^{N-1} x[n] e^{-j\pi \frac{k}{N} n} \;, \quad k = 0,1,2 \ldots (N-1) \qquad \text{(Ecuación 71)}$$

La transformada inversa será:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j2\pi \frac{k}{N} n} \;, \quad n = 0,1,2 \ldots \qquad \text{(Ecuación 72)}$$

4. Transformada discreta de Fourier bidimensional

Para la secuencia bidimensional, x[m, n] su transformada discreta de Fourier es:

$$X[k,l] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} x[m,n] e^{-j2\pi \frac{k}{M} m} e^{-j2\pi \frac{l}{N} n} \;, \quad \begin{matrix} 0 < k < (M-1) \\ 0 < l < (N-1) \end{matrix} \qquad \text{(Ecuación 73)}$$

Su transformada inversa será:

$$X[m,n] = \frac{1}{MN} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} x[k,l] e^{j2\pi \frac{k}{M} m} e^{j2\pi \frac{l}{N} n} \qquad \text{(Ecuación 74)}$$

5. Propiedades de la transformada de Fourier

- Linealidad

El espectro de la suma lineal de imágenes es igual a la suma lineal de los espectros.

Teniendo:

$$TF\{x_1[m,n]\} = X_1[k,l] \qquad \text{(Ecuación 75)}$$

$$TF\{x_2[m,n]\} = X_2[k,l] \qquad \text{(Ecuación 76)}$$

Si:

$$x[m,n] = a x_1[m,n] + b x_2[m,n] \qquad \text{(Ecuación 77)}$$

Se cumple que:

$$X[k,l] = aX_1[k,l] + bx_2[k,l] \quad \text{(Ecuación 78)}$$

- Translación

Teniendo:

$$TF\{x_1[m,n]\} = X_1[k,l] \quad \text{(Ecuación 79)}$$

Si se desplaza x_1:

$$x_2[m,n] = x_1[(m-m_0),(n-n_0)] \quad \text{(Ecuación 80)}$$

Se cumple que:

$$X_2[k,l] = X_1[k,l]e^{j2\pi m_0}e^{j2\pi n_0} \quad \text{(Ecuación 81)}$$

- Separabilidad

Mediante esta propiedad podemos calcular la transformada discreta de Fourier de una función bidimensional como una multiplicación de dos transformadas de Fourier discretas, calculando primero una TFD sobre la variable de uno de los ejes y multiplicar el resultado por la TFD de la segunda imagen.

$$T(x) = T(x_1) * T(x_2) \quad \text{(Ecuación 82)}$$

- Periodicidad

La transformada de una secuencia x[m,n] de tamaño MxN es una función continua y periódica. La transformada discreta de Fourier son muestras de la anterior, pero sólo se considera un periodo. El resto es repetido. Es importante tener esto en cuenta cuando hay que hacer la convolución entre dos espectros.

- Simetría conjugada

La transformada de Fourier cumple:

$$\begin{aligned} X[k,l] &= \\ &= X^*[-k,-l] \end{aligned} \begin{cases} X[k,l] = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x[m,n]e^{-j2\pi\frac{k}{M}m}e^{-j2\pi\frac{l}{N}n} \\ X[-k,-l] = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x[m,n]e^{j2\pi\frac{k}{M}m}e^{j2\pi\frac{l}{N}n} \end{cases} \quad \text{(Ecuación 83)}$$

- Centrado de la transformada

Los valores significativos suelen estar en los extremos. Al multiplicar la imagen por $(-1)^{m+n}$ antes de la transformación, implica el centrado de la transformada.

Si:

$$TF\{f[m,n]\} = F(k,l) \quad \text{(Ecuación 84)}$$

Entonces:

$$TF\{f[m,n](-1)^{m+n}\} = F(k-\frac{M}{2}, l-\frac{N}{2}) \quad \text{(Ecuación 85)}$$

- Rotación

Si rotamos una imagen, su espectro también rota.

- Convolución

Teniendo:

$$TF\{x_1[m,n]\} = X_1[k,l] \quad \text{(Ecuación 86)}$$

$$TF\{x_2[m,n]\} = X_2[k,l] \quad \text{(Ecuación 87)}$$

Se cumple:

$$x_1[m,n]^* x_2[m,n] = X_1[k,l]X_2[k,l] \quad \text{(Ecuación 88)}$$

- Producto

Teniendo:

$$TF\{x_1[m,n]\} = X_1[k,l] \quad \text{(Ecuación 89)}$$

$$TF\{x_2[m,n]\} = X_2[k,l] \quad \text{(Ecuación 90)}$$

Se cumple:

$$X_1[k,l]X_2[k,l] = x_1[m,n]^* x_2[m,n] \quad \text{(Ecuación 91)}$$

Esta convolución ha de ser circular porque el espectro es periódico, no limitado. El producto elemento a elemento coincide con la convolución circular.

- Conservación de la energía

$$\sum\sum |x|^2 = \sum\sum |X|^2 \quad \text{(Ecuación 92)}$$

$$\sum\sum xx^* = \sum\sum XX^* \quad \text{(Ecuación 93)}$$

2.2.1.10 Operaciones área de filtros

En las operaciones de punto, cada píxel de salida dependía sólo de un píxel de entrada, no teníamos en cuenta la relación con entre los píxeles vecinos.

En las operaciones locales o de área el resultado de la transformación depende de los valores de los píxeles vecinos al considerado. Dependiendo de la valoración del entero así es el tipo de filtro. Lo normal es que la zona a considerar se centre sobre el píxel y además sea cuadrada.

El tamaño de la imagen y la posición de los píxeles se mantiene pero cambia su valor en función de las características de la zona de influencia.

Matemáticamente podemos verlo en las siguientes funciones:

- En operaciones de punto:

$$v[m,n] = f(u[m,n]) \text{ ó } v[m,n] = f(N_1[m,n], N_2[m,n]) \quad \text{(Ecuación 94)}$$

- En operaciones de área:

$$v[m,n] = f(u[m-k,n-k]) \dots u[m,n] \dots + u_1[m+k,n+k] \quad \text{(Ecuación 95)}$$

Un tipo de operaciones de área de interés son las convoluciones discretas, que son transformaciones en las que el valor del píxel resultante es una combinación lineal de los valores de los píxeles vecinos en la imagen. La matriz de los coeficientes de esta combinación lineal la denominaremos máscara o núcleo de convolución.

Matemáticamente lo podríamos reflejar así:

- Sea a una máscara de convolución.
- La imagen de entrada sea u.
- La imagen de salida sea v.
- Algoritmo:

  Calcular la convolución $v = a \emptyset u$ para cada píxel (m,n) de la imagen u, haciendo:

$$v[m,n] = \sum\sum a[k,l]u[m-k,n-l] \quad \text{(Ecuación 96)}$$

Uno de los problemas que nos surgen a la hora de convolucionar una imagen con una máscara es qué hacer con los bordes. Esta situación puede ser resuelta de estos cuatro modos:

1. Asignar un cero en el resultado a los píxeles donde no cabe la máscara.

2. Suponer que los píxeles que se salen tienen un valor constante (normalmente cero).

3. Modificar el valor en los píxeles que no caben (variar el multiplicador).

4. Suponer que la imagen se pliega por los extremos.

Según apliquemos los distintos operadores de convolución es posible obtener diferentes efectos en nuestra imagen.

- Suavizado o difuminado de la imagen: reducir contrastes grandes en la imagen.

- Perfilado: resaltar los contrastes, es la operación opuesta al suavizado.

- Bordes: detectar zonas de variación de la imagen.

- Detección de cierto tipo de características tales como esquinas, segmentos…

El uso de las dos primeras es más habitual en restauración y mejora de imágenes mientras que las dos últimas suelen usarse más en análisis de imágenes.

Tipos de filtros:

1. Filtros suavizantes lineales

   Cumplen una doble acción, por un lado difuminan la imagen comportándose como un filtro paso bajo y por otro disminuyen el ruido en imágenes ruidosas.

   Las máscaras de estos filtros se caracterizan por tener todos sus coeficientes positivos, no como los filtros paso alto que presentan coeficientes positivos y negativos.

   - Filtro de media

   El operador de suavizado más simple es la convolución de media (media aritmética). Tiene como parámetros el ancho y el alto en la que se aplica (w y h), así como una posición ancla. Normalmente w y h son impares y el ancla es el píxel central. La máscara es una simple tabla de unos de tamaño w por h.

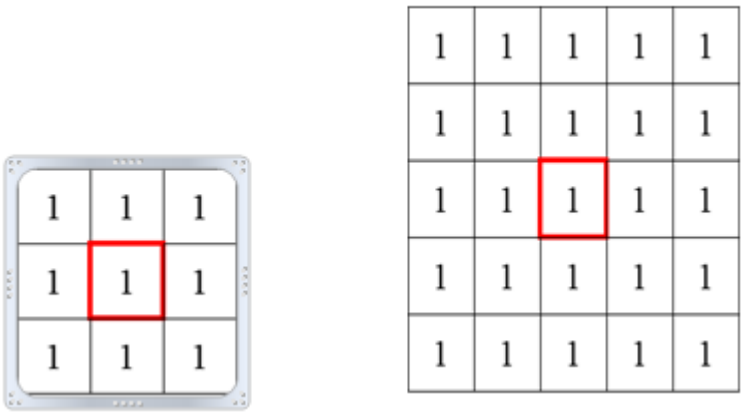


Imagen 17. Máscara 3x3 y 5x5

La descripción práctica del proceso sería:

1º Superponemos la máscara sobre cada punto de la imagen.

2º Sumamos los productos de los coeficientes de la máscara y los valores de los puntos de la imagen situados bajo esta.

El valor de brillo de cada píxel se reemplaza por una media ponderada o no del brillo de dicho píxel y el de los píxeles vecinos.

Para forma y tamaño de máscara dados, si la media ponderada de puntos da mayor peso al valor del punto central que a los puntos vecinos implica que tendrá un menor efecto de difuminado. Por otro lado cuanto mayor sea el tamaño de la máscara, mayor será el efecto de difuminado de la imagen.

Sus ventajas son que es sencillo y rápido de aplicar, es fácil de definir un comportamiento para los píxeles de los bordes: tomar la media de los píxeles que quepan y por último, el operador de media es separable.

También podemos comentar sus ventajas respecto del nivel de ruido. La potencia del ruido se reduce por un factor igual al número de píxeles que abarca la máscara. Cuanto mayor sea el tamaño de la máscara, mejor será la relación señal a rudo. Si el tamaño de la máscara es demasiado grande, puede dar lugar a una imagen muy borrosa y desenfocada.

- Suavizado direccional

Como el filtro de media suaviza mucho los bordes, usamos como alternativa el suavizado direccional, el cual calcula las medias de regiones longitudinales a lo largo de todas las direcciones posibles y de todas las medias direccionales posibles, elegiremos la más parecida al valor del punto en la imagen de entrada.

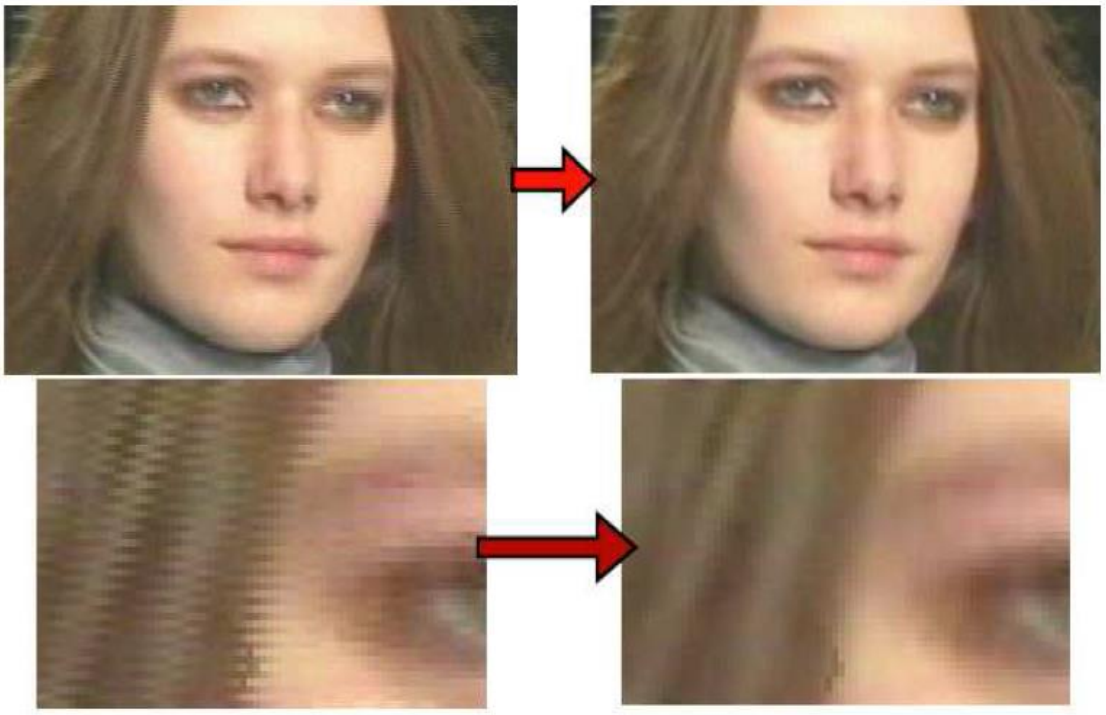


Imagen 18. Suavizado direccional

- Suavizado gaussiano

  Usamos este tipo de suavizado debido a que cuando se aplica la media con tamaños grandes se obtienen resultados artificiosos (a menudo indeseables). Consiste en aplicar una media ponderada, donde los pesos toman la forma de una campana gaussiana.

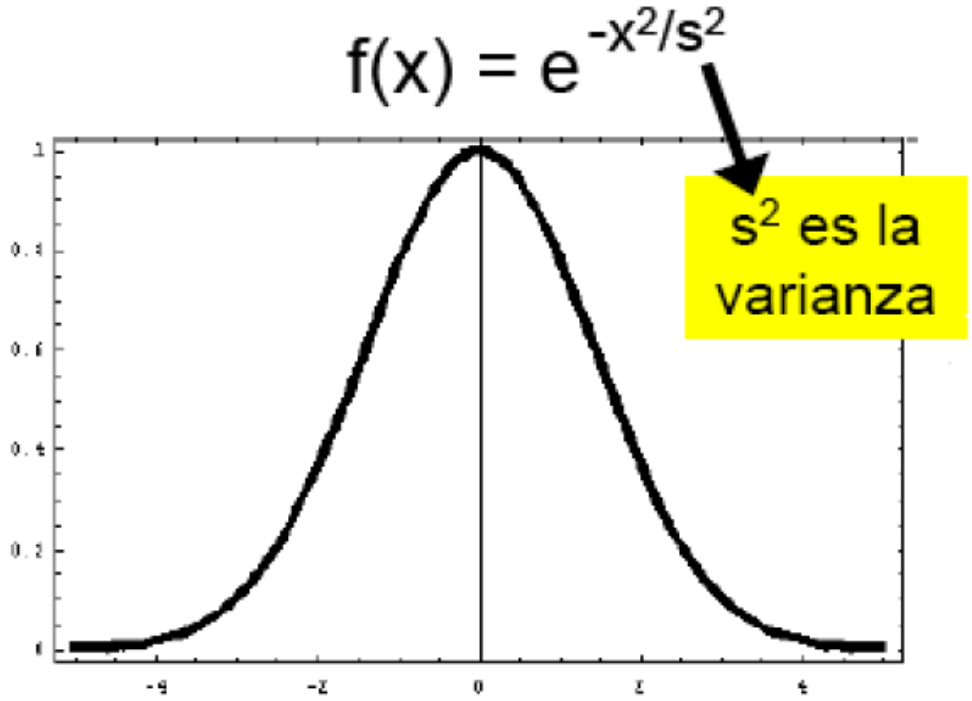


Imagen 19. Suavizado gaussiano

La varianza; s^2, indica el nivel de suavizado. Una varianza grande implica una campana más estrecha y más suavizado mientras que una varianza pequeña da como resultado una campana más estrecha y menor suavizado. Este se mide en píxeles.

Para el cálculo de la máscara gaussiana de una dimensión calculamos la función, discretizamos el rango, discretizamos el valor y calculamos el multiplicador. De forma más rápida usamos el triángulo de pascal ya que sus filas forman discretizaciones de la campana de Gauss.

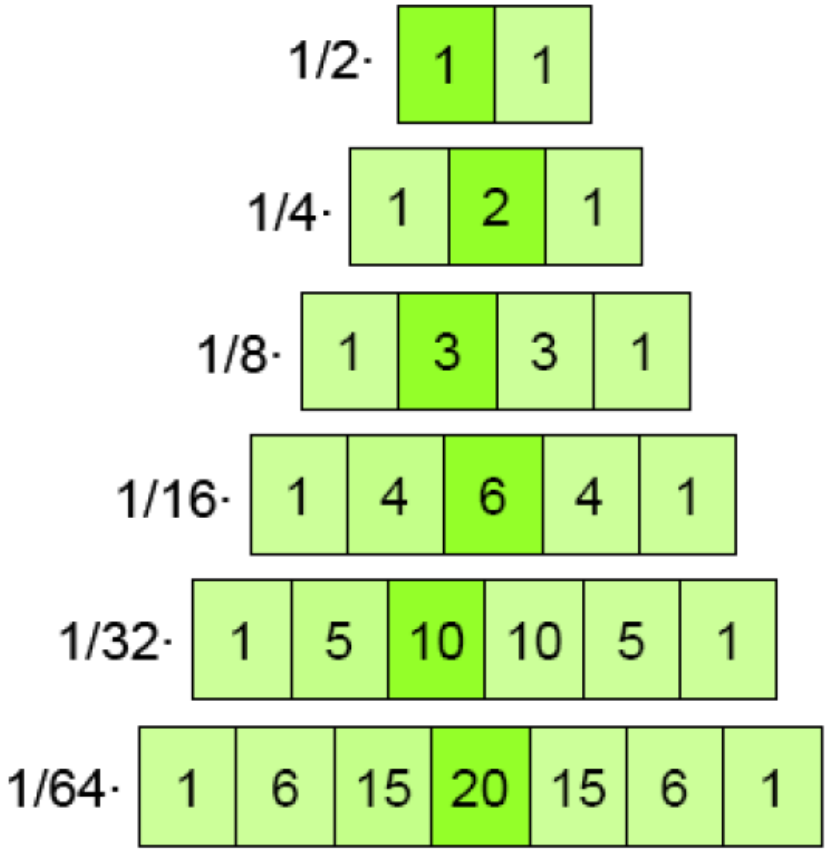


Imagen 20. Campana de Gauss

Normalmente, el suavizado gaussiano se aplica en dos dimensiones. Los pesos de la máscara dependen de la distancia al píxel central.

Campana de Gauss de dos dimensiones:

$$f(x,y) = e^{-(x^2+y^2)/s^2} \qquad \text{(Ecuación 97)}$$

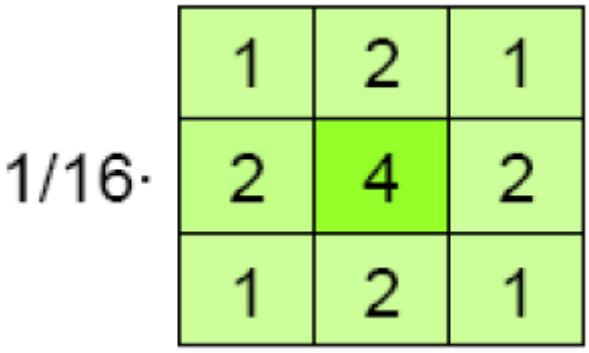


Imagen 21

Podemos simplificando teniendo en cuenta una propiedad interesante y es que el filtro es separable. Se puede obtener como resultado un suavizado en dos dimensiones aplicando dos máscaras gaussianas bidimensionales, una horizontal y otra vertical.

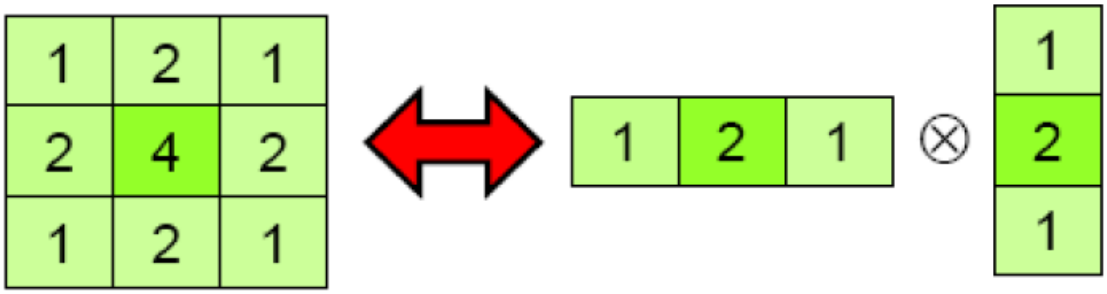


Imagen 22

En comparación con el filtro de media obtenemos como resultados que para conseguir un mismo grado de suavizado, la máscara gaussiana debe ser de mayor tamaño y que el efecto del suavizado gaussiano es más habitual (más habitual al desenfoque) que la media lo que implica que suele ser más habitual en procesamiento y análisis de imágenes. Ponemos como ejemplos el emborronamiento de las caras para proteger el anonimato de las personas y el resaltado de objetos de interés.

2. Filtros suavizantes no lineales

En las convoluciones, f es una combinación lineal cualquiera pero también puede ser interesante usar otras funciones no lineales.

- Media geométrica

$$v[m,n] = \sqrt[4]{u[m-1,n-1]\cdot u[m,n-1]\cdot u[m-1,n]\cdot u[m,n]} \quad \text{(Ecuación 98)}$$

Imagen 23

Aunque existen muchas (en teoría infinitas) posibles transformaciones no lineales, en la práctica no todas son útiles e interesantes. Las que más se usan son: máximo, mínimo y media.

- Filtro de máximo

$$v[m,n] = max\{u[m-k,n-k], \dots, u[m,n], \dots, u[m+k,n+k]\} \quad \text{(Ecuación 99)}$$

Donde k es el radio y el tamaño o apertura es $2k+1$.

El resultado de aplicar este filtro es un cierto efecto de difuminación y aclaramiento de la imagen. Desaparecen los detalles más oscuros.

Si el tamaño es grande pueden ocurrir dos efectos. En primer lugar puede darse un efecto de cuadriculado, como el máximo se aplica en una zona cuadrada, los píxeles muy claros generan un cuadrado uniforme alrededor. En segundo lugar pueden aparecer colores falsos ya que al aplicarlo en los tres canales (R, G, B) independientemente, el máximo en los tres puede no corresponder a un color presente en la imagen original.

- Filtro de mínimo

$$v[m,n] = min\{u[m-k,n-k], \dots, u[m,n], \dots, u[m+k,n+k]\} \quad \text{(Ecuación 100)}$$

Donde k es el radio y el tamaño o apertura es $2k+1$.

Su efecto es parecido al máximo, pero tomando los valores menores (los más oscuros).

Para minimizar sus inconvenientes se podría aplicar el máximo/mínimo a una zona circular para evitar el efecto de cuadriculado y se podría tomar el máximo de las sumas de R+G+B para evitar la aparición de colores falsos.

- Filtro de mediana

  Esta es una técnica alternativa cuando el objeto a alcanzar es más la reducción del ruido que el difuminado de la imagen.

  Consiste en sustituir el valor del nivel de gris de cada punto de la imagen de entrada por el valor mediano de los puntos que están incluidos dentro de una ventana entorno.

$$v[m,n] = mediana\{u[m-k, n-1]\}, para(k,l) \sum w \qquad \text{(Ecuación 101)}$$

  Propiedades del filtro de mediana:

- Este filtro no es lineal

$$mediana\{ax(m,n) + by(m,n)\} \neq a \cdot mediana\{x(m,n)\} + b \cdot mediana\{y(m,n)\} \qquad \text{(Ecuación 102)}$$

- La supresión del ruido la realiza bien si es binario (compuesto de fuertes componentes en forma de pico).

- Los resultados son pobres si el ruido tiene distribución gaussiana o si el número de píxeles de la ventana afectados por el ruido es más de la mitad.

  La mediana produce un efecto de suavizado, aunque es más abrupto en los bordes que la media y el suavizado gaussiano. Pero el verdadero interés es la eliminación de ruido puntual.

3. Filtros agudizadores

   Los filtros agudizadores se usan para agudizar los detalles finos (altas frecuencias) de la imagen. El objetivo de este realce es contrarrestar las degradaciones que dan a la imagen aspecto desenfocado, bien sea por error o por efecto natural del modo de adquisición de la imagen. Provocan el efecto contrario a los filtros de suavizado.

   Estos filtros destacan las variaciones en la imagen (bordes). Un borde es la frontera entre dos regiones con propiedades de nivel de gris relativamente distintas.

   En el apartado de detección de bordes del tema de análisis de imágenes, hablaremos más concretamente de los operadores usados para su detección.

3. Análisis de imágenes

Definimos el análisis de imágenes como el estudio de sus características con el fin de extraer información adicional contenida en las imágenes que a simple vista no es evidente. Esta información puede ser: medidas sobre la imagen, cambios abruptos de niveles de brillo, vectores de movimiento, reconocimiento de formas…

Está compuesto por toda esa serie de procesos que permiten extraer la información de una imagen, los podemos resumir de la siguiente manera:

1. Extracción de las características globales de la imagen
   - Detección de bordes, líneas y esquinas.
   - Texturas.
   - Detección de movimiento.
2. Segmentación de imágenes
   - Mediante umbralización.
   - Mediante extracción de contornos.
   - Orientada a regiones.
3. Transformaciones morfológicas.
4. Representación de contornos y regiones mediante información extraída de la imagen segmentada.

3.1 Extracción de las características de una imagen

Dependiendo del medio de extracción las clasificaremos en dos familias diferentes:

1. Características espaciales de la imagen.
2. Características de transformadas.

3.1.1 Características espaciales de la imagen

Este tipo de características pueden extraerse a partir de los niveles de gris que representa el objeto en la imagen (características de amplitud) o por medio de la distribución espacial de píxeles del objeto en función del resto de los de la imagen (características del histograma).

3.1.1.1 Características de amplitud

Se caracteriza el objeto dentro de la imagen mediante sus características físicas. Podemos tomar como ejemplo las imágenes adquiridas por ratos X donde la amplitud de los niveles de gris, representa la absorción de las diferentes partes del cuerpo, lo que permite distinguir los tejidos de las partes óseas, los tejidos dañados de los sanos la intrusión de objetos externos al cuerpo humano o la formación de tumores.

Un objeto caracterizado por su amplitud puede ser fácilmente extraído de la imagen determinando un umbral de nivel de gris para discernir el fondo del objeto y poder así separarlos.

3.1.12 Características del histograma

Como hemos comentado anteriormente, el histograma representa el número de píxeles en función de sus niveles de gris, por ello nos basamos en la observación del histograma para discernir los diferentes objetos en la imagen.

Para obtener estas características vamos a dividir cada valor de brillo del histograma por el número total de píxeles que poseen dicho brillo obteniendo así el histograma normalizado representando así la probabilidad de que los diferentes niveles aparezcan en la imagen. Definimos esta probabilidad como:

$$Prob[u = x] = \frac{numero\ de\ píxeles\ con\ nivel\ de\ gris\ x}{número\ de\ píxeles\ totales\ en\ la\ región}, \quad x = 0,1 \dots L-1 \qquad \text{(Ecuación 103)}$$

Donde el valor de L-1 representa el nivel de gris máximo posible y viene dado por el máximo valor de cuantificación asignado a la escala de grises.

A partir del histograma obtenemos características de la variable aleatoria, como son la dispersión, la varianza, la desviación típica, la media y dos muy importantes para la evaluación de la imagen como son la mediana y la moda.

El proceso sería el siguiente: se elige el tamaño de la ventana a analizar para la aplicación del cálculo y dependiendo del efecto deseado, se sustituye el valor de cada píxel por el valor de la media, moda o varianza de la ventana centrada en él, obteniendo así otra imagen con el efecto deseado o el valor de la característica extraída.

Es importante indicar que el histograma es utilizado para umbralizar la imagen, extrayendo primero un valor umbral para posteriormente binarizar la imagen en dos tipos de gris (blanco y negro) para segmentar el fondo del objeto deseado.

3.1.2 Características de la transformada

Son aquellas características obtenidas a partir de la transformada de una determinada imagen. Teniendo en cuenta que en muchas ocasiones una transformación a un determinado dominio puede proporcionar gran información acerca de la imagen.

Como la imagen es una función discreta, usamos la transformada discreta de Fourier (descrita en apartados anteriores) para obtener estas características. A partir de dicha transformada podemos calcular las respuestas en frecuencia que presentan determinados objetos dentro de la imagen. Genéricamente las altas frecuencias se emplean para detectar contornos y las líneas aisladas que aparecen en la imagen transformada aportan información sobre la orientación del objeto en la imagen.

La utilidad principal de estas transformadas es identificar objetos y orientaciones en una imagen desconocida conociendo previamente las transformadas de Fourier de ciertas figuras como cuadrados, rectángulos, etc.

3.1.3 Detección de bordes, texturas y movimientos

Los bordes se definen como cambios locales significativos de luminancia en la imagen, que generalmente están asociados a una discontinuidad.

Constituyen una de las características más importantes que se pueden encontrar en una imagen ya que delimitan los objetos que en ella se encuentran. Existen michos tipos de bordes diferentes, como los debidos a las sombras, a la variación de la reflectancia de los objetos, a discontinuidades, etc. Que luego habrá que relacionar con las condiciones específicas de la escena.

Atendiendo a su localización, podemos clasificarlos en:

- Bordes con transiciones abruptas, en los que hay grandes cambios de luminancia entre píxeles adyacentes, por lo que serán una serie de píxeles concretos.

- Bordes con transiciones progresivas, en los que los cambios de luminancia son progresivos y la variación entre píxeles adyacentes no es grande, localizándose así en una franja más amplia.

Otra posible clasificación puede ser, según el tipo de discontinuidad que representan:

- Bordes de salto, donde la luminancia cambia de un valor a un lado y a otro de la discontinuidad.

- Bordes de línea, donde la luminancia cambia volviendo en poco espacio a su valor inicial.

Las técnicas empleadas en la detección de bordes tienen por objeto localizar en la imagen los píxeles en los que se produce una variación de luminancia, empleando operadores derivativos. Estos operadores son básicamente dos: operadores de primera derivada o Gradiente y operadores de segunda derivada o Laplacianos. Los primeros buscan grandes picos en la variación mientras que los segundos buscan pasos de respuesta positiva a negativa y viceversa; es decir, el paso por cero.

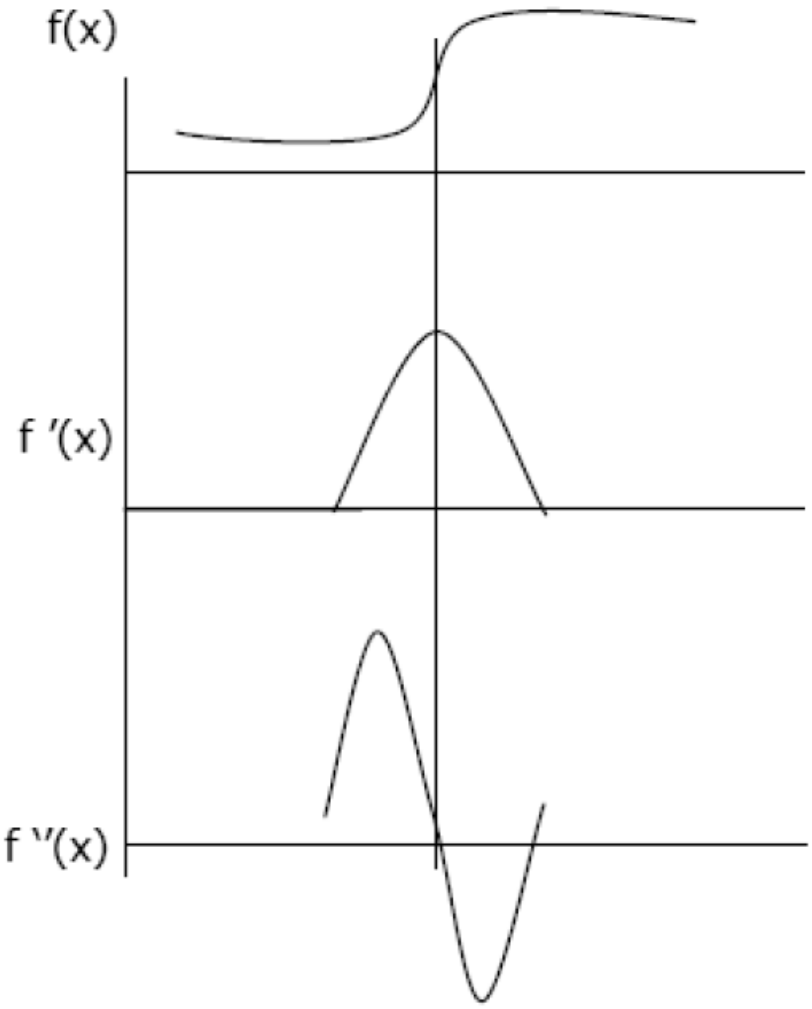


Imagen 24. Primera y segunda derivada unidimensional de f

3.1.3.1 Operadores tipo gradiente

Como hemos comentado la detección de bordes se realiza detectando los cambios locales significativos de luminancia. Estos cambios están asociados con un pico local de la primera derivada (en una dimensión). El equivalente en una imagen bidimensional es el gradiente. Localizando los puntos donde el gradiente es elevado, se sabrá donde se han producido las máximas variaciones de luminancia y por tanto donde estarán los bordes. Definimos el operador gradiente aplicado sobre una imagen como:

$$\nabla f(x,y) = \left[G_x, G_y\right] = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right] \qquad \text{(Ecuación 104)}$$

Esta ecuación indica que el gradiente de un punto (píxel) queda determinado conociendo las derivadas parciales en dos direcciones ortogonales, ejes x e y. Su módulo y dirección vendrán dados por:

$$|\nabla f| = \sqrt{G_x^2 + G_y^2} \quad \text{(Ecuación 105)}$$

$$\angle \nabla f = tang^{-1}\left(\frac{G_y}{Gx}\right) \quad \text{(Ecuación 106)}$$

La dirección del gradiente es perpendicular al borde. Para evitar el cálculo computacional, el módulo se suele aproximar a:

$$|\nabla f| = |G_x| + |G_y| \quad \text{(Ecuación 107)}$$

Como en la práctica manejaremos imágenes que son secuencias discretas bidireccionales, aproximamos el concepto de derivada a espacios discretos. Esta expresión es la diferencia finita entre píxeles vecinos. La expresión del gradiente será:

$$\nabla f(x,y) = [G_x, G_y] = \left[\frac{\Delta f}{\Delta x}, \frac{\Delta f}{\Delta y}\right] \quad \text{(Ecuación 108)}$$

Con lo que:

$$G_x = \frac{\Delta f}{\Delta x} = f(x,y) - f(x+1,y) \quad \text{(Ecuación 109)}$$

$$G_y = \frac{\Delta f}{\Delta y} = f(x,y) - f(x,y+1) \quad \text{(Ecuación 110)}$$

Esto mismo se puede expresar a partir de sus máscaras de convolución, una que realiza la derivada en la dirección x, y la otra en la dirección y:

$$H_x = \begin{bmatrix} 1 & -1 \\ 0 & 0 \end{bmatrix} \quad \text{(Ecuación 111)}$$

$$H_y = \begin{bmatrix} 1 & 0 \\ -1 & 0 \end{bmatrix} \quad \text{(Ecuación 112)}$$

Estas máscaras no se suelen usar debido a su sensibilidad frente al ruido por lo que se han desarrollado otros operadores con máscaras de mayor orden.

La forma de calcular la imagen gradiente se representa en la siguiente figura:

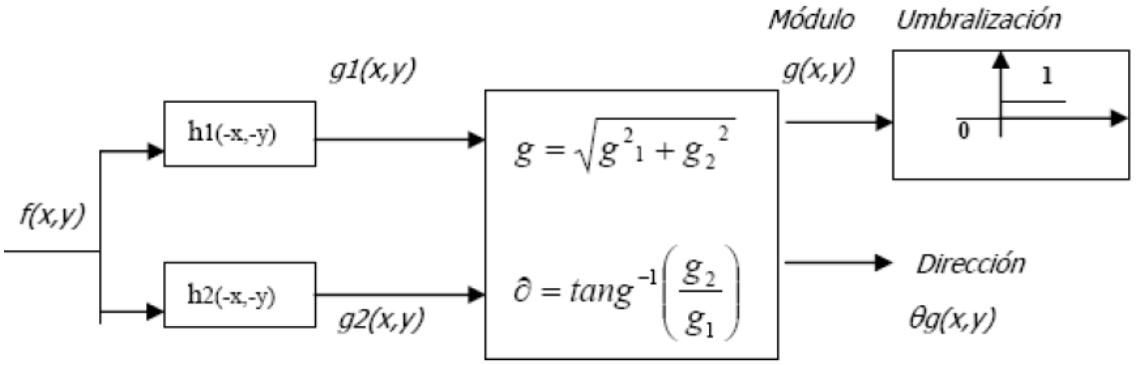


Imagen 25. Esquema de cálculo del Gradiente

El resultado que obtenemos es una imagen g(x,y) cuyos puntos equivalen al módulo del gradiente de la imagen original. Dada esta imagen se suele considerar que un píxel forma parte de un borde si su valor se encuentra entre el 10% de los valores más altos. El valor umbral a partir del cual un píxel se considera borde es calculado mediante la observación del histograma de la imagen gradiente. El resultado es una imagen binaria (Umbralizada) I(x,y), que representará los bordes que aparecen en la imagen original y que expresamos así:

$$I(x,y) = \begin{cases} 1 & si\ g(x,y) > umbral \\ 0 & en\ otro\ caso \end{cases} \quad \text{(Ecuación 113)}$$

3.1.3.1.1 Operador de Roberts

Es uno de los más antiguos. Usa máscaras de tamaño 2x2:

$$H_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad \text{(Ecuación 114)}$$

$$H_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad \text{(Ecuación 115)}$$

De la misma manera que en el caso genérico, este operador usa muy pocos píxeles para aproximar el gradiente haciéndolo muy sensible al ruido. Esto cambiará en los operadores posteriores.

3.1.3.1.2 Operador de Prewitt

Este operador usa máscaras de 3x3:

$$H_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{(Ecuación 116)}$$

$$H_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \quad \text{(Ecuación 117)}$$

3.1.3.1.3 Operador de Sobel

Es muy parecido al anterior, con la salvedad de que le da más peso a los píxeles más cercanos al centro de las máscaras:

$$H_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{(Ecuación 118)}$$

$$H_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad \text{(Ecuación 119)}$$

3.1.3.1.4 Operadores tipo compás

Los operadores anteriores usan dos matrices para obtener el gradiente, cuyas direcciones de cálculo son perpendiculares y siempre en las direcciones x e y. Los operadores compás son operadores que miden el gradiente en cualquier dirección (en intervalos de 45º) y la única ventaja que presentan respecto a los otros es que para calcular el gradiente, sólo se necesita usar una matriz de una determinada dirección en vez de dos.

⇑ (N) $\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$ ⇐⇑ (NO) $\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix}$

⇐ (O) $\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$ ⇐⇓ (SO) $\begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix}$

⇓ (S) $\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$ ⇓⇒ (SE) $\begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$

⇒ (E) $\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$ ⇑⇒ (NE) $\begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix}$

Imagen 26. Operadores tipo compás

Estos operadores permiten calcular la variación de luminancia de los píxeles de una imagen en cualquier dirección. Para determinados casos son más efectivos que los convencionales dado que evitan el tener que calcular la dirección del gradiente y su módulo a partir de las componentes ortogonales calculadas, la desventaja es que para un punto determinado, hay que calcular el gradiente en todas las direcciones.

La forma de manejar estos operadores es la siguiente: se calculan los gradientes en las ocho direcciones posibles y se toma como valor del gradiente el valor del máximo gradiente direccional. Matemáticamente lo expresamos de la siguiente forma:

Se llama $g_k(x,y)$ al gradiente compás en la dirección $\theta = \frac{\pi}{2} + k\frac{\pi}{4}$, donde $k = 0, \dots, 7$. El gradiente en la posición (x,y) es:

$$g(x,y) = máx\{g_k(x,y)\} \quad \text{(Ecuación 120)}$$

3.1.3.2 Operadores tipo laplaciano

El laplaciano de una función bidimensional f(x,y), es el resultado de aplicar un operador derivativo de segundo orden. Lo podemos escribir así:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \quad \text{(Ecuación 121)}$$

De la misma manera que el operador gradiente, con el laplaciano realizmos la misma aproximación discreta de la segunda derivada, usando las ecuaciones en diferencias:

$$\frac{\partial^2 f}{\partial x^2} = \frac{G_x}{\partial x} = \frac{\partial(f(i+1,j) - f(i,j))}{\partial x} = f(i+2,j) - 2f(i+1,j) + f(i,j) \quad \text{(Ecuación 122)}$$

$$\frac{\partial^2 f}{\partial y^2} = \frac{G_y}{\partial y} = \frac{\partial(f(i,j+1) - f(i,j))}{\partial y} = f(i,j+2) - 2f(i,j+1) + f(i,j) \quad \text{(Ecuación 123)}$$

Las centramos en la posición (i,j):

$$\frac{\partial^2 f}{\partial x^2} = f(i+1,j) - 2f(i,j) + f(i-1,j) \quad \text{(Ecuación 124)}$$

$$\frac{\partial^2 f}{\partial y^2} = f(i,j+1) - 2f(i,j) + f(i,j-1) \quad \text{(Ecuación 125)}$$

Combinando estas dos ecuaciones en un único operador se obtendría la máscara 1 de la siguiente figura:

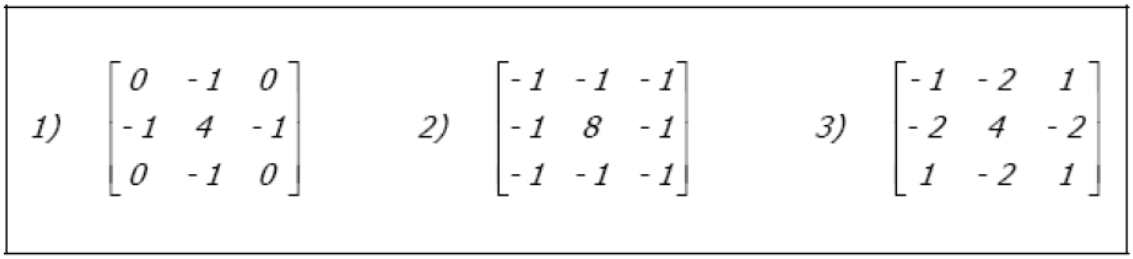


Imagen 27. Operadores laplacianos de tercer orden

En la ilustración anterior podemos observar que el valor central toma el valor positivo de la suma de todos los píxeles que le rodean, por lo que la suma de los coeficientes de la máscara es cero, por lo que si se aplica a una región de píxeles constante, el resultado será nulo.

Este detectaría un borde cuando la salida del operador realiza una transición por cero. Es un operador muy sensible al ruido por lo que no se utiliza como tal, sino como paso intermedio para el operador de Marr-Hildreth que se verá a continuación.

3.1.3.3 Operadores derivadas de gausianas

3.1.3.3.1 Laplaciana de gausiana

También llamado operador de Marr-Hildreth o de cruce por cero, parte del principio de que para detectar un borde de salto, la primera derivada de la función imagen debe tener un máximo en ese punto (que lo tienen en cuenta los operadores gradiente) y a la vez la segunda derivada (operador laplaciano) debe presentar un cruce por cero en ese mismo punto. Siendo mucho más fácil encontrar un cruce por cero que un máximo. Como el operador laplaciano es muy sensible al ruido y por ello es necesario suavizar previamente la imagen mediante un filtro para poder aplicar la segunda derivada de una forma robusta. El filtro empleado debe cumplir dos condiciones:

- El filtro debe ser local y tomar información únicamente de los puntos cercanos al que se va a analizar.

- Debe ser paso banda, para reducir el número de frecuencias en las que puede producirse el cambio.

La distribución que optimiza simultáneamente ambas condiciones es la gausiana. El filtro de suavizado gausiano viene dado por:

$$G(x,y) = ce^{\left[-\frac{x^2+y^2}{2\sigma^2}\right]} \qquad \text{(Ecuación 126)}$$

Donde c es una constante de normalización y σ es la desviación típica, x e y son las coordenadas de la imagen.

Primero se convoluciona la imagen con la Gausiana G(x,y) y a continuación se le aplica al resultado la Laplaciana:

$$H(x,y) = \nabla^2(G(x,y) * f(x,y)) \qquad \text{(Ecuación 127)}$$

Este es el operador Marr-Hildreth o LoG.

Operando, se obtiene que el filtro de convolución es de la siguiente manera:

$$h(x,y) = \nabla^2 G(x,y) = \frac{c}{\sigma^2}\left(\frac{x^2+y^2}{\sigma^2} - 2\right)e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)} \qquad \text{(Ecuación 128)}$$

Este proceso de obtención de la segunda derivada es muy robusto ya que el filtro gausiano elimina la influencia de los píxeles lejanos haciendo así que la Laplaciana realice una medida estable y eficiente de los cambios en la imagen.

La ventaja de este operador frente a los demás, es que se tiene en cuenta un mayor área de píxeles lejanos, influencia que viene marcada por σ.

La cuestión ahora sería cómo operar con la aproximación ofrecida por las máscaras. La primera operación es aplicar a la imagen un filtro gausiano, pasándole su correspondiente máscara y, a continuación, aplicarle a la imagen filtrada una de las máscaras Laplacianas.

A continuación, se van a ver unas máscaras que aproximan la forma gausiana, para distintas σ.

$\sigma = 0{,}391 pixels$

$$\begin{bmatrix} 1 & 4 & 1 \\ 4 & 12 & 4 \\ 1 & 4 & 1 \end{bmatrix}$$

$\sigma = 0{,}625 pixels$

$$\begin{bmatrix} 1 & 2 & 3 & 2 & 1 \\ 2 & 7 & 11 & 7 & 2 \\ 3 & 11 & 17 & 11 & 3 \\ 2 & 7 & 11 & 7 & 2 \\ 1 & 2 & 3 & 2 & 1 \end{bmatrix}$$

$\sigma = 1{,}0 pixels$ *

$$\begin{bmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 3 \\ 1 & 2 & 3 & 6 & 7 \\ 1 & 3 & 6 & 9 & 11 \\ 1 & 3 & 7 & 11 & 12 \end{bmatrix}$$

Imagen 28. Operadores de Marr-Hildreth

3.1.3.3.2 Detector de Canny

Se obtiene optimizando las siguientes condiciones:

- Error. Se deben detectar todos los bordes.
- Localización. La distancia entre el píxel obtenido como borde y el borde real debe ser lo más pequeña posible.

- Respuesta. No debe identificar varios píxeles como bordes cuando sólo exista uno.

Expresamos matemáticamente estas condiciones de la siguiente manera:

$$SNR = \frac{A\left|\int_{-W}^{0} f(x)dx\right|}{n_0\sqrt{\int_{-W}^{W} f^2(x)dx}} \quad \text{(Ecuación 129)}$$

$$Localización = \frac{A|f(0)|}{n_0\sqrt{\int_{-W}^{W} f^2(x)dx}} \quad \text{(Ecuación 130)}$$

$$Distancia = \pi\sqrt{\left(\frac{\int_{-\infty}^{\infty} f^2(x)dx}{\int_{-\infty}^{\infty} f'^2(x)dx}\right)} \quad \text{(Ecuación 131)}$$

Con el detector de Canny se busca la optimización del producto de la relación señal ruido por la localización, así como el cumplimiento de la tercera condición. El detector que cumple con lo anteriormente expuesto es la derivada primera de una Gausiana; es decir, un operador que es la combinación de un filtro Gausiano con una aproximación de gradiente.

3.1.3.4 Medida de la calidad de un detector de bordes

El comportamiento de todos los operadores vistos hasta ahora es similar siempre que en la imagen haya ausencia de ruido, las diferencias fundamentales de comportamiento entre unos y otros se presentan cuando se está en presencia de ruido.

Si llamamos n_0 al número total de píxeles que forman el borde y n_1 al número de píxeles que son erróneamente considerados como borde, debido a la presencia de ruido en la imagen, el error de detección es:

$$P = \frac{n_1}{n_0} \quad \text{(Ecuación 132)}$$

Por ejemplo, con el operador de Sobel se comete un error del 24%.

Otra forma diferente de medir el error de detección es la siguiente:

$$P = \frac{1}{máx(N_i, N_d)} \cdot \sum_{i=1}^{N_d} \frac{1}{1 + \alpha d_i^2} \quad \text{(Ecuación 133)}$$

Dónde:

- d_i = menor distancia del borde i-ésimo al borde ideal.

- $\propto$ = constante positiva.
- N_i = número ideal de píxeles de borde.
- N_d = número total de píxeles detectados.

3.1.3.4 Detección de líneas

La detección de líneas se puede considerar como una extensión de la detección de bordes ya que para ello se emplean cuatro máscaras para detectar líneas en distintas direcciones. Ejemplos de máscaras:

$$E-O \quad \begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix} \qquad NE-NO \quad \begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix} \qquad N-S \quad \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix} \qquad NO-NE \quad \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$$

Imagen 29. Detección de líneas

Cada máscara detecta las líneas en la dirección de los máximos valores, en el caso anterior la primera detectaría líneas horizontales, la segunda líneas que forman un ángulo de 45º y la tercera y la cuarta, líneas verticales y con ángulo de -45º respectivamente.

3.1.3.5 Detección de esquinas

Las esquinas que aparecen en una imagen pueden ser debidas a dos razones:

- A la intersección de dos bordes de un objeto.
- A una discontinuidad en los niveles de gris debido a la textura de los objetos.

Proponemos cuatro métodos para su detección:

1. El primero se basa en detectar los bordes de los objetos y buscar aquellos puntos en los que la curvatura del borde varíe bruscamente.

2. Método de Tomasi y Kanade consistente en determinar partes de la imagen que tienen un elevado gradiente horizontal y vertical al mismo tiempo. Se analiza el entorno de un punto con la siguiente matriz:

$$C = \begin{bmatrix} \sum E_x^2 & \sum E_x E_y \\ \sum E_x E_y & \sum E_y^2 \end{bmatrix} \qquad \text{(Ecuación 134)}$$

A partir de los autovalores y autovectores de la matriz se puede obtener la distribución de los niveles de gris dentro del entorno del punto.

3. Método de Kitchen y Rosendfeld, basado en la variación de la dirección del gradiente. La fórmula utilizada es el producto de la curvatura horizontal por el gradiente de la imagen que es la siguiente:

$$C = \frac{I_{xx}I_y^2 - 2I_{xy}I_xI_y + I_{yy}I_x^2}{I_x^2 + I_y^2} \qquad \text{(Ecuación 135)}$$

Donde I_x, I_y se obtienen a partir de la convolución de las derivadas unidimensionales de la gausiana G, G_x y G_y con la imagen I.

4. Método SUSAN, acrónimo de Smallest Univalue Segment Assimilating Nucleus, que parte de una imagen con varios niveles de gris.

Primero calculamos una tabla T que marcará el valor que se debe asignar a la diferencia de nivel de gris de dos píxeles. Una vez obtenida la tabla, se realiza un barrido sobre la imagen de izquierda a derecha y de arriba abajo.

El algoritmo de detección de esquinas acumulará los valores de la tabla precalculada para las diferencias de brillos entre el píxel central de la máscara x y cada uno de los píxeles tratados. Tras el cómputo de la suma de los valores de la tabla se comprobará si se ha superado un cierto umbral, en cuyo caso no existirá esquina, en caso contrario se pasará por un segundo filtro que asegure que sea un máximo local para asegurar que sea una esquina.

3.1.4 Textura

La superficie de algunos materiales presenta una apariencia homogénea aunque no tengan un nivel de gris o color constante.

Imagen 30. Ejemplos de textura

Las texturas se definen como patrones visuales homogéneos que se observan en cierto tipo de materiales como pueden ser madera, tela, piedras, pelo, etc. Obviamente dependen del nivel de resolución de la imagen ya que pueden presentar aspecto homogéneo a cierta distancia y sin embargo; de cerca, no presentarlo o tener una textura diferente.

La extracción de esta característica va a tener importancia en la segmentación de imágenes por texturas así como en la descripción de regiones. Podemos realizar el análisis de texturas mediante dos procedimientos:

- Análisis estadístico, en el cual se analizan los estadísticos de primer orden o superiores, de los niveles de gris o de otra propiedad de una determinada zona de la imagen.

- Análisis frecuencias, en el cual se realiza el estudio a partir de la transformada de Fourier, lo que se traduce en el dominio transformado como un valor elevado a esa frecuencia de repetición.

3.1.4.1 Análisis estadístico de texturas

Realiza el análisis de alguna propiedad para cada píxel de la imagen. El orden de los estadísticos depende del número de puntos que definen la textura.

3.1.4.1.1 Estadísticos de primer orden

Hemos visto como el histograma normalizado de una imagen podía modelarse como una función de probabilidad, que posee una serie de propiedades. La media es una estimación del nivel de gris de la textura, la desviación típica indica la dispersión respecto a la media, el tercer momento mide la asimetría del histograma, el apuntamiento indica cómo se reparte el histograma entre la parte central y los extremos, la entropía mide la uniformidad del histograma.

Estas características serán empleadas en la segmentación a partir de texturas ya que las caracterizan y estas a su vez determinan los objetos que se van a separar del fondo.

3.1.4.1.2 Estadísticos de segundo orden

Se han desarrollado debido a que los de primer orden no captan toda la información espacial, al estar basados en el histograma. Tenemos dos tipos: los obtenidos a partir de matrices de concurrencia y a partir de las diferencias.

3.1.4.1.2.1 Matrices de coocurrencia

Se obtienen calculando la probabilidad condicional $P\delta$, que nos indica la probabilidad de que dos propiedades aparezcan separadas por una distancia δ, que es un vector de coordenadas polares $\delta = (r, \theta)$.

Para el caso particular de los niveles de gris, la matriz de coocurrencia estaría formada por los elementos P_r, $\theta(i,j)$ describiendo la frecuencia de aparición de dos píxeles con niveles de gris i, j, en una determinada ventana, separados una distancia r en la dirección θ. Se suele limitar el cálculo a una serie de ángulos: 0º, 45º, 90º, 135º y al valor de $r = 1$ píxel.

3.1.4.1.2.2 Estadísticos de las diferencias

Se obtienen a partir de la distribución de probabilidad $P\delta(k)$ de los valores pertenecientes a los píxeles intermedios, entre los que están separados por una distancia δ. Se define de la siguiente forma:

$$P_\delta(k) = \sum_{i=1}^{L}\sum_{j=1}^{L} P_\delta(i,j)\ ,\ \ |i-j| = k \qquad \text{(Ecuación 136)}$$

A partir de ella obtendremos el segundo momento angular, el contraste, la entropía y la media.

3.1.4.2 Análisis frecuencial de texturas

Se realiza a partir de la transformada de Fourier de la imagen, teniendo en cuenta dicha transformada de la imagen $f(x,y)$ es $F(u,v)$, su módulo será:

$$P(u,v) = |F(u,v)|^2 \qquad \text{(Ecuación 137)}$$

Tomando coordenadas polares se obtienen dos distribuciones, la primera indicará el tamaño de la textura dominante y la segunda la dirección de esta:

$$P(r) = 2\sum_{\vartheta=0}^{\pi} P(r,\delta) \qquad \text{(Ecuación 138)}$$

$$P(\theta) = \sum_{r=0}^{L/2} P(r,\theta) \qquad \text{(Ecuación 139)}$$

3.1.5 Detección de movimiento

Esta característica ayuda a la segmentación, ya que dos puntos de un mismo objeto deben tener velocidades parecidas.

Los movimientos detectados pueden ser debidos a movimientos de la cámara, de los objetos, a cambios de iluminación o a cambios en la forma o tamaño de los objetos. Uno de los métodos más simples para la detección de movimiento es el basado en la diferencia de imágenes

Partiendo de una secuencia de imágenes $f(x,y,t)$ y la imagen que indicará el campo de velocidades entre los instantes t_0 y t_1 será:

$$F_{t0\ t1} = \begin{cases} 1 & si\ |f(x,y,t_0) - f(x,y,t_1)| \geq T \\ 0 & en\ otro\ caso \end{cases} \quad \text{(Ecuación 140)}$$

Se utiliza la umbralización para minimizar la influencia del ruido. Este método es inmune a los cambios de iluminación debido al corto tiempo transcurrido entre toma y toma pero presenta el inconveniente de que los cambios detectados son debidos a la parte del fondo tapado por el objetivo en la segunda imagen y a la parte del fondo que estaba oculto en la primera imagen. Por último no aporta información sobre el sentido del movimiento.

Existen dos grandes grupos de métodos de obtención del campo de velocidades, que solucionan los problemas anteriores:

- El primero busca una serie de características en la primera imagen, como son los bordes o las esquinas, para encontrar su correspondencia en la segunda. El problema es que el número de estas características suele ser bajo para implementar el método.

- El segundo grupo pretende la obtención de un mapa denso del campo de velocidades, busca en un entorno la zona más parecida a la segunda imagen dependiendo de sus variaciones espaciales y temporales. Presentan el problema de la apertura.

Es importante indicar que no se puede calcular la velocidad exacta de los puntos de la imagen, si solamente se tiene información de una pequeña vecindad en torno a un punto puesto que la única información obtenida de esa vecindad es la componente normal al borde del objeto.

3.2 Segmentación de imágenes

Una vez extraídas las características de una imagen, procedemos a la partición de la misma para su posterior reconocimiento e interpretación. Por ello decimos que la segmentación consiste en separar los objetos localizados en un entorno sobre la base de una o varias de las características obtenidas en el proceso de extracción.

Se basan en tres propiedades:

- Similitud: los píxeles de un elemento tienen valores parecidos de alguna propiedad.

- Discontinuidad: los objetos destacan del entorno, teniendo unos bordes definidos.

- Conectividad: los píxeles pertenecientes a un mismo objeto deben ser contiguos y estar agrupados.

Las diferentes técnicas de segmentación se basan en estas propiedades, dando lugar a la búsqueda de partes uniformes de la imagen o por el contrario a la búsqueda de partes donde se produce un cambio. Agrupamos los métodos de segmentación en tres grupos:

- Segmentación basada en píxeles.
- Segmentación basada en bordes.
- Segmentación basada en regiones.

3.2.1 Segmentación basada en píxeles

Este método tiene en cuenta el nivel de gris de un píxel para decidir si el mismo pertenece o no al objeto de interés, para ello debemos encontrar el rango de niveles de gris que caracterizan dicho objeto, teniendo en cuenta la propiedad de similitud. Para ello obtendremos previamente el histograma de la imagen.

Esta técnica también se conoce como segmentación basada en umbralización, porque convertimos una imagen con varios niveles de gris en una nueva con sólo dos niveles, de manera que los objetos quedan separados del fondo, basándonos en que los píxeles de un determinado objeto tienen el mismo valor de gris.

Mostramos la función que define la imagen umbralizada según el caso de que se trate:

Si el valor buscado es mayor que el umbral:

$$g(x,y) = \begin{cases} 1 & si\ T \geq f(x,y) \\ 0 & en\ otro\ caso \end{cases} \qquad \text{(Ecuación 141)}$$

Si el valor buscado es menor que el umbral:

$$g(x,y) = \begin{cases} 1 & si\ T \leq f(x,y) \\ 0 & en\ otro\ caso \end{cases} \qquad \text{(Ecuación 142)}$$

Si los objetos pertenecen a un intervalo a-b:

$$g(x,y) = \begin{cases} 1 & si\ T_a \leq f(x,y) \leq T_b \\ 0 & en\ otro\ caso \end{cases} \qquad \text{(Ecuación 143)}$$

3.2.2 Segmentación basada en bordes:

Esta técnica realiza la búsqueda del valor máximo del gradiente sobre cada línea que forma la imagen. Cuando se encuentra un máximo, usamos un algoritmo de trazado para seguir el máximo del gradiente alrededor del objeto hasta encontrar de nuevo el punto inicial. Y volvemos a buscar un nuevo máximo, etc.

Se parte de los bordes encontrados en la imagen por los detectores, puesto que por sí mismos no pueden unir los bordes, debido a que están afectados por el ruido de la imagen y además hay partes que no son detectadas.

Existen diferentes métodos para la segmentación que difieren en la estrategia para la construcción del contorno, así como en la cantidad de información previa que se incorpora en el método. Incluimos algunos de estos métodos:

3.2.2.1 Conectividad

Aquí la extracción de contornos se entiende como la unión de los bordes que se encuentren conectados. Decimos que un píxel está conectado cuando tiene alguna propiedad común con los píxeles de su entorno.

3.2.2.2 Seguimiento del contorno

Estos trazan los contornos ordenando los puntos del tipo borde sucesivo. Plantearemos un algoritmo como ejemplo.

- Primero, comenzaremos por el primer píxel encontrado en la región al hacer un rastreo por filas (por ejemplo).

- Seguidamente, giramos a la izquierda y pasamos al siguiente píxel, si se encuentra dentro de la misma región, en caso contrario giraremos a la derecha y pasamos al siguiente píxel.

- Por último, continuaremos hasta que se llegue al punto del que se partió.

3.2.2.3. Unión de bordes y búsqueda heurística en grafos

Aquí entendemos el contorno como un camino, formado a través de un grafo construido uniendo los elementos bordes. Suponiendo que un grafo se forma desde el nodo A hasta el B. Los algoritmos de búsqueda heurística, examinan los sucesores del nodo de salida A y seleccionan el que maximice la función. El nodo seleccionado se convierte ahora en el de salida y se repite el proceso hasta alcanzar el nodo B. La sucesión de nodos seleccionada constituye el contorno.

3.2.2.4 Programación dinámica

El método anterior sólo nos asegura el camino óptimo entre dos nodos consecutivos, pero no el camino óptimo global. Para resolver este problema aparecen los métodos de programación dinámica que se basan en el principio de optimización de Bellman, el cual nos indica que el camino óptimo entre dos puntos dados de un camino, es también óptimo entre dos puntos cualquiera que se encuentren en él.

Para aplicar esta idea a la extracción de contornos, se va a suponer que se ha convertido el mapa de los puntos que son bordes de una imagen, en un grafo de N nodos. La función de evaluación será la siguiente:

$$S(x_{1,}x_2,\dots,N)=\sum_{K=1}^{N}|g(x_k)| - \alpha\sum_{K=2}^{N}|\theta(x_k)-\theta(x_{k-1})| - B\sum_{K=2}^{N}d(x_k,y_{k-1}) \quad \text{(Ecuación 144)}$$

Donde $x_K, K=1,\dots,N$ representan los nodos; es decir, los píxeles pertenecientes al borde, $d(x,y)$ es la distancia entre dos nodos x e y. $|g(x_k)|, \theta(x_k)$ son el módulo y el ángulo del gradiente del nodo en la posición x_k y los parámetros α y β son constantes no negativas. Teniendo en cuenta esta definición, decimos que el término que conecta los N nodos, es óptimo cuando la función S sea máxima.

$$\emptyset(x_N,N)=máx\{S(x_1,\dots,x_N,N)\} \quad \text{(Ecuación 145)}$$

3.2.2.5 Relajación de bordes

Este método tiene en cuenta los bordes vecinos para la elaboración del contorno. Se evalúan iterativamente con más precisión todas las propiedades de una imagen junto con la posibilidad de existencia de más bordes, hasta que su contexto queda claro. Se basa en la fuerza de los bordes de una vecindad local especificada, aumentando o disminuyendo la confianza de cada borde. Un borde débil posicionado entre dos fuertes, proporciona un ejemplo de contexto; por otra parte, si un borde, incluso uno fuerte no tiene contexto que lo soporte, puede que no forme parte de ningún contorno.

3.2.2.6 La transformada de Hough

Está basada en la información que suministra toda la imagen. Hace una transformación de los puntos de la imagen a un espacio de parámetros de dimensión n a partir del cual, determina la forma geométrica en cuestión en función del valor del parámetro más repetido.

3.2.3 Segmentación orientada a regiones

Podemos valorar múltiples técnicas.

3.2.3.1 Segmentación de regiones por clasificación

Se utilizan niveles de gris presentes en la imagen para obtener una partición del espacio. Se asocia a cada píxel su clase de nivel de gris. Las regiones son definidas por los conjuntos de píxeles conexos pertenecientes a una misma clase. Utiliza el histograma como herramienta para la clasificación de los distintos brillos. Lo convierte en un método eficaz si la clasificación permite definir las diferentes regiones homogéneas de la imagen.

3.2.3.2 Segmentación por crecimiento de regiones

Para la implementación de este método, las regiones han de ser homogéneas maximales, lo que significa que los criterios de homogeneidad a partir de los cuales se forman regiones no se seguirán cumpliendo tras la unión de una región con alguna de sus regiones adyacentes.

3.2.3.3 Unión de regiones

Considera a cada píxel como una región. Estas serán homogéneas pero no necesariamente maximales por lo que el proceso deberá repetirse hasta que las regiones sean cumplan este criterio. A continuación describimos el método:

- Definimos una segmentación inicial que cumpla el criterio de homogeneidad.
- Definimos un criterio para unir regiones adyacentes.
- Unimos las regiones adyacentes si cumplen el criterio común, para cuando no puedan unirse dos regiones sin romper el criterio de homogeneidad.

Podemos resumir este proceso indicando que la descripción de una región se compara con la de otra adyacente. Si coinciden, se unen las regiones y se computa el nuevo descriptor de la región; en caso contrario, las regiones se marcan como que no pueden unirse y el proceso continúa.

3.2.3.4 Unión de regiones por agregación de píxeles

El crecimiento de regiones se realiza agrupando píxeles o subregiones en regiones mayores. La agrupación de píxeles se realiza si son similares (niveles de gris, color y textura) respecto a la propiedad usada para su segmentación. El punto de partida es un conjunto de píxeles semilla; es decir, los píxeles sobre los cuales se evalúan los demás y a partir de los cuales se hace crecer la región.

3.2.3.5 Separación y unión de regiones

Consiste en subdividir la imagen inicialmente en un conjunto de regiones disjuntas y luego unirlas o separarlas en un intento de que todas ellas:

- Cubran la imagen completa.
- Cada una de ellas esté conectada.
- Sus intersecciones dos a dos, sean el conjunto vacío.
- Que algún criterio de homogeneidad sea cumplido en cada región.
- Que la unión de dos cualesquiera de ellas no cumpla algún criterio de la homogeneidad.

3.2.4 Segmentación basada en texturas

Esta técnica se caracteriza por las propiedades explicadas en el apartado 3.2, para realizar la partición de las imágenes. Pudiendo optar por una aproximación estadística con sus correspondientes propiedades diferenciadoras entre unas partes y otras o por una aproximación por frecuencias, definiendo el contenido local en frecuencias de una determinada región.

3.3 Transformaciones morfológicas

Obtenemos de la imagen los elementos más importantes que la constituyen mediante la segmentación. Estos elementos; que son los objetos sobre la imagen, presentan defectos inherentes al proceso de segmentación, por lo que necesitan un posterior procesado consistente en la realización de transformaciones morfológicas para eliminar ruido o píxeles mal clasificados.

Las transformaciones morfológicas son aquellas que modifican la forma o la estructura de los objetos segmentados de forma que se simplifiquen los datos de la imagen y se preserven las características esenciales eliminando los aspectos irrelevantes.

Primero vamos a recordar algunas operaciones básicas sobre conjuntos para mejorar el entendimiento del procesado que implica la morfología matemática. Para ello partiremos de dos conjuntos A y B en un espacio E de cierta dimensión, con elementos a y b pertenecientes a A y B respectivamente.

- Inclusión: A está incluido en B, $A \subset B$, si todo elemento de A pertenece a B.

- Complemento: A^C es el complemento de A y contiene todos los elementos que no pertenecen a A.

- Unión: la unión de dos conjuntos $A \cup B$, son todos los elementos pertenecientes a uno de los dos conjuntos.

$$A \cup B = \{x | x \in A \text{ ó } x \in B\} \qquad \text{(Ecuación 145)}$$

- Intersección: la intersección de dos conjuntos $A \cap B$, son los elementos comunes a ambos conjuntos.

$$A \cap B = \{x | x \in A \, y \, x \in B\} \qquad \text{(Ecuación 146)}$$

- Reflexión: $\hat{A}$ es la reflexión de A.

$$\hat{A} = \{x | x = -a, \quad para\ todo\ \ a \in B\} \qquad \text{(Ecuación 147)}$$

- Diferencia entre A y B: $A - B = \{x | x \in A, x \notin B\}$.

$$A - B = A \cap B^C \qquad \text{(Ecuación 149)}$$

- Traslación: un conjunto A es trasladado un vector v cuando cada uno de los elementos de A sufre esa traslación.

$$A_v = \{x | x = a + v, \quad para\ todo\ a \in A\} \qquad \text{(Ecuación 150)}$$

Dos operaciones importantes que sirven de base a la morfología matemática y a su vez están basadas en la traslación, son la Suma y la Resta de Minkowski, las cuales se van a definir a continuación:

Suma de Minkowski: $A \oplus B = \bigcup_{x \epsilon B} A_x$ (Ecuación 151)

Resta de Minkowski: $A \ominus B = \bigcap_{x \epsilon B} A_x$ (Ecuación 152)

Donde la suma es la unión y la resta es la intersección del conjunto A y el B trasladado su origen a cada elemento de A.

3.3.1 Transformaciones morfológicas en imágenes binarias

En una imagen binaria los píxeles con valor 1 se pueden considerar como un conjunto del universo de píxeles de la imagen, constituyendo los píxeles de valor 0 el conjunto complementario del anterior y siendo considerados puntos pertenecientes al fondo.

Este tipo de transformaciones requiere la utilización de un elemento estructural que es un subconjunto de puntos cuya representación en el plano tiene cierta forma y tamaño. Con él se irá probando la imagen y cuantificando el modo en que está contenido en la misma.

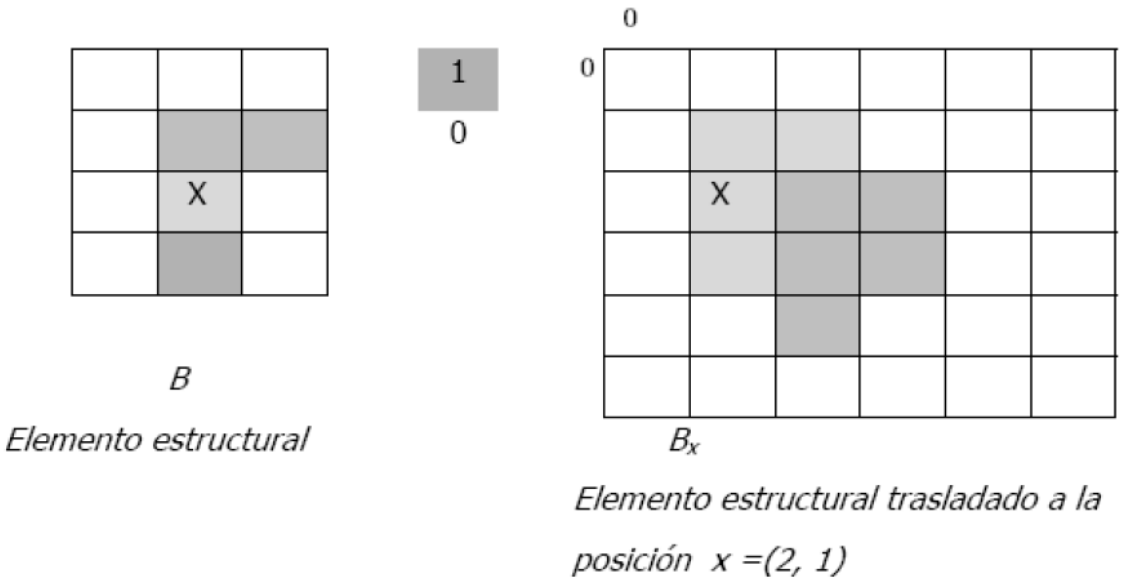


Imagen 31. Elemento estructural

La forma de operar con un elemento estructural es la siguiente:

- Elegir un elemento estructural (B) adecuado al problema.

• Desplazar B por todas las posiciones del espacio E en el que se encuentra el conjunto a estudiar A: $B_x \equiv B_{(x,y)}$

• En cada posición preguntarse si la unión, la intersección o la inclusión de A con B, está dentro de A.

• Los puntos con respuesta afirmativa formarán parte de un nuevo conjunto que será en este caso la imagen transformada.

En los puntos siguientes haremos referencia a las principales operaciones morfológicas y se considerará el conjunto A como la imagen sobre la que se realiza la transformación y el conjunto B el elemento estructural.

3.3.1.1 Erosión

Consiste en una reducción de algún elemento de la imagen original, es la degradación progresiva de uno de los campos 0 ó 1 pertenecientes a la imagen binaria. Un punto del campo a degradar seguirá perteneciendo al mismo si está rodeado de puntos iguales a él, de lo contrario pasará a formar parte del otro campo.

Definimos la erosión de A por B como: $er(A,B) = \{x | B_x \subset A\}$.

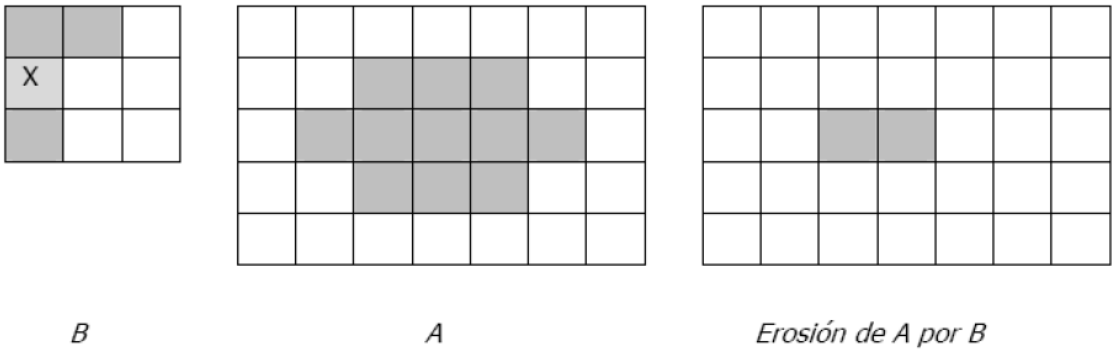


Imagen 32. Erosión

Como esta definición es difícil de implementar, daremos una definición de la erosión, equivalente a la anterior, en función de la Resta de Minkowski. De esta forma no tendremos que recorrer toda la imagen píxel a píxel:

$$A \ominus \bar{B} = \bigcap_{b \epsilon \hat{B}} A_b \qquad \text{(Ecuación 153)}$$

Con esta definición se considera la erosión de una imagen A por un elemento estructural B, como la intersección de todas las traslaciones de A por los puntos –b pertenecientes a la reflexión de $B(\bar{B})$.

3.3.1.1.1 Propiedades

• Antiextensiva: si el origen se encuentra dentro del elemento estructural (que el centro del elemento estructural sea 1), la erosión tiene el efecto de encoger la imagen de entrada, obteniendo una imagen incluida dentro de la original. Lo expresamos matemáticamente de la siguiente manera:

Si B contiene al origen, $A \ominus B \subseteq A$ (Ecuación 154)

• Invariante a traslaciones:

$$A_x \ominus B = (A \ominus B)_x \quad \text{(Ecuación 155)}$$

$$A \ominus B_x = (A \ominus B)_{-x} \quad \text{(Ecuación 156)}$$

• Creciente:

$$A \subseteq C \Rightarrow A \ominus B \subseteq C \ominus B \quad \text{(Ecuación 157)}$$

• La intersección es distributiva con respecto a la erosión:

$$(A \cap C) \ominus B = (A \ominus B) \cap (C \ominus B) \quad \text{(Ecuación 158)}$$

• La unión no es distributiva con respecto a la erosión pero cumple:

$$(A \cup C) \ominus B \supseteq (A \ominus B) \cup (C \ominus B) \quad \text{(Ecuación 159)}$$

3.3.1.2 Dilatación

Es el crecimiento progresivo de uno de los campos (1 ó 0). Si un elemento del campo que crece tiene un vecino del otro campo, este último se convierte al campo que se dilata. Y si el vecino es del mismo campo, el elemento no se altera.

Definimos la dilatación de A por B como: $dil(A,B) = \{x | B_x \cap A \neq 0\}$

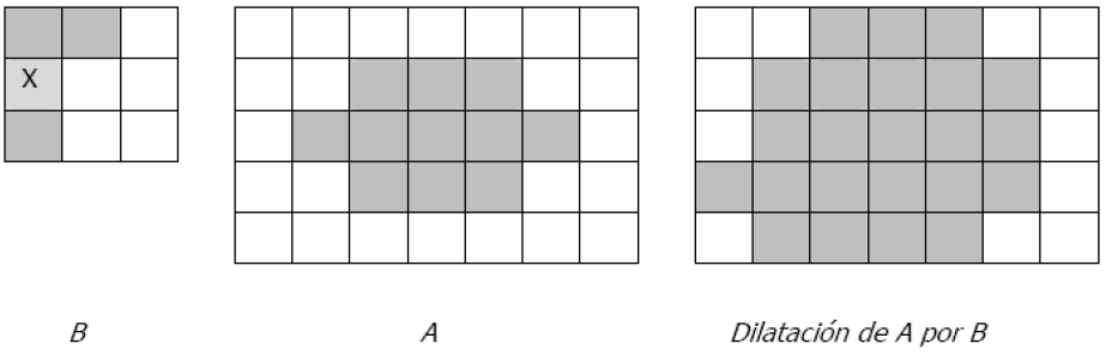


Imagen 33. Dilatación

Al igual que con la erosión, la dilatación se puede poner en función de la Suma de Minkowski:

$$A \oplus \bar{B} = \bigcap_{b \epsilon \bar{B}} A_b \quad \text{(Ecuación 160)}$$

De esta manera queda expresada como la unión de traslaciones de A por los elementos de B.

3.3.1.2.1 Propiedades

• Conmutativa:

$$A \oplus B = B \oplus A \quad \text{(Ecuación 161)}$$

• Asociativa:

$$A \oplus (B \oplus C) = (A \oplus B) \oplus C \quad \text{(Ecuación 162)}$$

• Invariante a traslaciones:

$$A_x \oplus \bar{B} = (A \oplus B)_x \quad \text{(Ecuación 163)}$$

• Una traslación en la imagen puede compensarse en la definición del elemento estructural, tomando este último trasladado en la dirección opuesta:

$$A_x \oplus B_{-x} = A \oplus B \quad \text{(Ecuación 164)}$$

• Creciente:

$$A \subseteq C \Rightarrow A \oplus B \subseteq C \oplus B \quad \text{(Ecuación 165)}$$

• Extensiva: si el origen se encuentra dentro del elemento estructural, la dilatación agranda la imagen de entrada, obteniendo una imagen que incluye a la original, de no ser así, la imagen resultante puede no incluir a la imagen original. Matemáticamente lo expresamos:

Si B contiene al origen $A \oplus B \supseteq A$ (Ecuación 166)

• La unión es distributiva con respecto a la dilatación:

$$(A \cup C) \oplus B = (A \oplus C) \cup (B \oplus C) \quad \text{(Ecuación 167)}$$

$$A \oplus (B \cup C) = (A \oplus B) \cup (A \oplus C) \quad \text{(Ecuación 168)}$$

Esta última ecuación permite descomponer el elemento estructural en la unión de elementos estructurales.

• La intersección no es distributiva con respecto a la dilatación, pero cumple:

$$(A \cap B) \oplus C \subseteq (A \oplus C) \cap (B \oplus C) \quad \text{(Ecuación 169)}$$

$$A \oplus (B \cap C) \subseteq (A \oplus B) \cap (A \oplus C)$$ (Ecuación 170)

• Dualidad entre erosión y dilatación: dos operadores son duales cuando la negación de una formulación empleada en el primero es igual a la misma formulación empleando el segundo operador en la variable negada (lo que uno hace al objeto, el otro lo hace al fondo). Lo expresamos matemáticamente:

$$(A \ominus B)^C = A^C \oplus \bar{B}$$ (Ecuación 171)

$$(A \oplus B)^C = A^C \ominus \bar{B}$$ (Ecuación 172)

Donde A^C es la negación lógica de A y $\bar{B}$ la negación geométrica (reflexión) de B.

• La regla de la cadena para la erosión se verifica cuando el elemento estructural se puede descomponer mediante dilatación:

$$A \ominus (B \oplus C) = (A \ominus B) \ominus C$$ (Ecuación 173)

3.3.1.3 Acierta o falla (hit or miss)

Se aplica a cada punto del conjunto A de la siguiente manera:

A partir del elemento estructural B se forma el conjunto B_x, que es ese mismo elemento estructural desplazado para todo elemento $x \in A$. Se considera B_x al formado por dos subconjuntos B_{x1} y B_{x2} donde el primero corresponde a los elementos del primer plano y el segundo a los del fondo.

Un punto x pertenece a la transformación Acierta o Falla, denotada por $A \otimes B$, si y sólo si B_{x1} está incluido en A y B_{x2} está incluido en A^C.

$$A \otimes B = \{x | B_{x1} \subset A; B_{x2} \subset A^C\}$$ (Ecuación 174)

Esta transformación indica donde coincide exactamente el elemento estructural B en el conjunto A.

También podemos expresar esta transformación en función de la erosión y de la dilatación:

$$A \otimes B = (A \ominus B_1) \cap (A^C \ominus B_2)$$ (Ecuación 175)

$$A \otimes B = (A \ominus B_1) - (A \oplus) \bar{B}_2$$ (Ecuación 176)

3.3.1.4 Apertura

Consiste en una erosión, seguida de una dilatación, empleando el mismo elemento estructural.

$$A^{\circ} B = (A \ominus B) \oplus B$$ (Ecuación 177)

El efecto que produce sobre una imagen, dependiendo del elemento estructural elegido, será el suavizado de los contornos de los objetos, rotura de enlaces delgados y descomposición de objetos.

3.3.1.5 Cierre

Es la operación dual de la apertura, realizada a partir de una dilatación seguida de una erosión.

$$A \bullet B = (A \oplus B) \ominus B \qquad \text{(Ecuación 178)}$$

Esta operación tiende a suavizar el contorno de la imagen de entrada, rellenando roturas y pequeños agujeros y completando espacios del contorno.

Las operaciones morfológicas que se van a ver a continuación, están basadas en algunas operaciones vistas hasta ahora y pueden emplearse con fines variados como puede ser la detección de bordes, segmentación o en procesamiento (realce) además de postprocesamiento.

3.3.1.6 Extracción del contorno

Operación basada en la erosión, con la que se obtiene el contorno de un conjunto A, erosionando A por B y realizando la diferencia entre A y su erosión, obteniendo así el contorno interior:

$$\beta(A) = A - (A \ominus B) \qquad \text{(Ecuación 179)}$$

También se puede realizar a partir de la dilatación, en este caso es llamado contorno exterior:

$$P(A) = (A \oplus B) - A \qquad \text{(Ecuación 180)}$$

3.3.1.7 Relleno de regiones

Mediante esta operación pretendemos rellenar el interior de un contorno a partir de operaciones morfológicas. Se parte del supuesto que los puntos que no pertenecen al contorno tienen valor cero, formando el complementario de dicho contorno.

$$X_k = (X_{k-1} \oplus B) \cap A^C \quad k = 1,2,3 \ldots \qquad \text{(Ecuación 181)}$$

3.3.1.8 Extracción de componentes conexas

Pretendemos extraer las componentes conexas contenidas en un conjunto A, obtenemos los elementos conectados entre sí de dicho conjunto.

Con Y se representa una componente conexa contenida en un conjunto A y se supone conocido el punto p que pertenece a dicha región. Para extraer Y se emplea el siguiente proceso iterativo, basado en la dilatación:

$$X_k = (X_{k-1} \oplus B) \cap A \quad k = 1,2,3 \ldots \qquad \text{(Ecuación 182)}$$

3.3.1.9 Envolvente convexa (convex hull)

La envolvente convexa es la región más pequeña que contiene al objeto de forma que dos puntos cualesquiera pertenecientes a dicha envolvente pueden conectarse a través de una línea recta, perteneciendo a la región todos los puntos de ella.

Para obtener morfológicamente la envolvente convexa de un conjunto A, seguimos el siguiente procedimiento:

$$X_k^i = \left(X \otimes B^i\right) \cup A \qquad k = 1,2,3 \ldots \qquad i = 1,2,3 \ldots \qquad \text{(Ecuación 183)}$$

3.3.1.10 Adelgazamiento (Thinning)

El adelgazamiento de un conjunto A por un elemento estructural B se define en la función de la transformación acierta o falla de la siguiente manera:

$$A \emptyset B = A - (A \otimes B) = A \cap (A \otimes B)^C \qquad \text{(Ecuación 184)}$$

3.3.1.11 Engrosamiento (Thickening)

Es el proceso morfológico dual al adelgazamiento. Lo definimos de la siguiente manera:

$$A(\bullet)B = A \cup (A \otimes B) \qquad \text{(Ecuación 185)}$$

3.3.1.12 Esqueletización (Skeletonization)

Es la obtención del esqueleto de una región (objeto), que no es más que la representación en forma de grafo de dicha región.

Pequeñas variaciones de su región se traducen en grandes variaciones en su esqueleto, lo que resulta muy útil para detectar pequeños defectos en piezas y en sistemas para el control de calidad.

$$S(A) = \bigcup_{k=0}^{L} S_k(A) \qquad \text{(Ecuación 186)}$$

Siendo:

$$S_k(A) = \bigcup_{k=0}^{K} \{(A \ominus kB) - [(A \ominus kB) \circ B]\} \qquad \text{(Ecuación 187)}$$

3.3.1.13 Eliminación de ramas (Pruning)

Tras un proceso de adelgazamiento o de esqueletización, el esqueleto presenta ramificaciones acompañando al tronco principal, que no aportan información acerca del objeto y dificultan su identificación, por ello es necesario un proceso de poda o eliminación de ramas.

3.3.1.14 Eliminación del ruido

Tras la segmentación de una imagen habrá píxeles pertenecientes al objeto que se nos hayan colado del fondo y viceversa. Resolvemos el problema con erosiones y dilataciones eliminando los huecos de los objetos y los puntos ruidosos del fondo mediante la siguiente expresión:

$$X_4 = (((A \ominus B) \oplus B) \oplus B) \ominus B \qquad \text{(Ecuación 188)}$$

Lo que equivale a una apertura seguida de un cierre:

$$X_4 = (A \circ B) \bullet B \qquad \text{(Ecuación 189)}$$

3.3.2 Transformaciones morfológicas en imágenes con varios niveles de gris

Se trata de extender las transformaciones morfológicas vistas anteriormente a imágenes con distintos niveles de gris.

3.3.2.1 Erosión

Definimos erosión para varios niveles de gris de la siguiente manera:

$$(f \ominus b)(s,t) = max\{f(s+x,t+y) - b(x,y) | (s+x,t+y) \in D_f; (x,y) \in D_b\} \qquad \text{(Ecuación 190)}$$

Donde D_f es el dominio de la imagen f(x,y) y D_b el del elemento estructural.

3.3.2.2 Dilatación

Definimos la dilatación de una imagen con varios niveles de gris por un elemento estructural:

$$(f \oplus b)(s,t) = max\{f(s-x,t-y) + b(x,y) | (s-x,t-y) \in D_f; (x,y) \in D_b\} \qquad \text{(Ecuación 191)}$$

Donde D_f es el dominio de la imagen f(x,y) y D_b el del elemento estructural. Con los dominios se asegura que las dos imágenes coincidan en algún punto.

3.3.2.2 Apertura y cierre

Son las mismas que para las imágenes binarias.

La apertura de f por b:

$$f \circ b = (f \ominus b) \oplus b \qquad \text{(Ecuación 192)}$$

De la misma manera, el cierre de f por b:

$$f \bullet b = (f \oplus b) \ominus b \qquad \text{(Ecuación 193)}$$

3.4 Representación y descripción de contornos y regiones

Una vez que la imagen ha sido segmentada en regiones y en algunos casos corregidos los errores cometidos en dicha segmentación, es necesario representar y describir dichos píxeles, mediante unas características que se obtendrán a partir de esas regiones, para su posterior procesamiento.

Podemos representar una región de dos maneras, mediante su contorno o representando el interior de la región. Dependiendo de las características que queramos extraer usaremos un método u otro. En general para la extracción de las características geométricas usaremos el contorno y para las características físicas se representará la región.

3.4.1 Representación del contorno

3.4.1.1 Códigos cadena

Estos códigos representan contornos mediante una sucesión de segmentos conectados de una longitud y dirección dada.

3.4.1.2 Curvas $\emptyset - S$

Es un método similar al de los códigos cadena. En este caso modificamos el ángulo $\emptyset$ de los vectores tangentes en cada punto del contorno, en función de la longitud de arco S del contorno. El ángulo $\emptyset$ está hace referencia al ángulo de la tangente en el punto origen del contorno (ángulo cero), de forma que la curva $\emptyset - S$ empieza y termina en cero. Posee múltiples ventajas:

- Es invariante a traslaciones.
- Las limitaciones en los ángulos posibles sólo están condicionadas por el método de segmentación utilizado.
- Es invariante a rotaciones.

3.4.1.3 Aproximaciones poligonales

Estos métodos aproximan el contorno mediante un polígono; es decir, representan tramos curvos mediante tramos lineales. De esta forma hacen corresponder segmentos de línea al contorno, almacenando sólo los parámetros de esos segmentos de línea en lugar de los puntos discretos del contorno.

3.4.1.4 Método de firmas

Una firma es una representación unidimensional de un contorno bidimensional y se puede generar de diferentes formas.

3.4.2 Representación de regiones

La forma de un objeto puede representarse directamente por la región que ocupa en la imagen, de manera que se construye una imagen binaria con la expresión:

$$f(x,y) = \begin{cases} 1 & si\ (x,y)\ pertenece\ a\ la\ región \\ 0 & en\ caso\ contrario \end{cases} \quad \text{(Ecuación 194)}$$

A continuación veremos una serie de métodos para representar de forma más efectiva dicha región:

3.4.2.1 Códigos de longitud variable

Una región o imagen binaria es vista como una sucesión en la que se alternan series de ceros y unos. Si consideramos el objeto codificado con unos, ocupa en la mayoría de los casos un espacio en la imagen inferior al que ocupa el fondo. Si codificamos exclusivamente las cadenas de unos consecutivos se minimizará el espacio de memoria ocupado por la imagen. De esta forma, se codificará el comienzo de las cadenas de unos y la longitud de esas cadenas.

3.4.2.2 Proyecciones

Una región puede representarse por sus proyecciones. La proyección de una imagen sobre una recta consiste en obtener para cada punto de ella, el número de píxeles que pertenecen a la perpendicular que pasa por ese punto.

3.4.3 Descriptores de contornos

Son un conjunto de números producidos para describir la forma de un objeto. Se hallan mediante el cálculo de algunas características en particular. Como la forma del objeto no se puede describir de manera completa mediante los descriptores, estos deben ser suficientemente diferentes para poder distinguir objetos con distintas formas.

3.4.3.1 Longitud de un contorno

Lo obtenemos contando el número de píxeles del contorno. Si este viene codificado en códigos cadena, se sumarán los componentes horizontales, los verticales y los diagonales multiplicados por $2^{1/2}$.

3.4.3.2 Diámetro de un contorno

Se define como:

$$Diam(C) = máx\left[D(p_i, p_j)\right] \quad \text{(Ecuación 195)}$$

Donde D es la distancia entre dos puntos del contorno, C y p_i, p_j son dos puntos pertenecientes al contorno. La línea que une los dos puntos del diámetro llamada eje mayor del contorno también es un descriptor útil.

3.4.3.3 Esquinas

La curvatura a lo largo del borde de un objeto. Lo definimos.

$$|k(t)|^2 = \left(\frac{d^2y}{dt^2}\right)^2 + \left(\frac{d^2x}{dt^2}\right)^2 \quad \text{(Ecuación 196)}$$

Son marcados como esquinas, aquellos puntos donde la función k(t) toma valores elevados, quedando así el objeto definido por sus esquinas.

3.4.3.4 Descriptores de Fourier

Se realiza calculando la transformada de Fourier de un conjunto de píxeles. Si hay contenido en alta frecuencia significa que hay cambios rápidos en la coordenada correspondiente, por el contrario si encontramos contenido en baja frecuencia significa que el contorno varía de forma suave. Por lo que los componentes en baja frecuencia capturan la forma general del contorno y las de alta frecuencia capturan los detalles. Si realizamos un filtrado paso bajo de los descriptores de Fourier de un contorno sería equivalente a suavizar este contorno.

3.4.4 Descriptores de regiones

Se encargan de caracterizar la región resultante de una segmentación. Cumplen los mismos requisitos que los descriptores de contornos en cuanto a que deben ser invariantes.

3.4.4.1 Área

El área de una región vendrá dada por la siguiente ecuación:

$$A = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \quad \text{(Ecuación 197)}$$

Donde f(x,y) es la imagen binaria.

3.4.4.2 Perímetro

El perímetro está constituido por el número de píxeles exteriores de la región, los del contorno.

3.4.4.3 Compacticidad

Refleja el empaquetamiento de la región y se define como el perímetro al cuadrado dividido por el área.

$$C = \frac{P^2}{A} \qquad \text{(Ecuación 198)}$$

Es un descriptor sin dimensiones por lo que es insensible a cambios de escala e invariante a la rotación.

3.4.4.4 Descriptores topológicos

Indican de una forma sencilla alguna idea sobre la forma de la región:

- El número de agujeros en la región (H).
- El número de componentes conectados, partes separadas que forman la región (C).
- El número de Euler, es la diferencia entre los dos anteriores ($E = C - H$).

3.4.4.5 Texturas

Algunas regiones pueden ser descritas como texturas siendo obtenidas a partir de los métodos estadísticos o frecuenciales comentados anteriormente.

4. Correlación digital de imágenes (DIC)

En las décadas de los 60 y 70, investigadores en inteligencia artificial y robótica comenzaron a desarrollar algoritmos para ser usados en visión artificial y metodologías de estéreo visión en paralelo con aplicaciones de fotogrametría. A medida que las técnicas basadas en imágenes digitales fueron mejorando, el campo experimental de la ingeniería fue buscando aplicaciones de estas técnicas como la holografía, interferometría etc.

En la década de los 80 se publica el primer resultado de una investigación sobre deformaciones de un objeto usando imágenes digitales. A tal efecto, compararon la localización de pequeñas regiones en una imagen digital del objeto ensayado, antes y después de ser aplicada la carga, pudiendo obtener una medición de campo completo de los desplazamientos producidos entre ambos estados. Más tarde, este método fue mejorado incluyendo en el algoritmo numérico optimizaciones mediante el método de Newton-Ralphson (el cuál comentaremos ampliamente más adelante), desembocando en el desarrollo de la técnica de Correlación Digital de Imágenes.

Mediante este análisis nos proponemos realizar una explicación del marco de trabajo matemático y algoritmos modernos utilizados para la correlación digital de imágenes en dos dimensiones en general.

4.1 El problema general

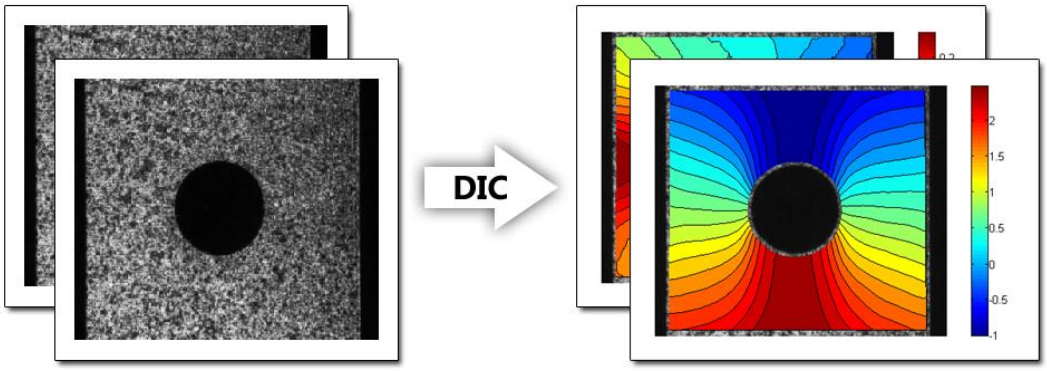


Imagen 34

El objetivo general de DIC es la obtención de campos de desplazamientos y deformaciones dentro de una región de interés (ROI) para una muestra de material sometida a una deformación. DIC utiliza técnicas de procesamiento de imágenes en un intento de resolver este problema.

Básicamente se toman imágenes de una muestra mientras se deforman; estas imágenes se usan como entradas a un programa de DIC. La idea es obtener uno por uno (de alguna manera), una correspondencia entre los puntos significativos de la referencia (imagen sin deformada inicial) y la configuración actual (posteriores imágenes deformadas). DIC realiza este proceso tomando pequeñas subsecciones de la imagen de

referencia; denominados subconjuntos, determinando sus respectivas ubicaciones en la configuración actual. Para cada subconjunto obtenemos el desplazamiento y la información de la deformación a través de la transformación usándola para que coincida con la ubicación del subconjunto en la configuración actual. Muchos subconjuntos son recogidos en la configuración de referencia, a menudo con un parámetro de separación para reducir el coste computacional (teniendo en cuenta que normalmente se solapan subconjuntos).

El resultado final es una cuadrícula que contiene la información del desplazamiento y de la deformación con respecto a la configuración de referencia, también conocida como desplazamientos/deformaciones de Lagrange. Los campos de desplazamiento/deformación pueden entonces ser reducidos o interpolados para formar un campo continuo. Estas ideas serán más precisas en las siguientes secciones.

4.2 Implementación Computacional

Para ser más específicos, los subconjuntos son esencialmente un grupo de puntos de coordenadas, la idea de subconjuntos en la imagen de referencia y la actual se muestra en la figura 1.

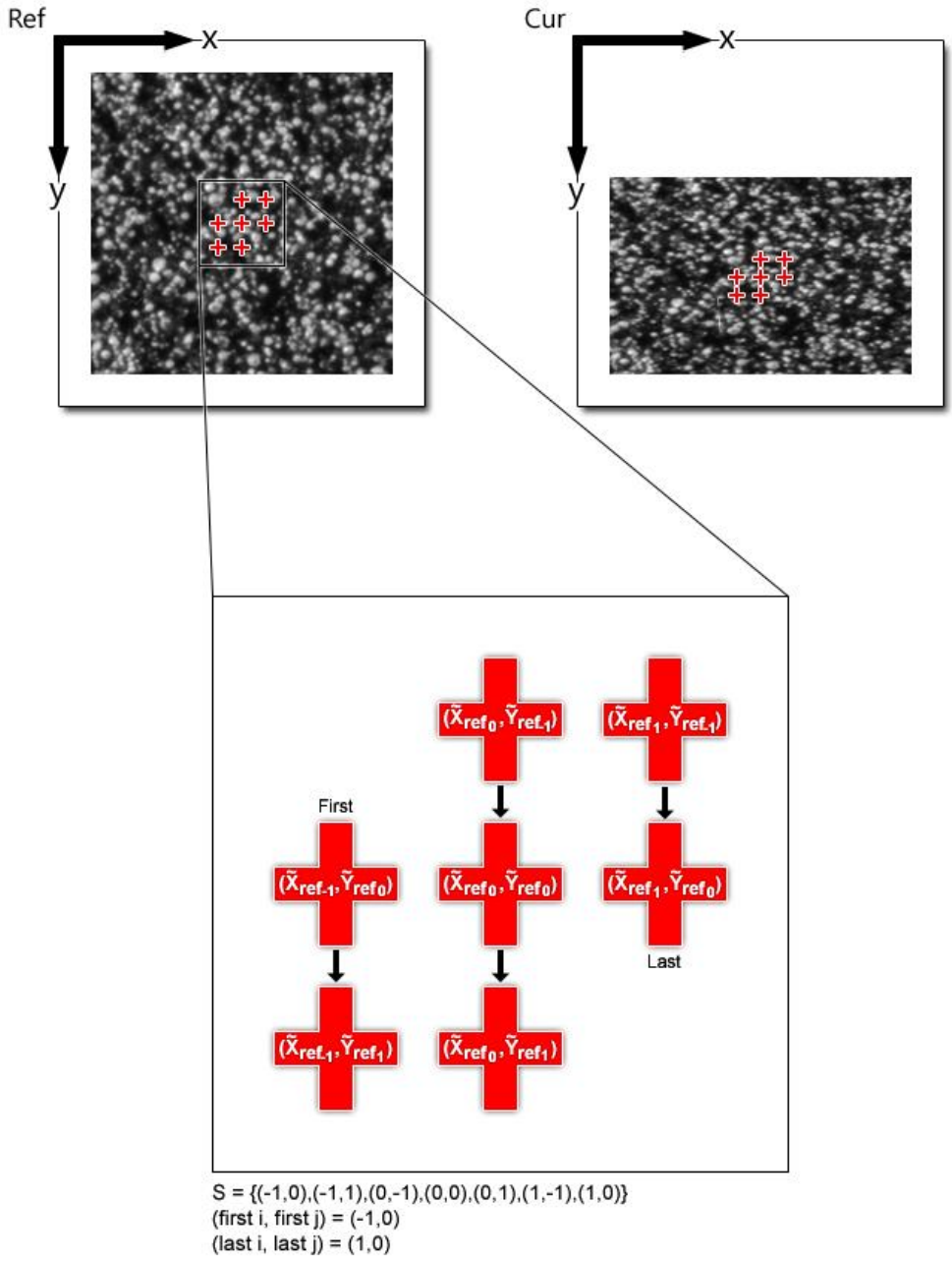


Figura 1. Las coordenadas del subconjunto se muestran como cruces rojas. El subconjunto no necesita ser cuadrado, como se muestra arriba; la forma puede ser arbitraria siempre que contenga un punto central.

La transformación de puntos de subconjuntos de referencia inicial a la configuración actual está normalmente restringida a una transformación de primer orden lineal (aunque las transformaciones de segundo orden también pueden ser usadas si se prevén deformaciones grandes como se muestra a continuación).

$$\in S \begin{cases} \hat{x}_{curi} = x_{refi} + u_{rc} + \dfrac{\partial u}{\partial x_{rc}}\left(x_{refi} - x_{refc}\right) + \dfrac{\partial u}{\partial y_{rc}}\left(y_{refi} - y_{refc}\right) \\ \hat{y}_{curj} = y_{refj} + v_{rc} + \dfrac{\partial v}{\partial x_{rc}}\left(x_{refi} - x_{refc}\right) + \dfrac{\partial v}{\partial y_{rc}}\left(y_{refj} - y_{refc}\right) \end{cases}^{(i,j)} \quad \text{(Ecuación 197)}$$

$$p = \left\{u \;\; v \;\; \frac{\partial u}{\partial x} \;\; \frac{\partial u}{\partial y} \;\; \frac{\partial v}{\partial x} \;\; \frac{\partial v}{\partial y}\right\}^{7} \quad \text{(Ecuación 198)}$$

Donde $Xref_i$ e $Yref_j$ son las coordenadas X e Y del punto del subconjunto de la referencia inicial, $Xref_c$ e $Yref_c$ son las coordenadas X e Y del centro del subconjunto de referencia inicial, $\hat{x}_{curi}, \hat{y}_{curj}$ son las coordenadas X e Y del punto final del subconjunto actual, (i,j) son los índices usados para la localización relativa de los puntos del subconjunto con respecto al centro del subconjunto, así como para ver las correspondencias entre los puntos del subconjunto en la configuración actual y de referencia. S es el conjunto que contiene todos los puntos del subconjunto (tenemos un ejemplo en la figura 1).

El subíndice rc usado en la ecuación 197 pretende significar que la transformación es desde la de referencia al sistema de coordenadas actual. Además, $\left\{u \;\; v \;\; \frac{\partial u}{\partial x} \;\; \frac{\partial u}{\partial y} \;\; \frac{\partial v}{\partial x} \;\; \frac{\partial v}{\partial y}\right\}^{7}$ es la forma general del vector de deformación p, como se define en la ecuación 198 (su efecto sobre los subconjuntos se muestra en la figura 2). Por último, la ecuación 197 puede ser escrita en forma de matriz como se muestra a continuación:

$$\xi_{reci} + w\left(\Delta\xi_{ref;p_{rc}}\right) = \begin{Bmatrix} x^t_{refc} \\ y^t_{refc} \\ 1 \end{Bmatrix} + \begin{bmatrix} 1 + \dfrac{du}{dx_{rc}} & \dfrac{du}{dy_{rc}} & u_{rc} \\ \dfrac{dv}{dx_{rc}} & 1 + \dfrac{dv}{dy_{rc}} & v_{rc} \\ 0 & 0 & 1 \end{bmatrix} * \begin{Bmatrix} \Delta x^t_{ref} \\ \Delta y^t_{ref} \\ 1 \end{Bmatrix} \quad \text{(Ecuación 199)}$$

Donde ξ es un vector ampliado que contiene las coordenadas X e Y de los puntos del subconjunto, Δx e Δy son las distancias entre un punto del subconjunto y el centro del subconjunto, "w" es una función llamada "warp" (deformar, alabear, torcer). Para fines de cálculo, también permitimos la deformación dentro de la configuración de referencia, como se muestra a continuación:

$$\in S \begin{cases} \hat{x}_{refi} = x_{refi} + u_{rr} + \dfrac{\partial u}{\partial x_{rr}}\left(x_{refi} - x_{refc}\right) + \dfrac{\partial u}{\partial y_{rr}}\left(y_{refi} - y_{refc}\right) \\ \hat{y}_{refj} = y_{refj} + v_{rr} + \dfrac{\partial v}{\partial x_{rr}}\left(x_{refi} - x_{refc}\right) + \dfrac{\partial v}{\partial y_{rr}}\left(y_{refj} - y_{refc}\right) \end{cases}^{(i,j)} \quad \text{(Ecuación 200)}$$

Donde $\hat{x}_{refi}, \hat{y}_{refj}$ son las coordenadas X e Y del punto final del subconjunto de referencia. El subíndice "rr" es necesario para entender que la transformación es de la referencia del sistema de coordenadas para el sistema de coordenadas de referencia.

Para nuestros propósitos, queremos encontrar el p_{rc} óptimo, cuando $p_{rr} = 0$, de modo que las coordenadas $\hat{x}_{refi}, \hat{y}_{refj}$ coinciden mejor con las coordenadas en $\hat{x}_{curi}$ y $\hat{y}_{curj}$ respectivamente. Denotamos el valor p_{rc} como p_{rc}^{*}. Por ahora, puede parecer un sin sentido usar la ecuación 200 y sólo restringir el subconjunto de coordenadas de referencia de forma que sean indeformables. Pero, uno de los métodos que eventualmente usaremos (el método compositivo inverso) requiere las coordenadas del subconjunto de referencia para ser deformables, por lo que introducimos la notación ahora para suavizar la derivación posterior.

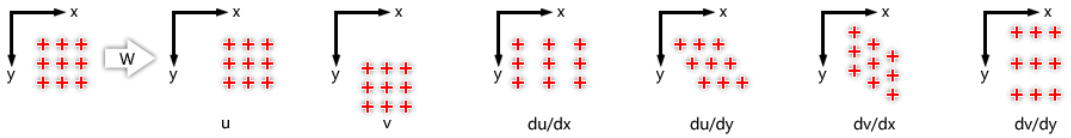


Figura 2. Esta figura muestra las transformaciones lineales para las coordenadas del subconjunto. Cualquier combinación lineal de los 6 parámetros mostrados anteriormente se puede usar para deformar las coordenadas del subconjunto a través de la función warp.

4.2.1 Criterios de correlación

El siguiente paso es establecer una medida de similitud entre la referencia final del subconjunto y el subconjunto final actual. Esto es llevado a cabo comparando los valores de la escala de grises en el final de referencia de los puntos del subconjunto con los valores de grises de los puntos finales del subconjunto actual. Las siguientes dos formas de medida son las más comunes usadas en DIC:

$$C_{CC} = \frac{\sum(i,j) \in S(f(\hat{x}_{refi}, \hat{y}_{refj}) - f_m)(g(\hat{x}_{curi}, \hat{y}_{curj}) - g_m)}{\sqrt{\sum(i,j) \in S[f(\hat{x}_{refi}, \hat{y}_{refj}) - f_m]^2 \sum(i,j) \in S[g(\hat{x}_{curi}, \hat{y}_{curj}) - g_m]^2}} \quad \text{(Ecuación 201)}$$

$$C_{LS} = \sum_{(i,j)\in S} \left[\frac{f(\hat{x}_{refi}, \hat{y}_{refj}) - f_m}{\sqrt{\sum_{(i,j)\in S}[f(\hat{x}_{refi}, \hat{y}_{refj}) - f_m]^2}} - \frac{g(\hat{x}_{curi}, \hat{y}_{curj}) - g_m}{\sqrt{\sum(i,j) \in S[g(\hat{x}_{curi}, \hat{y}_{curj}) - g_m]^2}} \right]^2 \quad \text{(Ecuación 202)}$$

Donde f y g son la referencia y la función de la imagen actual, respectivamente, y devuelve un valor de la escala de grises correspondiente a los puntos específicos (x, y), f_m y g_m son el valor medio de los valores de la escala de grises del final de referencia y del subconjunto actual respectivamente, son definidos por:

$$f_m = \frac{\sum_{(i,j)\in S} f(\hat{x}_{refi}, \hat{y}_{refj})}{n(s)} \quad \text{(Ecuación 203)}$$

$$g_m = \frac{\sum_{(i,j)\in S} g(\hat{x}_{curi}, \hat{y}_{curj})}{n(s)} \quad \text{(Ecuación 204)}$$

Donde $n(S)$ es el número de elementos en S.

La ecuación 201 es el **criterio de correlación cruzada normalizada** e indica cuando C_{CC} es cercano a la unidad. Este criterio es el que usaremos en la parte práctica de nuestro proyecto.

La ecuación 202 es el **criterio normalizado de mínimos cuadrados** e indica cuando C_{LS} es cercano a cero.

La resta de la componente media (f_m y g_m) en las ecuaciones 201 y 202 permite los criterios para ser invariante a los cambios en los valores de escala de grises, mientras que la división por la cantidad en el denominador permite invariancia con respecto a los valores de escalar la escala de grises; el resultado final es un criterio de correlación que no varían ante cambios en los valores de escala de grises. En la actualidad, estos criterios de correlación se relacionan directamente, pero cada criterio de correlación respectivo es más fácil de calcular en ciertas situaciones que es por lo que ambas se enumeran. Un ejemplo sencillo de cómo se utilizan se da en la figura 3.

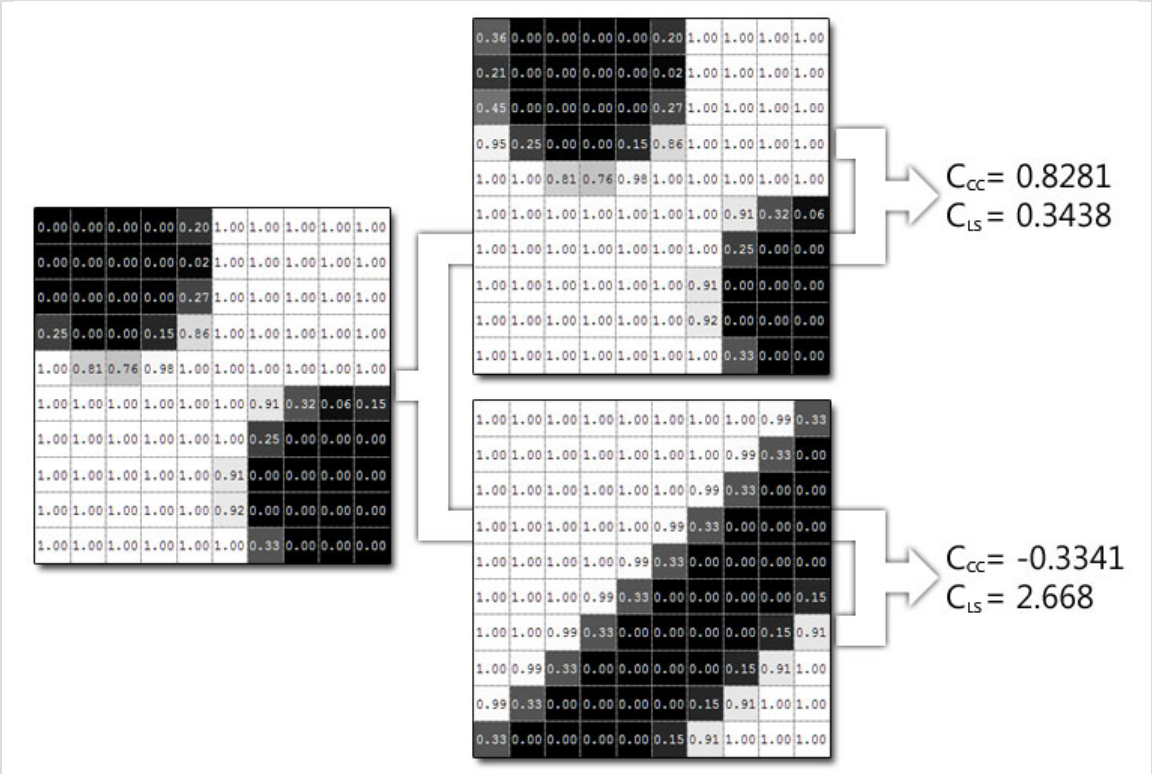


Figura 3. Demuestra la eficacia de los dos criterios de correlación. Los valores en escala de grises de la izquierda se comparan con los dos de la derecha. Los criterios de correlación muestran que los valores de arriba a la derecha de la escala de grises coinciden con el conjunto de la izquierda (C_{CC} es cercano a 1 y C_{LS} es cercano a 0), que es evidente a través de la inspección visual.

Los pasos generales para la utilización de estos criterios de correlación se describen a continuación:

1. Formar un subconjunto inicial en la configuración de referencia, por lo general en lugares de píxeles enteros. Aplicar una deformación, con p_{rr}, al subconjunto de referencia, a la muestra f inicial, a los valores de escala de grises de referencia, en estos puntos y almacenar estos valores en una matriz

2. Aplicar una deformación, con p_{rc}, al subconjunto de referencia inicial para llevarlo a la configuración actual, a la muestra g, los valores actuales de la escala de grises en estos puntos y almacenarlos en una matriz de tamaño equivalente como en el paso 1 (hay que tener en cuenta que el almacenamiento de estos puntos es esencialmente una transformación inversa y permite una correspondencia / comparación directa entre los valores de escala de grises en la referencia y la configuración actual).

3. Ahora podemos comparar estas dos matrices usando cualquiera de los dos criterios de correlación de las ecuaciones 201-202. Este proceso se describe en la siguiente figura:

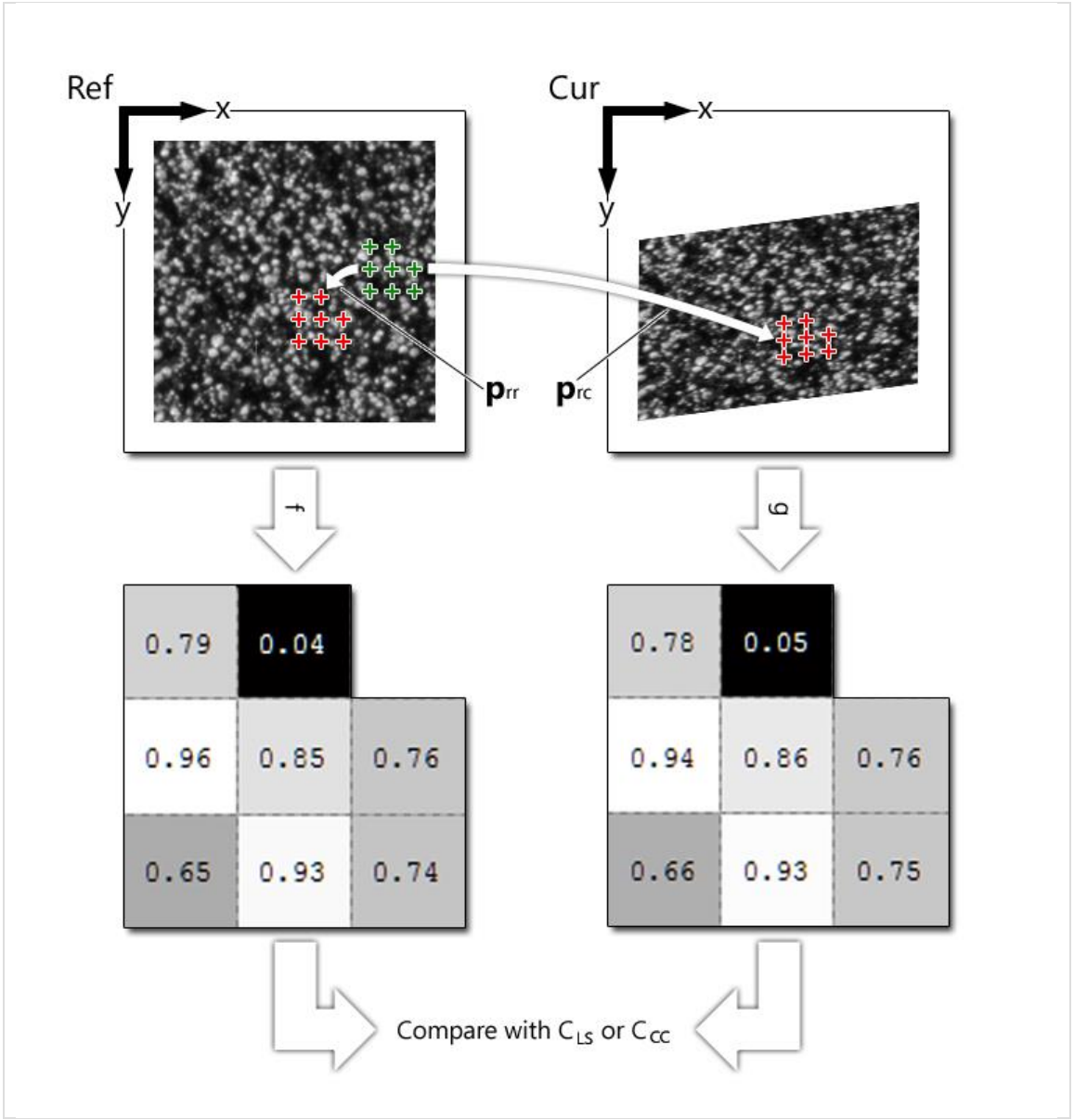


Figura 4. Esta figura es un ejemplo gráfico de los pasos descritos anteriormente. El subconjunto verde en la imagen de referencia es el subconjunto de referencia inicial. Los subconjuntos rojos marcan las ubicaciones de subconjuntos finales en la referencia y la configuración actual.

Hay que tener en cuenta que si p_{rr} y p_{rc} se encuentran de manera que los subconjuntos finales coinciden bien y p_{rr} es pequeña en magnitud de traslación; entonces, se puede aproximar cogiendo p_{rc} y componiéndolo con el inverso de p_{rr}. Imaginemos esto como una transformación directa desde el subconjunto de referencia final hasta el subconjunto final actual. Esta idea es importante para la comprensión del algoritmo de composición inversa que se discutirá más adelante.

De todos modos, el siguiente paso del análisis es cómo establecer un método para encontrar de forma automática. La principal aplicación en DIC es utilizar un esquema iterativo no lineal de optimización de mínimos cuadrados que minimiza la ecuación 202

(hay que tener en cuenta que la ecuación 202 es estrictamente positiva y un número menor indica una mejor coincidencia).

4.3 Optimización no lineal

La sección de optimización no lineal se divide en tres partes principales:

1. Proporcionar una estimación inicial.

2. El esquema de Gauss-Newton (GN) de optimización iterativo.

3. Interpolación.

Se requiere una estimación inicial, porque los esquemas de optimización iterativos convergen a un máximo / mínimo local; por lo tanto, se requiere una estimación inicial cerca del máximo / mínimo global. En la sección del esquema de optimización iterativo GN, se discuten dos esquemas diferentes: el método de Gauss-Newton aditivo hacia adelante (FA-GN), y el método de Gauss-Newton de composición inversa (IC-GN).

El método FA-GN se discute en primer lugar puesto que es un algoritmo utilizado comúnmente en DIC (y en la optimización en general). Por otro lado, el método IC-GN es un caso especial de optimización no lineal (comúnmente utilizado en los algoritmos de alineación de la imagen), que es más rápido que el método estándar FA-GN y fue aplicado recientemente a DIC.

Por último, la interpolación se analiza porque se requieren valores de escala de grises y los gradientes en las localizaciones del subpíxel para calcular las cantidades utilizadas en los métodos FA-GN y IC-GN.

A continuación se muestra el flujo general de un esquema de optimización no lineal:

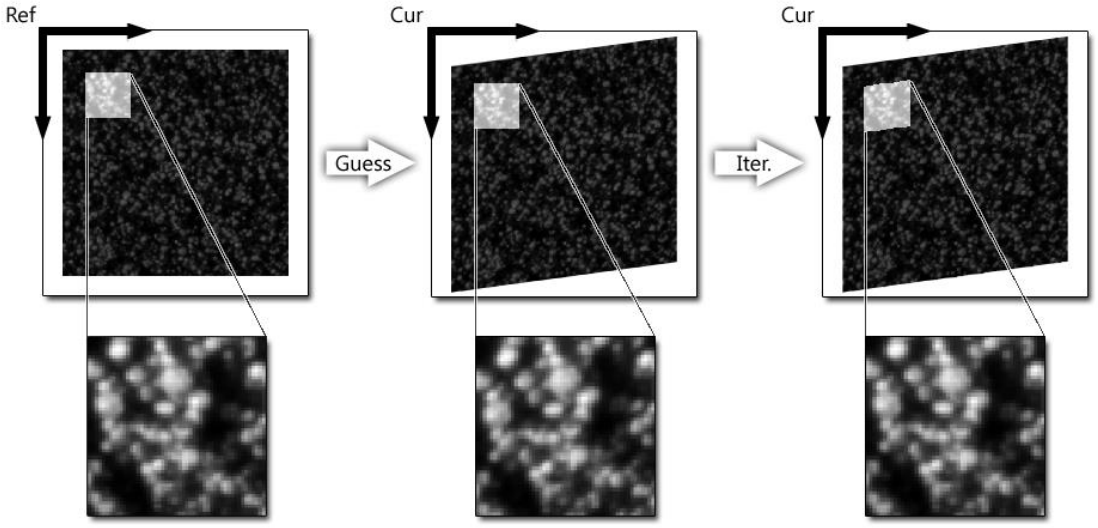


Figura 5. Muestra los pasos básicos del esquema de optimización no lineal utilizado en DIC. El primer paso es obtener la estimación inicial. Esta suposición alimenta como

entrada inicial al esquema de optimización iterativo, que proporciona resultados más precisos, como se muestra en la última imagen de la derecha.

4.3.1. La estimación inicial

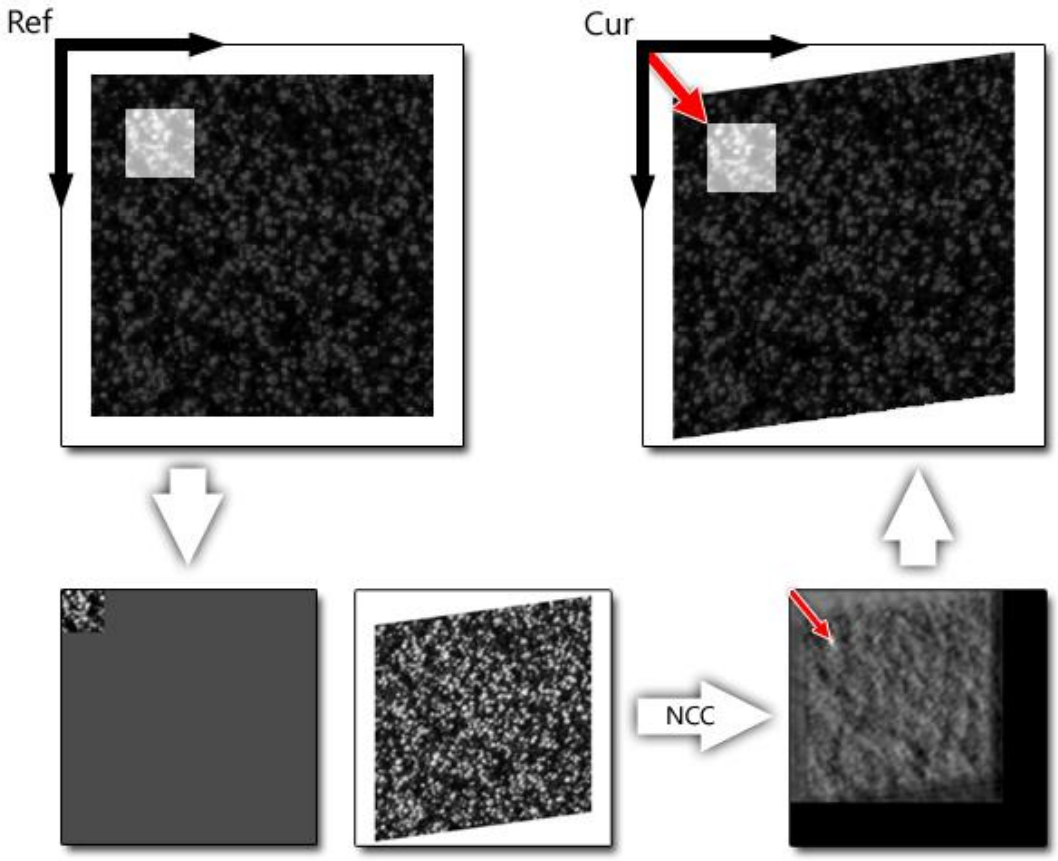


Figura 6. Muestra la correlación cruzada normalizada (ecuación 201)

La imagen superior izquierda muestra la selección de un subconjunto de referencia. El subconjunto se rellena a continuación, en la imagen de la parte inferior izquierda y es alimentada como entrada junto con la imagen actual a una función de correlación cruzada normalizada. La salida, en la parte inferior derecha, es un conjunto de valores de los coeficientes de correlación. La flecha roja apunta al valor más alto (que se muestra como un pico blanco); debido a la manera en que el subconjunto se construye y se rellena en este ejemplo, este punto se refiere a la ubicación de la esquina superior izquierda en la configuración actual. Por último, se muestra la ubicación del subconjunto con respecto a la configuración actual.

Hay varios métodos para obtener una estimación inicial para el esquema de optimización iterativo no lineal. El método más común y establecido, es la correlación cruzada normalizada rápida (NCC). Este método utiliza el coeficiente de correlación de la ecuación 201 y lo calcula en cada ubicación del píxel en la configuración actual de una manera computacional eficiente. Utiliza la FFT para calcular las partes no precomputables del numerador y, a continuación, utiliza una tabla de área sumada para

calcular las partes no precomputables del denominador. La FFT se puede utilizar para la obtención de los valores de escala de grises dentro del subconjunto de referencia y rellenar luego con ceros estos valores para igualar al mismo tamaño que la imagen actual (véase la parte inferior izquierda de la figura 6).

Al final del proceso, se obtiene una matriz de coeficientes de correlación; las coordenadas del valor más alto del coeficiente de correlación producirán la ubicación del subconjunto en la configuración actual. Es importante tener en cuenta que el método NCC sólo puede determinar desplazamientos enteros (es decir, números enteros para u y v) y por lo tanto no pueden proporcionar estimaciones iniciales para los otros cuatro parámetros en p. Así, el método de correlación cruzada normalizada es el más adecuado para proporcionar aproximaciones iniciales en las zonas de deformación relativamente baja (es decir, no hay grandes rotaciones). El proceso se describe en la figura 6.

Hay que tener en cuenta que NCC no es el único método disponible para proporcionar una estimación inicial. Más recientemente, Bing Pan ha introducido un algoritmo basado en SIFT, que puede proporcionar una estimación inicial de los 6 parámetros de deformación mediante la formación de correspondencias entre puntos de características SIFT en la referencia y la configuración actual. Los puntos característicos que se encuentran dentro de una región de referencia seleccionada y sus correspondientes ubicaciones en la configuración actual, se pueden formular en un sistema lineal sobre-limitado y luego resolver usando un solucionador lineal de mínimos cuadrados.

4.3.2. El método de mínimos cuadrados iterativo no lineal de Gauss-Newton

El método de Gauss-Newton se utiliza para encontrar las raíces de una función en el caso de que una solución analítica no esté disponible. Este escenario se puede extrapolar a la optimización de la búsqueda de las raíces de la derivada de una función. Aún más, puede ser generalizado a optimización multivariante utilizando el gradiente en lugar de la derivada y luego determinar donde la norma del gradiente converge a cero.

La función, C_{LS} (ecuación 201), utilizada en el procedimiento FA-GN y IC-GN, se define aquí como una función que acepta un solo argumento, p, que puede ser aplicado ya sea como p_{rr} o p_{rc}. Cómo se aplica el argumento depende de qué método (FA-GN o IC-GN) se está utilizando y se hará evidente en las secciones respectivas.

De todos modos, la forma general de la ecuación iterativa utilizada en este análisis se puede derivar tomando el polinomio de Taylor de segundo orden de C_{LS} alrededor de p_0, donde p_0 contiene los parámetros de deformación de partida (es decir, alguna forma de estimación inicial) y, a continuación, determinar donde su derivada con respecto a Δp es igual al vector cero.

$$C_{LS}(p_0 + \Delta p) \approx C_{LS}(p_0) + \nabla C_{LS}(p_0)^t * \Delta p + \frac{1}{2} * \Delta p^t * \nabla\nabla C_{LS}(p_0) * \Delta p \qquad \text{(Ecuación 205)}$$

$$\frac{dC_{LS}(p_0 + \Delta p)}{d\Delta p} \approx \nabla C_{LS}(p_0) + \nabla\nabla C_{LS}(p_0) * \Delta p = 0 \qquad \text{(Ecuación 206)}$$

Donde $\nabla C_{LS}(p_0)$ es el gradiente de C_{LS} en p_0 y $\nabla\nabla C_{LS}(p_0)$ es la hessiana de C_{LS} en p_0. La idea básica detrás de esto es que el polinomio de Taylor nos permite formar una aproximación a la función C_{LS} alrededor p_0 con otra función cuyo mínimo que podemos encontrar analíticamente (tomando su derivada y explícitamente despejando para Δp). Esto nos permitirá resolver para $p_0 + \Delta p$, que debería estar más cerca de la solución. A continuación, podemos repetir hasta que encontremos una solución más cercana. La forma general de la ecuación de optimización iterativa utilizada en este análisis se reorganiza desde la ecuación 206 y se muestra a continuación:

$$\nabla\nabla C_{LS}(p_0) * \Delta p = -\nabla C_{LS}(p_0) \qquad \text{(Ecuación 207)}$$

La ecuación 207 es en realidad el esquema iterativo de Newton-Raphson. Se convierte en el esquema iterativo de Gauss-Newton mediante el uso de una aproximación a la matriz hessiana (es una matriz de segundas derivadas parciales), que se elabora en las secciones en FA-GN y IC-GN específicas.

4.3.3 Método aditivo hacia adelante

El método FA-GN es esencialmente la aplicación estándar de un solucionador iterativo de Gauss-Newton. En esta situación, la ubicación final del subconjunto de referencia permanece constante y p_{rr} se establece en 0 para cada iteración. Por otro lado, la ubicación final del subconjunto actual es permitida para deformar y p_{rc} se establece en p_{old} al comienzo de cada iteración, donde p_{old} son los parámetros de deformación encontrados de la iteración previa, utilizando la técnica de adición hacia adelante, o la estimación inicial. Esto significa que p_0 de ecuación 207 equivale a p_{old}.

La forma compacta de C_{LS} para este método es:

$$C_{LS}(p_{old} + \Delta p) = \sum \left[\frac{f\left(\xi_{refc} + w(\Delta\xi_{ref}; 0)\right) - f_m}{\sqrt{\sum\left[f\left(\xi_{refc} + w(\Delta\xi_{ref}; 0)\right) - f_m\right]^2}} - \frac{g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old} + \Delta p)\right) - g_m}{\sqrt{\sum\left[g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old} + \Delta p)\right) - g_m\right]^2}} \right]^2 \qquad \text{(Ecuación 208)}$$

Donde $f\left(\xi_{refc} + w(\Delta\xi_{ref}; 0)\right)$ representa un vector en escala de grises de la imagen de referencia muestreada y $g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old} + \Delta p)\right)$ representa un vector de valores en escala de grises de la imagen actual muestreada.

Además, por razones de simplicidad, suponemos que:

$$\frac{d}{d\Delta p}(g_m) \approx 0 \qquad \text{(Ecuación 209)}$$

$$\frac{d}{d\Delta p}\left(\sqrt{\sum\left[g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old})\right) - g_m\right]^2}\right) \approx 0 \qquad \text{(Ecuación 210)}$$

Cuáles son las suposiciones razonables si los valores medios en escala de grises del subconjunto no varían tanto como el subconjunto es deformado por p; estos supuestos también se han verificado empíricamente para ser aceptables. El gradiente para este caso se encuentra que es:

$$\nabla C_{LS}(p_{old}) = \frac{dC_{LS}(p_{old})}{d\Delta p}$$
$$\approx \frac{-2}{\sqrt{\sum\left[g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old})\right) - g_m\right]^2}} \sum\left[\left[\frac{f\left(\xi_{refc} + w(\Delta\xi_{ref}; 0)\right) - f_m}{\sqrt{\sum\left[f\left(\xi_{refc} + w(\Delta\xi_{ref}; 0)\right) - f_m\right]^2}}\right.\right.$$
$$\left.\left. - \frac{g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old})\right) - g_m}{\sqrt{\sum\left[g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old})\right) - g_m\right]^2}}\right]\left[\frac{d}{d\Delta p} g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old})\right)\right]\right]$$

(Ecuación 211)

La hessiana es:

$$\nabla\nabla C_{LS}(p_{old}) = \frac{d^2 C_{LS}(p_{old})}{d\Delta p^2}$$
$$\approx \frac{-2}{\sqrt{\sum\left[g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old})\right) - g_m\right]^2}}$$

$$\left\{\sum\left[-\frac{\frac{d}{d\Delta p}g\left(\xi_{refc}+w(\Delta\xi_{ref};p_{old})\right)}{\sqrt{\sum\left[g\left(\xi_{refc}+w(\Delta\xi_{ref};p_{old})\right)-g_m\right]^2}}\right]\left[\frac{d}{d\Delta p}g\left(\xi_{refc}+w(\Delta\xi_{ref};p_{old})\right)\right]^t\right.$$
$$+\sum\left[\frac{f\left(\xi_{refc}+w(\Delta\xi_{ref};0)\right)-f_m}{\sqrt{\sum\left[f\left(\xi_{refc}+w(\Delta\xi_{ref};0)\right)-f_m\right]^2}}\right. \qquad \text{(Ecuación 212)}$$
$$\left.\left.-\frac{g\left(\xi_{refc}+w(\Delta\xi_{ref};p_{old})\right)-g_m}{\sqrt{\sum\left[g\left(\xi_{refc}+w(\Delta\xi_{ref};p_{old})\right)-g_m\right]^2}}\right]\left[\frac{d^2}{d\Delta p^2}g\left(\xi_{refc}+w(\Delta\xi_{ref};p_{old})\right)\right]\right\}$$

Esta es la hessiana para las iteraciones de Newton-Raphson. Se convierte en la hessiana de Gauss-Newton con la siguiente hipótesis:

$$\sum\left[\frac{f\left(\xi_{refc}+w(\Delta\xi_{ref};0)\right)-f_m}{\sqrt{\sum\left[f\left(\xi_{refc}+w(\Delta\xi_{ref};0)\right)-f_m\right]^2}}\right.$$
$$\left.-\frac{g\left(\xi_{refc}+w(\Delta\xi_{ref};p_{old})\right)-g_m}{\sqrt{\sum\left[g\left(\xi_{refc}+w(\Delta\xi_{ref};p_{old})\right)-g_m\right]^2}}\right]\left[\frac{d^2}{d\Delta p^2}g\left(\xi_{refc}+w(\Delta\xi_{ref};p_{old})\right)\right]\approx 0 \qquad \text{(Ecuación 213)}$$

Esta es una suposición razonable si p_{old} es cercano a la solución óptima (esto hace que la primera cantidad entre paréntesis se acerque a cero) y también si las segundas derivadas de los valores de la escala de grises son cercanos a cero (esto depende de la imagen y de la interpolación utilizada; pero en general, los valores en escala de grises deben ser bastante suaves por lo que esta cantidad debe ser pequeña). Las dos cantidades pequeñas que se multiplican entre sí en la ecuación 213 dan como resultado en una cantidad pequeña en magnitud.

La suposición anterior no viene sin un costo. El método de Gauss-Newton en realidad toma más iteraciones para converger que el método estándar de Newton-Raphson. Sin embargo, una sola iteración de Gauss-Newton cuesta menos que una sola iteración de Newton-Raphson. Por lo tanto, una mejora en la velocidad sólo se alcanzará si el coste de las iteraciones de Gauss-Newton es lo suficientemente barato como para compensar el coste agregado de más iteraciones. En nuestra experiencia, este es el caso para los subgrupos más pequeños, aunque para los subgrupos más grandes del método de

Newton-Raphson se vuelve más rápido. Fuera de la velocidad de cálculo, otra de las razones para poner en práctica el método de Gauss-Newton es que en realidad tiene mejores características de convergencia que el método de Newton-Raphson. También es más fácil de implementar que el método de Newton-Raphson.

De todos modos, esto produce una forma final de la hessiana como:

$$\nabla\nabla C_{LS}(p_{old}) = \frac{d^2 C_{LS}(p_{old})}{d\Delta p^2} \approx \frac{2}{\sqrt{\sum\left[g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old})\right) - g_m\right]^2}} \sum_{(i,j)\in S} \left[\frac{\partial}{\partial \Delta p} g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old})\right)\right]\left[\frac{\partial}{\partial \Delta p} g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old})\right)\right]^t \quad \text{(Ecuación 214)}$$

Tengamos en cuenta que para que el gradiente y la hessiana puedan ser calculables, es necesaria la interpolación y este procedimiento de interpolación pasa a ser una de las partes más costosas del proceso de cálculo, especialmente si se usa un alto orden en la interpolación.

El método general de las iteraciones FA-GN es calcular el gradiente y la hessiana, luego calcular Δp usando las ecuaciones 207, 211 y 214 con la descomposición de Cholesky, ya que la hessiana de Gauss-Newton debe ser definida como simétrica positiva. Este incremento; Δp, se utiliza entonces para actualizar p_{old} de la siguiente manera:

$$p_{new} = p_{old} + \Delta p \quad \text{(Ecuación 215)}$$

Donde p_{old} se configura en p_{new} al comienzo de cada iteración. Este método se llama el método aditivo hacia adelante debido a cómo se actualiza p_{old} como se muestra arriba (que tendrá más sentido de porqué hacemos esta distinción una vez que se discuta el método IC-GN).

Es importante tener en cuenta que la parte más costosa de los cálculos son las evaluaciones de g y dg/dΔp en $\xi_{refc} + w(\Delta\xi_{ref}; p_{old})$. El procedimiento iterativo puede ser reformulado para ser mucho más eficiente, lo que se explica con más detalle en la sección del método IC-GN.

De todos modos, el flujo general del método de FA-GN se da en la figura 7:

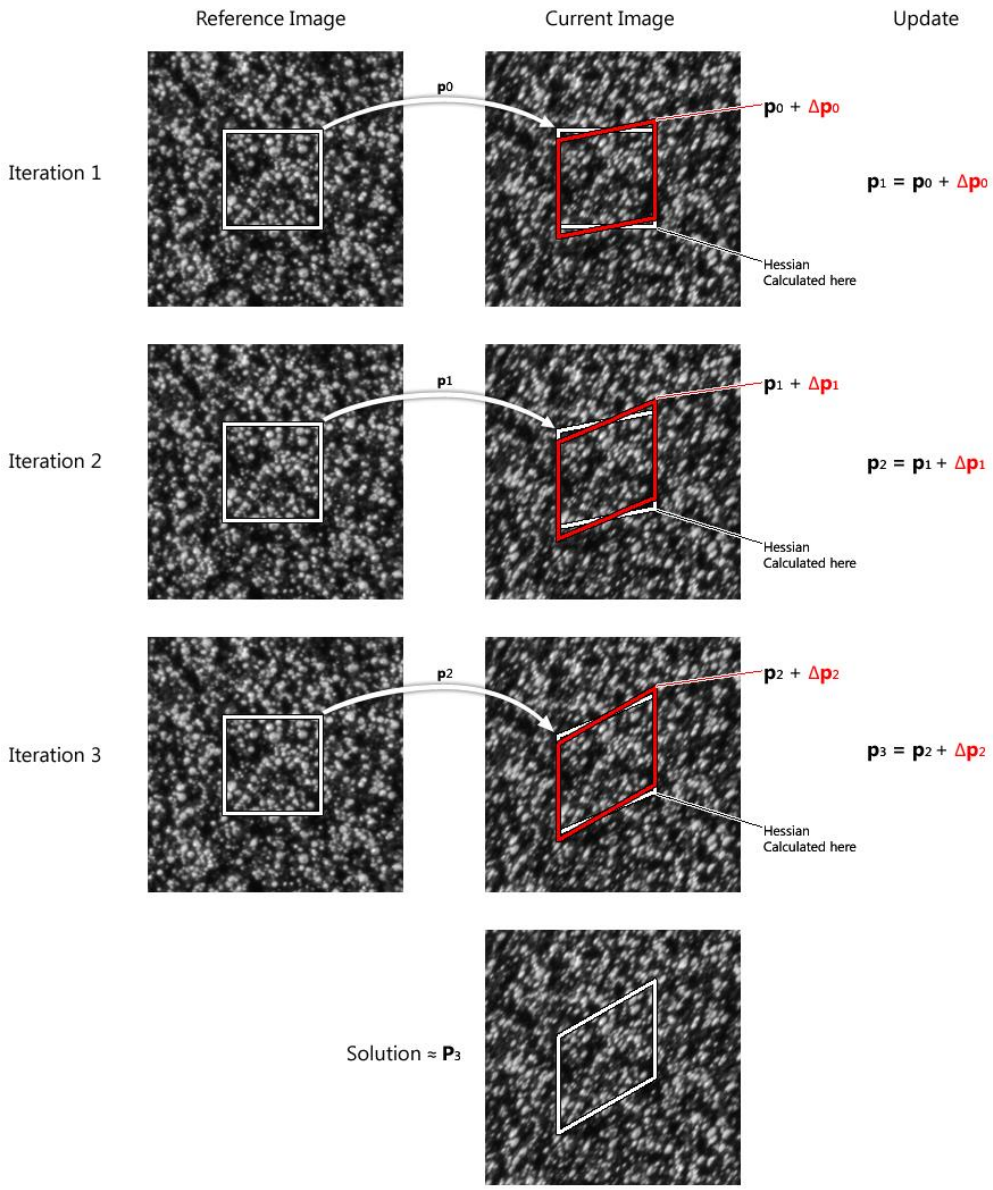


Figura 7. Esta es una representación gráfica del método FA-GN para la alineación de la imagen. La imagen de referencia tiene una cizalla aplicada. Lo más importante a señalar aquí es donde se evalúa la hessiana en $\xi_{refc} + w(\Delta\xi_{ref}; p_{old})$ en la imagen actual para cada iteración. Esta cantidad es el cuello de botella computacional del algoritmo y proporciona una buena percepción para la comprensión de la eficacia del método IC-GN.

4.3.4 Método de composición inverso

En este método, un par de cosas del método FA-GN cambian. En primer lugar, se permite variar la ubicación final del subconjunto de referencia, pero p_{rr} se configura a 0 al comienzo de cada iteración. Esto significa que p_o de la ecuación 207 es igual a 0. Por otra parte, no se permite la variación de la ubicación final del subconjunto actual, pero se ajusta a p_{old} al comienzo de cada iteración, p_{old} es la actualización de los parámetros para p_{rc} encontrados en la iteración anterior utilizando la técnica de composición inversa, que se explica más adelante.

La motivación para hacer este cambio es que la hassiana se calcula utilizando el subconjunto de referencia que comienza en el mismo lugar en cada iteración; es decir, la hassiana será la misma para cada iteración, por lo que sólo necesita ser calculada una vez. Esta idea se hará más clara en los párrafos siguientes.

La forma compacta de C_{LS} para este método es:

$$C_{LS}(\Delta p) = \sum \left[\frac{f\left(\xi_{refc} + w(\Delta\xi_{ref}; 0)\right) - f_m}{\sqrt{\sum\left[f\left(\xi_{refc} + w(\Delta\xi_{ref}; 0)\right) - f_m\right]^2}} - \frac{g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old})\right) - g_m}{\sqrt{\sum\left[g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old})\right) - g_m\right]^2}} \right]^2 \quad \text{(Ecuación 216)}$$

Los supuestos sobre los promedios y las cantidades, en el fondo siguen siendo los mismos:

$$\frac{d}{d\Delta p}(f_m) \approx 0 \quad \text{(Ecuación 217)}$$

$$\frac{d}{d\Delta p}\left(\sqrt{\sum\left[f\left(\xi_{refc} + w(\Delta\xi_{ref}; 0)\right) - f_m\right]^2}\right) \approx 0 \quad \text{(Ecuación 218)}$$

El gradiente para este caso es:

$$\nabla C_{LS}(0) = \frac{dC_{LS}(0)}{d\Delta p} \approx \frac{2}{\sqrt{\sum\left[f\left(\xi_{refc} + w(\Delta\xi_{ref}; 0)\right) - f_m\right]^2}} \sum \left[\left[\frac{f\left(\xi_{refc} + w(\Delta\xi_{ref}; 0)\right) - f_m}{\sqrt{\sum\left[f\left(\xi_{refc} + w(\Delta\xi_{ref}; 0)\right) - f_m\right]^2}} - \frac{g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old})\right) - g_m}{\sqrt{\sum\left[g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old})\right) - g_m\right]^2}} \right] \left[\frac{d}{d\Delta p} f\left(\xi_{refc} + w(\Delta\xi_{ref}; 0)\right) \right] \right] \quad \text{(Ecuación 219)}$$

La Hessiana es:

$$\nabla\nabla C_{LS}(0) = \frac{d^2 C_{LS}(0)}{d\Delta p^2}$$

$$\approx \frac{2}{\sqrt{\sum\left[f\left(\xi_{refc} + w(\Delta\xi_{ref};0)\right) - f_m\right]^2}} \left\{ \sum \left[\frac{\frac{d}{d\Delta p} f\left(\xi_{refc} + w(\Delta\xi_{ref};0)\right)}{\sqrt{\sum\left[f\left(\xi_{refc} + w(\Delta\xi_{ref};0)\right) - f_m\right]}} + w(\Delta\xi_{ref};0)) \right]^t \right.$$

$$+ \sum \left[\frac{f\left(\xi_{refc} + w(\Delta\xi_{ref};0)\right) - f_m}{\sqrt{\sum\left[f\left(\xi_{refc} + w(\Delta\xi_{ref};0)\right) - f_m\right]^2}} \right.$$

$$\left. \left. - \frac{g\left(\xi_{refc} + w(\Delta\xi_{ref};p_{old})\right) - g_m}{\sqrt{\sum\left[g\left(\xi_{refc} + w(\Delta\xi_{ref};p_{old})\right) - g_m\right]^2}} \right] \left[\frac{d^2}{d\Delta p^2} f\left(\xi_{refc} + w(\Delta\xi_{ref};0)\right) \right] \right\}$$

(Ecuación 220)

Una vez más, la suposición de Gauss-Newton es:

$$\sum \left[\frac{f\left(\xi_{refc} + w(\Delta\xi_{ref};0)\right) - f_m}{\sqrt{\sum\left[f\left(\xi_{refc} + w(\Delta\xi_{ref};0)\right) - f_m\right]^2}} \right.$$

$$\left. - \frac{g\left(\xi_{refc} + w(\Delta\xi_{ref};p_{old})\right) - g_m}{\sqrt{\sum\left[g\left(\xi_{refc} + w(\Delta\xi_{ref};p_{old})\right) - g_m\right]^2}} \right] \left[\frac{d^2}{d\Delta p^2} f\left(\xi_{refc} + w(\Delta\xi_{ref};0)\right) \right] \approx 0$$

(Ecuación 221)

Esto nos proporciona la hessiana en la forma final:

$$\nabla\nabla C_{LS}(0) \approx \frac{d^2 C_{LS}(0)}{d\Delta p^2}$$

$$\approx \frac{2}{\sqrt{\sum\left[f\left(\xi_{refc} + w(\Delta\xi_{ref};0)\right) - f_m\right]^2}} \sum \left[\frac{d}{d\Delta p} f\left(\xi_{refc} + w(\Delta\xi_{ref};0)\right) \right] \left[\frac{d}{d\Delta p} f\left(\xi_{refc} + w(\Delta\xi_{ref};0)\right) \right]^t$$

(Ecuación 222)

Δp se puede resolver usando las ecuaciones 207, 219 y 222 con la descomposición de Cholesky. El siguiente paso es averiguar qué hacer con Δp. Resulta que una aproximación más cercana a P_{rc}^*se puede encontrar tomando p_{rc} (p_{old} en este caso) y componiéndolo con el inverso de p_{rr} (Δp en este caso). El acto de hacer esto determina la transformación directa del subconjunto de referencia final al subconjunto actual final.

Una figura gráfica se muestra a continuación:

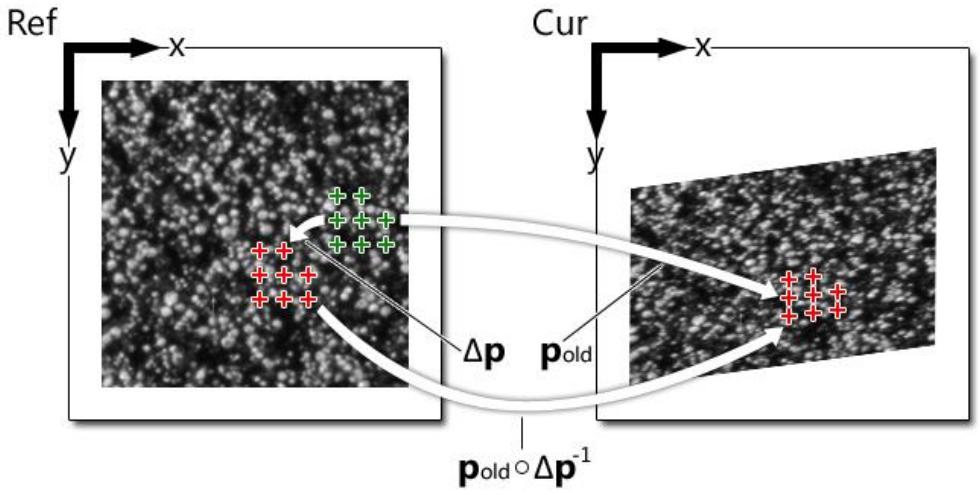


Figura 8. Esta es una representación gráfica de la actualización de la composición inversa. p_{rr} es igual a Δp y p_{rc} es igual a p_{old}. Mientras el componente de traslación de Δp es pequeño en magnitud, la composición de p_{old} y la inversa de Δp deben estar más cerca de P_{rc}^{*} que de p_{old}.

Esta actualización se puede mostrar en forma de matriz, como se muestra a continuación:

$$\begin{bmatrix} 1+\frac{du}{dx_{new}} & \frac{du}{dy_{new}} & u_{new} \\ \frac{dv}{dx_{new}} & 1+\frac{dv}{dy_{new}} & v_{new} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1+\frac{du}{dx_{old}} & \frac{du}{dy_{old}} & u_{old} \\ \frac{dv}{dx_{old}} & 1+\frac{dv}{dy_{old}} & v_{old} \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1+\Delta\frac{du}{dx} & \Delta\frac{du}{dy} & \Delta u \\ \Delta\frac{dv}{dx} & 1+\Delta\frac{dv}{dy} & \Delta v \\ 0 & 0 & 1 \end{bmatrix}^{-1} \quad \text{(Ecuación 223)}$$

$$w(\Delta\xi_{ref}; p_{new}) = w\left(w(\Delta\xi_{ref}; \Delta p)^{-1}; p_{old}\right) \quad \text{(Ecuación 224)}$$

Donde p_{old} se configurado como p_{new} al comienzo de cada iteración. El método compositivo inverso recibe su nombre por cómo se actualiza p_{old}, como se muestra en la ecuación 223. En general, es importante reiterar que inicializando el subconjunto de referencia final a una deformación 0 al comienzo de cada iteración nos permite calcular la hessiana en $\xi_{refc} + w(\Delta\xi_{ref}; 0)$ para cada iteración, lo que significa que sólo necesita ser calculada una vez.

Este método nos confundió cuando leímos por primera vez sobre el tema, por lo que hemos proporcionado algunos ejemplos a continuación para construir una mejor intuición sobre él:

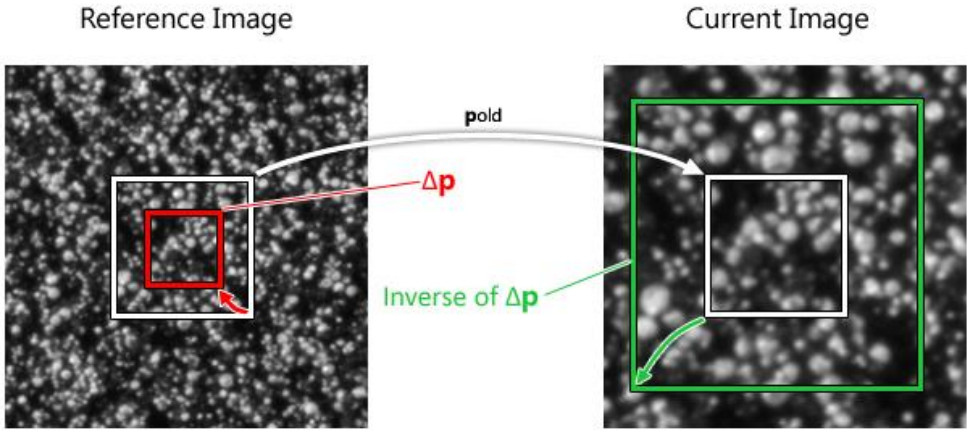


Figura 9. Esta figura demuestra lo que hay que hacer con la cantidad Δp después de que ha sido calculada. Esta figura debería dar alguna intuición sobre cómo se utiliza esta cantidad para calcular p_{new}.

Supongamos que en la figura 9, se nos ha provisto de una estimación inicial de p_{old}, que hemos ejecutado una iteración del algoritmo IC-GN y hemos encontrado Δp. Tengamos en cuenta que la imagen actual es la imagen de referencia ampliada un 100% en ambas dimensiones. Suponiendo que p_{old} es una estimación inicial sólo usando el desplazamiento y que hemos encontrado una respuesta muy cercana a la solución en una sola iteración.

Esto significa que Δp debe actuar para reducir las dimensiones del subconjunto de referencia inicial en aproximadamente un 50% (mostrado como el cuadrado rojo en el lado izquierdo de la figura 9). Para aplicar la actualización de la composición inversa correctamente (ecuación 223), vemos que hay que aplicar la inversa de Δp, seguido de p_{old}. Podemos ver la conveniencia de hacer esto, ya que la aplicación de la inversa de Δp ampliará las dimensiones de subconjunto un 100%, y luego p_{old} lo desplazará a su lugar; esto se muestra en el cuadro verde del lado derecho de la figura 9.

Mediante la comparación de la caja blanca del lado izquierdo de la figura 9 (subconjunto de referencia inicial) con el cuadro verde del lado derecho de la figura 9 (subconjunto actual final), vemos que se ha obtenido un buen emparejamiento, lo que significa que P_{rc}^{*} se ha encontrado.

Para reforzar el concepto, se da otro ejemplo a continuación:

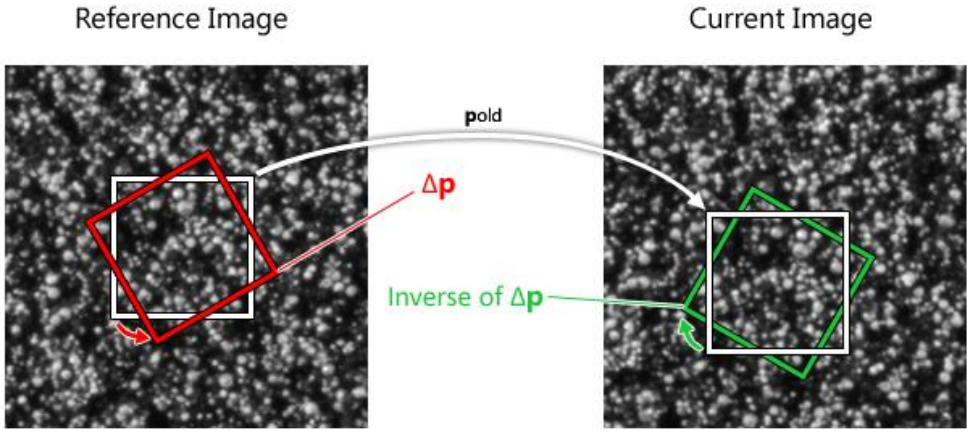


Figura 10. Esta figura está destinada a complementar la figura 9 edificando la intuición en el método de composición inversa.

El ejemplo en la figura 10 implica una rotación de 30 grados en sentido horario desde la referencia a la imagen actual. Asumiendo que p_{old} es una estimación inicial que implica un desplazamiento y también asumiendo que se ha encontrado una respuesta muy cerca de la solución después de una iteración. Como en el ejemplo anterior, es necesaria la transformación inversa de Δp (una rotación en la dirección opuesta a Δp) para actualizar p_{old} correctamente a través de la ecuación 223. Una vez más, la comparación de la caja blanca en el lado izquierdo de la figura 10 (subconjunto de referencia inicial) con el cuadro verde en el lado derecho de la figura 10 (subconjunto actual final), observamos que se obtiene un buen emparejamiento, lo que significa que P_{rc}^{*} una vez más se ha encontrado.

Esperamos que los dos ejemplos anteriores ayuden a edificar la intuición sobre el método IC-GN. A continuación se muestra el flujo general del método IC-GN:

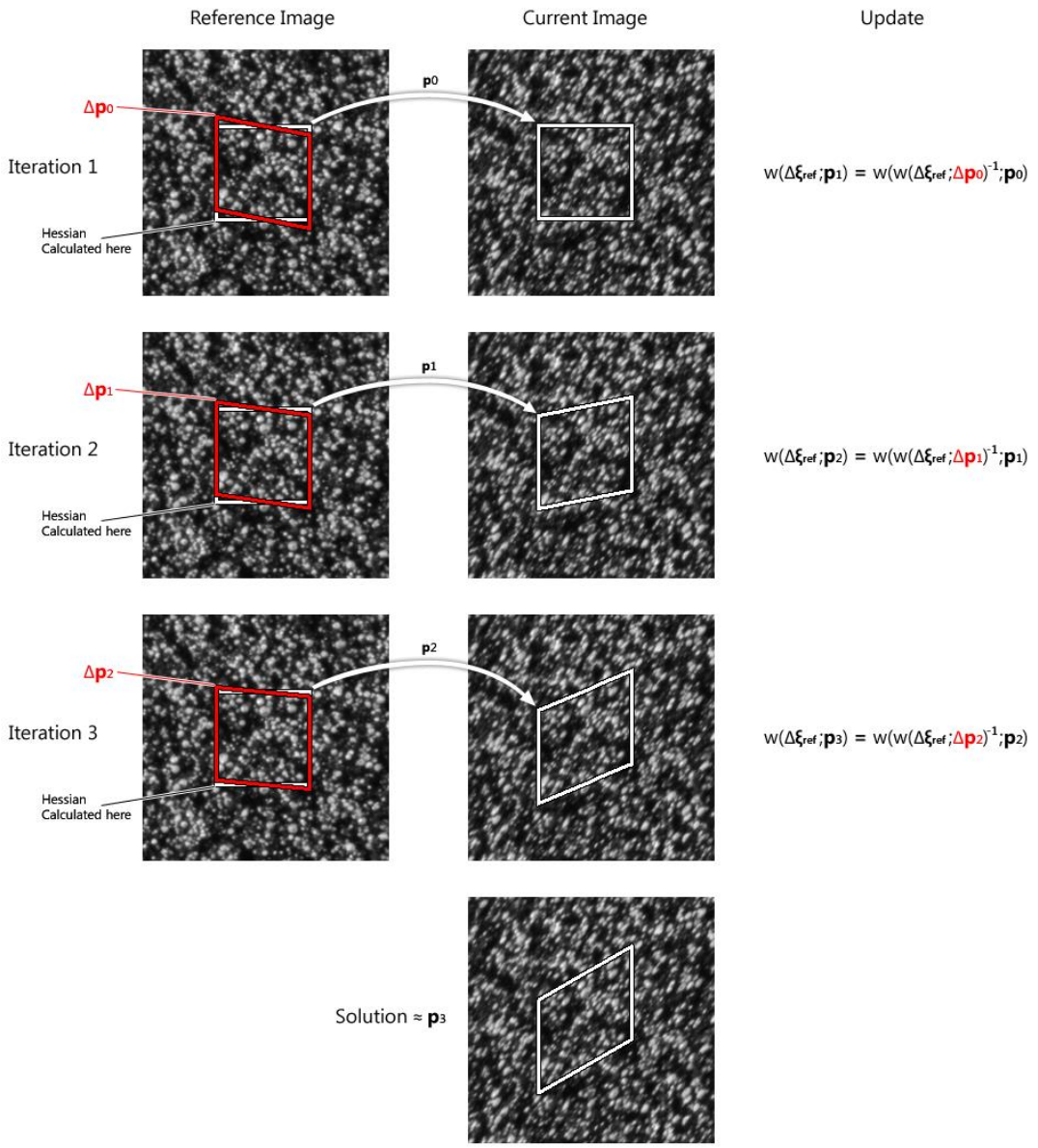


Figura 11. Esta es una representación gráfica del método IC-GN para la alineación de imagen.

La imagen de referencia tiene una cizalla aplicada a sí misma. Lo más importante a señalar aquí es donde se evalúa la hessiana en $\xi_{refc} + w(\Delta\xi_{ref}; p_{old})$ en la imagen de referencia para cada iteración. Debido a que esta cantidad se evalúa en la misma posición, sólo necesita ser calculada una vez. Se recomienda comparar esta figura con la figura 7 para tener una buena comparación visual entre los dos métodos iterativos.

4.3.5 Cálculo del gradiente y la hessiana para el método IC-GN

Ahora que se ha dado la descripción básica del método IC-GN, se procederá a explicar cómo calcular específicamente las cantidades del gradiente y la hessiana para el proceso de IC-GN.

La variable $\xi_{refc} + w(\Delta\xi_{ref}; 0)$ no es un problema, porque el subconjunto de referencia inicial es recogido de manera que $\Delta\xi_{ref}$ y ξ_{refc} se encuentran en las ubicaciones de píxeles enteros, por lo que la cantidad mencionada anteriormente se puede obtener directamente a partir de los valores de la escala de grises de referencia. La cantidad $\frac{d}{dip} f(\xi_{refc} + w(\Delta\xi_{ref}; 0))$; por otro lado, requiere un poco de trabajo. Necesita ser ampliado con la regla de la cadena con el fin de ser calculable.

Para mayor comodidad, se ha reescrito la ecuación 200 como se muestra a continuación:

$$\in S \begin{cases} \hat{x}_{refi} = x_{refi} + \Delta u + \Delta \frac{\partial u}{\partial x}(x_{refi} - x_{refc}) + \Delta \frac{\partial u}{\partial y}(y_{refi} - y_{refc}) \\ \hat{y}_{refj} = y_{refj} + \Delta v + \Delta \frac{\partial v}{\partial x}(x_{refi} - x_{refc}) + \Delta \frac{\partial v}{\partial y}(y_{refj} - y_{refc}) \end{cases} \quad (i, j) \quad \text{(Ecuación 225)}$$

Usando la regla de la cadena en $\frac{d}{dip} f(\xi_{refc} + w(\Delta\xi_{ref}; 0))$, obtenemos:

$$\begin{aligned} &\frac{d}{d\Delta p} f(\tilde{x}_{refi}, \tilde{y}_{refj}) \\ &= \frac{\partial}{\partial \tilde{x}_{refi}} f(\tilde{x}_{refi}, \tilde{y}_{refj}) * \frac{d\tilde{x}_{refi}}{dx} \\ &+ \frac{\partial}{\partial \tilde{y}_{refi}} f(\tilde{x}_{refi}, \tilde{y}_{refj}) * \frac{d\tilde{y}_{refi}}{d\Delta p} \end{aligned} \quad \text{(Ecuación 226)}$$

Para mayor comodidad, se han ampliado estas cantidades como se muestra a continuación:

$$\frac{\partial}{\partial \Delta u} f(\tilde{x}_{refi}, \tilde{y}_{refj}) = \frac{\partial}{\partial \tilde{x}_{refi}} f(\tilde{x}_{refi}, \tilde{y}_{refj}) \quad \text{(Ecuación 227)}$$

$$\frac{\partial}{\partial \Delta v} f(\tilde{x}_{refi}, \tilde{y}_{refj}) = \frac{\partial}{\partial \tilde{y}_{refj}} f(\tilde{x}_{refi}, \tilde{y}_{refj}) \quad \text{(Ecuación 228)}$$

$$\begin{aligned} &\frac{\partial}{\partial \left(\Delta \frac{\partial u}{\partial x}\right)} f(\tilde{x}_{refi}, \tilde{y}_{refj}) \\ &= \frac{\partial}{\partial \tilde{x}_{refi}} f(\tilde{x}_{refi}, \tilde{y}_{refj}) * (x_{refi} - x_{refc}) \end{aligned} \quad \text{(Ecuación 229)}$$

$$\frac{\partial}{\partial\left(\Delta\frac{\partial u}{\partial y}\right)}f\left(\tilde{x}_{refi},\tilde{y}_{refj}\right) = \frac{\partial}{\partial\tilde{x}_{refi}}f\left(\tilde{x}_{refi},\tilde{y}_{refj}\right)*\left(y_{refj}-y_{refc}\right) \quad \text{(Ecuación 230)}$$

$$\frac{\partial}{\partial\left(\Delta\frac{\partial v}{\partial x}\right)}f\left(\tilde{x}_{refi},\tilde{y}_{refj}\right) = \frac{\partial}{\partial\tilde{y}_{refj}}f\left(\tilde{x}_{refi},\tilde{y}_{refj}\right)*\left(x_{refi}-x_{refc}\right) \quad \text{(Ecuación 231)}$$

$$\frac{\partial}{\partial\left(\Delta\frac{\partial v}{\partial y}\right)}f\left(\tilde{x}_{refi},\tilde{y}_{refj}\right) = \frac{\partial}{\partial\tilde{y}_{refj}}f\left(\tilde{x}_{refi},\tilde{y}_{refj}\right)*\left(y_{refj}-y_{refc}\right) \quad \text{(Ecuación 232)}$$

Las únicas dos cantidades que necesitamos para calcular las ecuaciones 227 a 232 son $\frac{\partial}{\partial\tilde{x}_{refi}}f\left(\tilde{x}_{refi},\tilde{y}_{refj}\right)$ y $\frac{\partial}{\partial\tilde{y}_{refj}}f\left(\tilde{x}_{refi},\tilde{y}_{refj}\right)$. Se pueden calcular de diversas maneras, pero según Ncorr, se utiliza biquintic interpolación B-spline, por lo que con el fin de mantenerlo en consonancia se ha decidido calcular estos valores como si fueran las derivadas parciales de B-splines biquintic, que será explicado en la sección de interpolación.

La última cantidad que tenemos que abordar es $g(\xi_{refc}+w(\Delta\xi_{ref};p_{old}))$, que requiere interpolación.

Una vez que $\frac{\partial}{\partial\tilde{x}_{refi}}f\left(\tilde{x}_{refi},\tilde{y}_{refj}\right)$ y $\frac{\partial}{\partial\tilde{y}_{refj}}f\left(\tilde{x}_{refi},\tilde{y}_{refj}\right)$ están calculados para la imagen de referencia entera y $g(\xi_{refc}+w(\Delta\xi_{ref};p_{old}))$ es calculable, luego las ecuaciones 219 y 222 se pueden calcular e iterar con la ecuación 207 para encontrar una aproximación más cercana a P_{rc}^{*}.

El paso final para el solucionador iterativo es explicar el proceso de interpolación, que se realiza en la siguiente sección.

4.3.6 Panorama teórico de Biquintic B-spline (línea polinómica suave básica)

El propósito de esta sección es discutir la interpolación biquintic con el fin de calcular $\frac{\partial}{\partial\tilde{x}_{refi}}f\left(\tilde{x}_{refi},\tilde{y}_{refj}\right)$, $\frac{\partial}{\partial\tilde{y}_{refj}}f\left(\tilde{x}_{refi},\tilde{y}_{refj}\right)$ y $g\left(\hat{x}_{curi},\hat{y}_{curj}\right)$.

La interpolación Biquintic se utiliza porque mitiga los errores DIC que resultan de la interpolación de señales con contenido de alta frecuencia. Pero, es computacionalmente

costoso de implementar, aunque se justifica su uso debido a que DIC se hace a menudo "off-line" (es decir, después de que los experimentos se completen). Esta sección se centrará principalmente en la aplicación de biquintic B-splines con un poco de panorama teórico.

La idea principal detrás de la interpolación B-spline es la aproximación de la superficie de la imagen en escala de grises con una combinación lineal de B-spline o base spline. Estas splines son esencialmente pequeños baches en 2D con soporte finito (es decir, sólo una pequeña parte de su rango de distribución es diferente de cero). Estas splines se escalan por medio de los coeficientes B-spline y luego una combinación lineal de estas splines escaladas forman una aproximación de la superficie. Una vez que esta aproximación es completada, los puntos pueden ser interpolados a través de convoluciones, lo que se reduce a series de productos de punto simples.

Hay que tener en cuenta que toda la teoría dada en las siguientes secciones es para el caso de 1D. Las B-splines tienen una propiedad muy agradable, ya que son separables; esto significa que las interpolaciones en 2D se pueden desglosar en series de interpolaciones de 1D que es computacionalmente más rápido que una sola interpolación 2D (esta idea se ampliará más tarde). De todos modos, la interpolación se puede expresar como:

$$g(x) = \sum_{k \in z} c(k)\beta^n(x-k) \qquad \text{(Ecuación 233)}$$

Donde $c(k)$ es el valor del coeficiente B-spline en el número entero k, β^n(x-k) es el valor del núcleo B-spline en x - k y g(x) es el valor de la señal interpolada en x. n es el orden del núcleo B-spline, le hemos puesto un valor de 5 (el núcleo quinto) y Z es el conjunto de los números enteros.

Los coeficientes B-spline no son equivalentes a los datos muestreados (a diferencia de otras formas de interpolación y por lo tanto deben ser resueltos directamente. La ecuación para el núcleo de B-spline es:

$$\beta^n(x) = \frac{1}{n!}\sum_{k=0}^{n+1}\left(\frac{n+1}{k}\right)(-1)^k\left(x-k+\frac{n+1}{2}\right)^n \qquad \text{(Ecuación 234)}$$

Cuando resolvemos la ecuación 234 para n = 5, produce:

$$\beta^5(x)$$
$$= \begin{cases} \frac{1}{120}x^5 + \frac{1}{8}x^4 + \frac{3}{4}x^3 + \frac{9}{4}x^2 + \frac{27}{8}x + \frac{81}{40} & -2 \geq x \geq -3 \\ -\frac{1}{24}x^5 - \frac{3}{8}x^4 - \frac{5}{4}x^3 - \frac{7}{4}x^2 - \frac{5}{8}x + \frac{17}{40} & -1 \geq x - 2 \\ \frac{1}{12}x^5 + \frac{1}{4}x^4 - \frac{1}{2}x^2 \frac{11}{20} & 0 \geq x \geq -1 \\ -\frac{1}{12}x^5 + \frac{1}{4}x^4 - \frac{1}{2}x^2 + \frac{11}{20} & 1 \geq x \geq 0 \\ \frac{1}{24}x^5 - \frac{3}{8}x^4 + \frac{5}{4}x^3 - \frac{7}{4}x^2 + \frac{5}{8}x + \frac{17}{40} & 2 \geq x \geq 1 \\ -\frac{1}{120}x^5 + \frac{1}{8}x^4 - \frac{3}{4}x^3 + \frac{9}{4}x^2 - \frac{27}{8}x + \frac{81}{40} & 3 \geq x \geq 2 \end{cases} \quad \text{(Ecuación 235)}$$

Ahora que tenemos la mayoría de la información necesaria para iniciar el proceso de interpolación. El primer paso es determinar los coeficientes B-spline. Se pueden determinar mediante el uso de la convolución. Usamos la ecuación 233 y tomando la DFT, encontramos:

$$F(g) = F(c) * F(\beta^n) \quad \text{(Ecuación 236)}$$

El objetivo es resolver para c, los coeficientes de B-spline. Esto puede hacerse dividiendo los coeficientes de Fourier del núcleo B-spline elemento a elemento con los coeficientes de Fourier de la señal como se muestra a continuación:

$$F(c) = \frac{F(\beta^n)}{F(g)} \quad \text{(Ecuación 237)}$$

Tomando la DFT inversa de la ecuación 237 podremos encontrar los coeficientes B-spline, aunque se debe tener precaución al utilizar este método debido a la naturaleza circular de la DFT. Para mitigar errores wrap-around (envolventes), se debe utilizar el relleno.

Después de obtener los coeficientes de B-spline, el array de imagen puede ser interpolado point-wise (punto racional, sabio, prudente) usando la ecuación 223. Esto se lleva a cabo mediante una serie de productos de punto con las columnas del array del coeficiente B-spline y el núcleo B-spline, y después realizar un sólo producto a través de la fila resultante de los valores interpolados (el orden de esta operación no importa).

El primer paso del proceso antes mencionado puede ser pensado como la interpolación de la red B-spline en 2D para obtener valores de coeficiente B-spline de 1D, el segundo paso es interpolar el valor en escala de grises a partir de estos valores de coeficiente B-spline en 1D.

4.3.7 Resumen del método de composición inverso

Debido a que la explicación del método de composición inversa es muy largo, resumimos el método en esta sección.

Cálculo para todo el Análisis DIC:

1. **(opcional)** precalcular $[QK] * [c]_{(x_f - 2x_f + 3y_f - 2y_f + 3)} * [QK]^t$ para la imagen actual completa.

2. Evaluar $\frac{\partial}{\partial \tilde{x}_{refi}} f(\tilde{x}_{refi}, \tilde{y}_{refj})$ y $\frac{\partial}{\partial \tilde{y}_{refj}} f(\tilde{x}_{refi}, \tilde{y}_{refj})$ para la imagen de referencia completa usando las ecuaciones 244 y 245.

Cálculo por subgrupo:

3. Calcular las "imágenes de descenso más abruptas", usando las ecuaciones 227-232.

4. Calcular el GN-Hessiana en la ecuación 222.

5. Establecer el p_{old} inicial a la estimación inicial de NCC o al dato de deformación vecino.

Cálculo por iteración por subgrupo:

6. Calcular el subconjunto actual final deformado $g\left(\xi_{refc} + w(\Delta\xi_{ref}; p_{old})\right)$ usando la ecuación 241.

7. Calcular el gradiente, $\nabla C_{LS}(0)$ utilizando la ecuación 219.

8. Calcular Δp usando la ecuación 207 con la descomposición de Cholesky.

9. Actualizar p_{old} usando la ecuación 223.

10. Salir cuando la norma de Δp sea pequeña.

4.3.8 Breve introducción al método de correlación por áreas

Debemos destacar que los métodos explicados anteriormente son muy complicados a la hora de implementarse con un programa como Matlab debido a su complejidad matemática, que excede a nuestros conocimientos y coste computacional; el cuál se traduce en un coste temporal a la hora de analizar la correspondencia entre imágenes. Por ello, nos decantamos por el uso del método por áreas.

Una buena explicación intuitiva de este método es imaginarse las imágenes como matrices cuadradas en la que cada celda representa una faceta. Cada faceta será comparada con sus facetas vecinas en función de sus niveles de brillo para hallar la faceta que más se corresponda con la imagen de referencia. Por ello una limitación de este método son los grandes desplazamientos de píxeles entre la imagen de referencia y la imagen actual.

La gran diferencia entre nuestro proyecto y el proyecto padre es la implementación de la matriz de búsqueda que analiza las facetas vecinas de la imagen actual y su posterior

correlación con la imagen de referencia. Mientras que en el proyecto padre, la matriz de búsqueda recorre la imagen analizando las facetas vecinas en cada iteración, nosotros implementamos una rejilla compuesta por los puntos de trama que vamos a analizar y realizamos su correlación en un espacio de cuatro píxeles en todas direcciones, esta idea se ampliará en los próximos apartados.

5. Procedimiento experimental

En este punto empieza la parte práctica de nuestro proyecto, en la que implementamos un sencillo programa basándonos en el lenguaje de programación Matlab puesto que es un requisito inicial para este proyecto de fin de carrera.

Realizamos este programa con carácter pedagógico de forma que pueda ser reutilizado por la universidad, por otro lado somos conscientes de la ingente cantidad de temario que se estudia en esa asignatura por lo que hemos ideado una sencilla práctica que se podría llevar a cabo en una sola sesión de laboratorio, para que los alumnos se familiaricen con la correlación digital de imágenes (anexo B).

5.1 Investigación inicial

Nuestros conocimientos iniciales de correlación de imágenes eran muy básicos, ya que en nuestro plan de estudios, ninguna asignatura trataba este tema, por lo empezamos a recopilar información desde cero. Primero repasamos todos los conceptos tratados en la asignatura de Tratamiento Digital de la Imagen, recopilados en esta memoria, así como todas las prácticas de laboratorio con Matlab, tanto de esta asignatura como en la de Sistemas de Televisión, antes de empezar con la investigación sobre correlación digital de imágenes.

Después de buscar en múltiples páginas web y libros de la biblioteca del campus sur de la UPM, encontramos una página web que utiliza Matlab y la correlación de imágenes de manera profesional y de código abierto (Blaber, 2015) la cual nos ha servido para comprender los conceptos teóricos de la correlación digital de imágenes (DIC). De la gran cantidad de información que se encuentra en esa página, hemos seleccionado y traducido la parte que nos es de interés, pero el estudio de sus scripts en Matlab era demasiado complicado y no nos encajaba en el objetivo de éste proyecto, que tiene unos fines pedagógicos más que profesionales, por lo que nuestra búsqueda continuó. Nuestro tutor nos facilitó también gran cantidad de información que tenía recopilada en su ordenador, donde encontramos un tutorial de Matlab, (Chris Eberl, 2010) enfocado a la correlación digital de imágenes, que a pesar de su complejidad, nos ayudó a comenzar la programación por nosotros mismos sirviéndonos como base del proyecto.

Una vez tuvimos bastante información sobre la correlación de imágenes, una idea más clara de lo que iba a ser nuestro proyecto y teniendo ya elegido el método que íbamos a utilizar (se explica el porqué de esa elección en el siguiente punto), se nos ocurrió enfocar el proyecto a el análisis de la deformación del ábside de las iglesias con el paso del tiempo, ya que nuestro tutor José Manuel Díaz López, nos habló sobre ello en una de nuestras reuniones y así podíamos aprovechar un artilugio que él había creado junto con sus compañeros de departamento, para simular la desviación comentada, junto con un soporte con rosca universal de cámaras de fotos para poder realizar las fotografías sin variar la perspectiva entre tomas. Por lo que una vez que tuvimos el script casi finalizado, fuimos al laboratorio de televisión de la facultad a realizar las fotografías para su posterior análisis (figura 12).

El script ha sufrido muchas modificaciones desde su versión inicial, debido a la corrección de errores, sugerencias del tutor y nuevas ideas que iban surgiendo según íbamos programando, como pueden ser cambios en la adquisición de las imágenes para

hacerlo de manera más cómoda para el usuario, presentación de los resultados en milímetros en vez de píxeles, creación de un gif animado con los vectores de movimiento, etc.

Uno de los mayores problemas que hemos encontrado es la forma de introducir las imágenes, ya que no queríamos tener que seleccionarlas una a una desde Matlab, por lo que decidimos que la mejor manera era introducirlas con un nombre específico y un formato de 3 caracteres, para poder analizar su *path* como una cadena de texto y al tener una numeración correlativa, poder saber el nombre de las imágenes entre la primera y la última, pero esto junto con que queríamos analizarlas en escala de grises para simplificar su tratamiento, nos complicaba la tarea de pasar las fotografías de una cámara de fotos digital cualquiera al ordenador, con el formato y nombre específico que requiere nuestro script, por lo que investigamos alguna manera de poder modificarlas todas a la vez. La solución la encontramos en una herramienta que lleva incorporada el conocido programa de tratamiento de imágenes Adobe Photoshop, el cual permite automatizar una tarea y aplicarlo a todas las imágenes contenidas en una carpeta, en nuestro caso, nos permitió rotar todas las imágenes, pasarlas a escala de grises, reducir su resolución y cambiar el nombre con una numeración correlativa, lo que nos permite simplificar mucho esta tarea, y pensando en un futuro laboratorio de la asignatura en la facultad, hacer posible que cada alumno tome sus propias fotografías y las analice en el momento, la forma de realizarlo se explica en el anexo A.

Figura 12. Adquisición de las fotografías en el laboratorio de televisión.

5.2 Elección del método

Con unas bases teóricas más asentadas, decidimos utilizar un método por áreas mediante la Correlación Cruzada Normalizada (C_{CC}) (Ecuación 201).

La correlación por áreas implica un bajo coste computacional, en comparación con otros métodos de correlación, como puede ser mínimos cuadrados, por lo que nos interesa

bastante a la hora de utilizar este proyecto como una futura práctica de laboratorio de la asignatura.

En cuanto a usar C_{CC} como método de correlación, es debido a que es adecuado para proporcionar aproximaciones iniciales en zonas de deformación baja, y es un método con una implementación propia en las toolboxes de Matlab, lo que nos resulta muy interesante una vez más para reducir los tiempos en el cálculo computacional.

5.3 Introducción a Matlab

Matlab (abreviatura de *MATrix LABoratory*, "laboratorio de matrices") es una herramienta de software matemático que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio (lenguaje M). Está disponible para las plataformas Unix, Windows, Mac OS X y GNU/Linux .

Entre sus prestaciones básicas se hallan: la manipulación de matrices, la representación de datos y funciones, la implementación de algoritmos, la creación de interfaces de usuario (GUI) y la comunicación con programas en otros lenguajes y con otros dispositivos hardware. El paquete Matlab dispone de dos herramientas adicionales que expanden sus prestaciones; a saber, Simulink (plataforma de simulación multidominio) y GUIDE (editor de interfaces de usuario - GUI). Además, se pueden ampliar las capacidades de Matlab con las *cajas de herramientas* (*toolboxes*); y las de Simulink con los *paquetes de bloques* (*blocksets*).

Fue creado por el matemático y programador de computadoras Cleve Moler en 1984, surgiendo la primera versión con la idea de emplear paquetes de subrutinas escritas en Fortran en los cursos de álgebra lineal y análisis numérico, sin necesidad de escribir programas en dicho lenguaje. El lenguaje de programación M fue creado en 1970 para proporcionar un sencillo acceso al software de matrices *LINPACK* y *EISPACK* sin tener que usar Fortran.

En 2004, se estimaba que Matlab era empleado por más de un millón de personas en ámbitos académicos y empresariales.

En nuestro caso particular, el uso de Matlab radica en las posibilidades tan variadas que ofrece con el tratamiento de imágenes, ya que las imágenes digitales al fin y al cabo, no dejan de ser un conjunto de matrices (una matriz de tres dimensiones en el caso de imágenes en color o una matriz con diferentes niveles de brillo en el caso de las imágenes en escala de grises).

Por otra parte, al tener nuestro proyecto fines pedagógicos, que Matlab sea un programa con un lenguaje interpretado y no compilado es idóneo para poder analizar línea a línea lo que va sucediendo en el script. Si fuese un programa compilado, por ejemplo un archivo con extensión .exe, los tiempos de análisis se reducirían significativamente, pero haciendo todos los cálculos de forma transparente al usuario, sin que éste pueda analizarlos y comprender su funcionamiento, lo que difiere del objetivo de nuestro proyecto.

5.4 Explicación script

A continuación vamos a explicar el funcionamiento general del script, desde el punto de vista del usuario.

Lo primero que aparece en pantalla cuando se ejecuta el script de Matlab es un menú (figura 13) donde se explica la forma que deben de tener las imágenes para poder ser analizadas, solo se soportan esos formatos ya que tienen una extensión de tres letras, lo cual es necesario para poder analizar las fotografías según la forma en la que está programado el script.

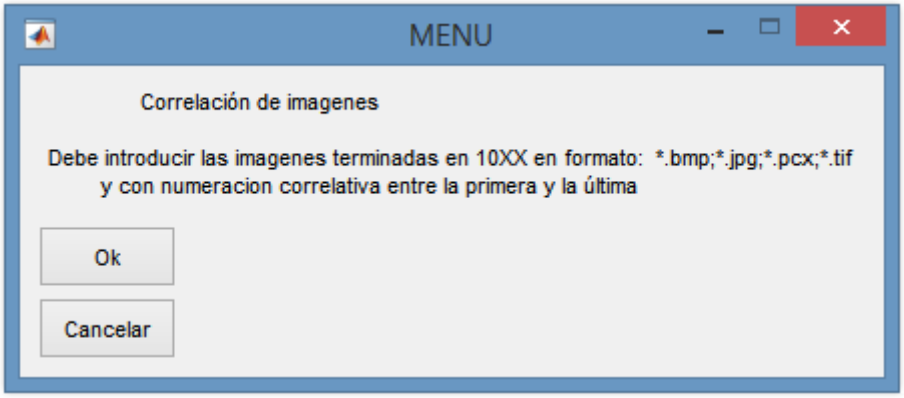


Figura 13. Primer menú del script donde se asegura el formato correcto.

A continuación se pide la primera y la última imagen a analizar, y se comprueba que el formato de las imágenes es el correcto, que se encuentran en escala de grises, y que hemos introducido un número de imágenes valido.

Posteriormente, analizamos el *path* de las imágenes, para crear una lista con todos los caminos que luego leeremos y guardaremos como variable propia de Matlab, lo hacemos de esta forma para no tener que pedir una a una todas las imágenes a analizar. Más tarde, se muestra otro menú (figura 14) para conocer la relación entre pixeles y milímetros, para al final poder mostrar los resultados de una manera que sea fácilmente compresible. Para ello hemos ideado dos formas: mirar manualmente la fotografía, y gracias a la regla milimetrada poder saber cuántos pixeles tiene un milímetro, o por otro lado, hacer *click* en los números de la regla y que el cálculo lo realice el propio programa. Hemos tenido en cuenta que se puede hacer fotografías en las que se vean diferentes longitudes de la regla, por lo que pedimos ese parámetro por pantalla.

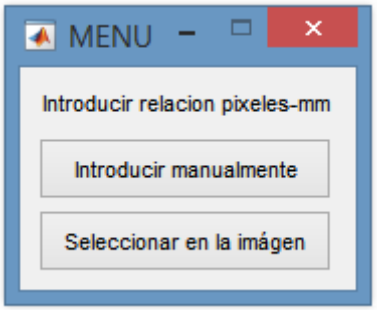


Figura 14. Relación píxeles milímetros manual.

Figura 15. Introduccion del tamaño de la regla por teclado.

Una vez introducida la relación pixeles-milímetros, se procede a determinar la región de interés de análisis de la imagen, haciendo *click* en los vértices superior izquierdo e inferior derecho de la imagen (figura 16) e introduciendo el espaciado entre los puntos de trama en píxeles. Los puntos de trama son todos aquellos puntos en los que vamos a analizar su desplazamiento posteriormente. Nos referimos a rejilla como el conjunto de puntos de trama en horizontal y vertical.

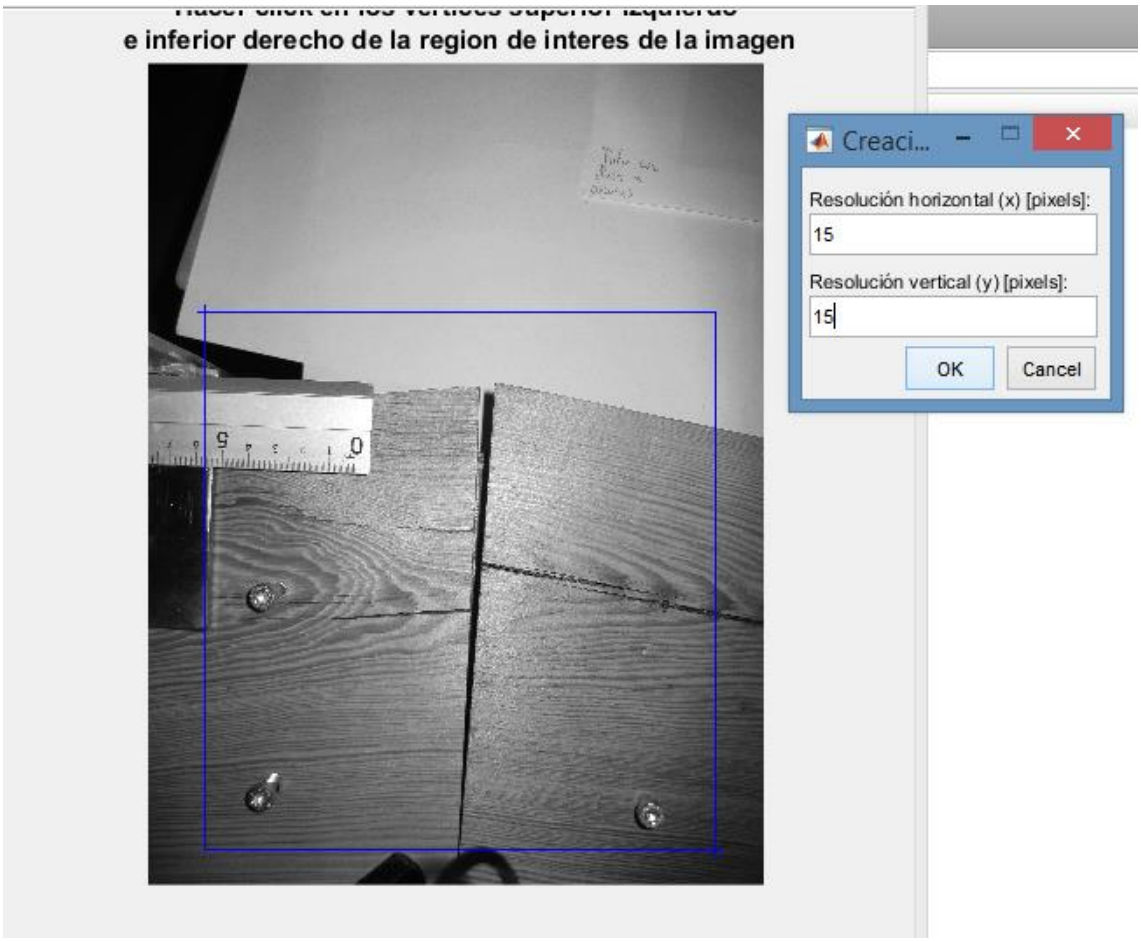


Figura 16. Selección de la región de interés.

Si la rejilla que hemos introducido no es la que queremos, podemos volver a introducirla, se pueden realizar diferentes pruebas con mayor o menor puntos de trama que representamos con cruces verdes (figura 17), que son los puntos a correlar.

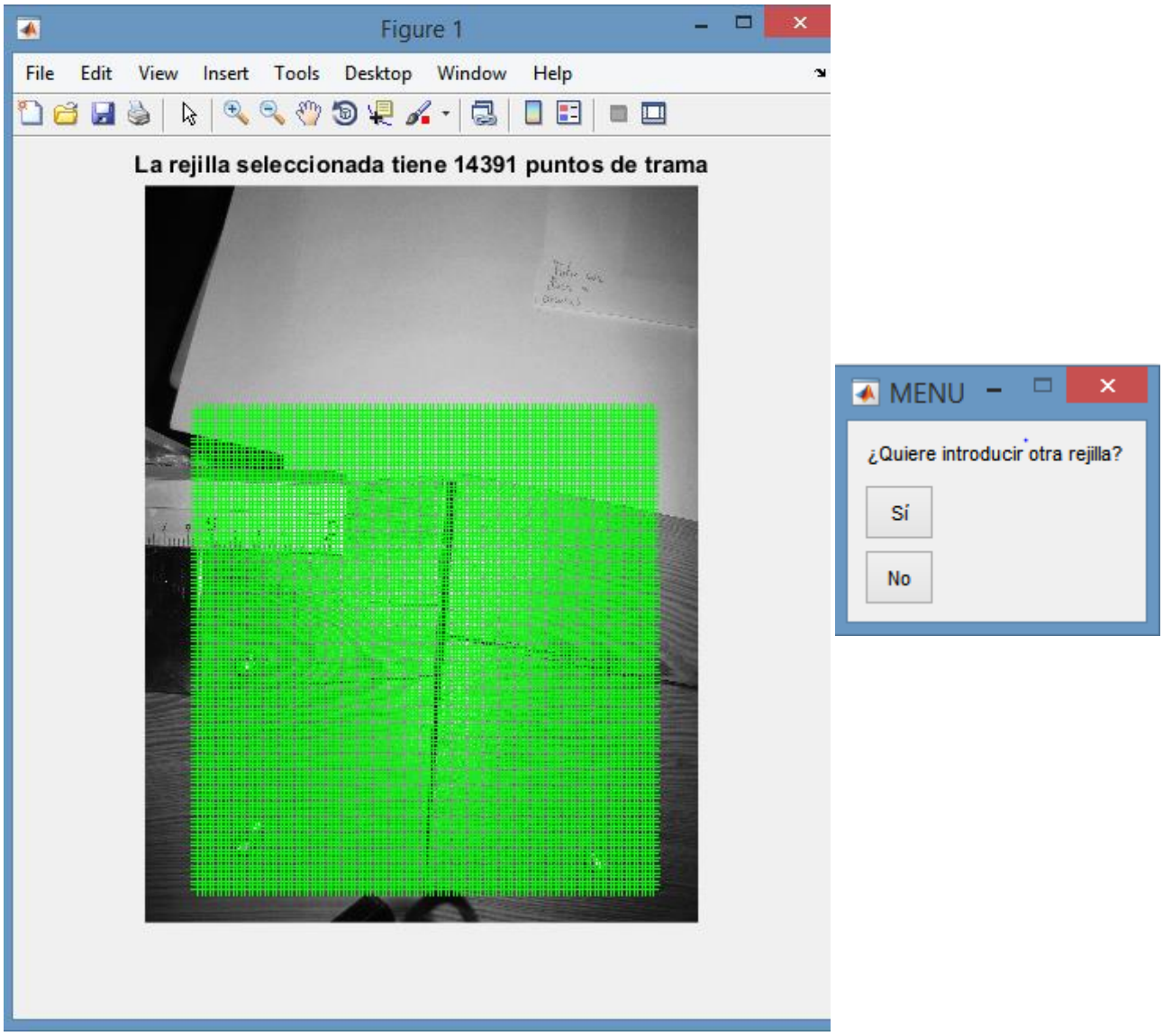


Figura 17. Confirmación de la rejilla.

Una vez introducida la rejilla deseada, se muestra un menú por pantalla con todas las opciones que tenemos de correlación (figura 18), las dos primeras opciones son una correlación simple entre dos imágenes, pasando a la función cpcorr, que es una función que se incluye en *"Image processing toolbox"* de Matlab ambas imágenes y una matriz con la posición de los puntos de trama a analizar (en realidad se le pasa un vector que se obtiene al reformar la matriz con todos los puntos, para más información mire el script adjunto) y devuelve la posición de los puntos de trama después de la correlación.

Figura 18. Opciones para la correlación.

La desviación de los puntos no puede ser muy grande, ya que de ser así, la función devuelve la posición inicial ya que solo modifica la posición de los puntos de trama en un máximo de 4 píxeles.

A continuación se muestra la ayuda de Matlab en español sobre esta función (Fiter, 2012).

Cpcorr

Afinado de la localización de los puntos de control mediante correlación cruzada.

Sintaxis

```
input_points = cpcorr(input_points_in, base_points_in, input,base)
```

Descripción

`input_points = cpcorr(input_points_in, base_points_in, input, base)` usa una correlación cruzada normalizada para ajustar cada par de puntos de control de las imágenes de entrada y referencia `input` y `base` especificados en las variables `input_points_in` y `base_points_in`.
`input_points_in` debe ser una matriz `M`-por-2 de clase `double` que contiene las coordenadas de los puntos de control de la imagen de entrada especificada en la variable `input`. `base_points_in` es una matriz `M`-por-2 de clase `double` que contiene las coordenadas de los puntos de control de la imagen de entrada especificada en la variable `base`.
`cpcorr` devuelve los puntos de control ajustados en la variable `input_points`, una matriz de tipo `double` del mismo tamaño que `input_points_in`. Si `cpcorr` no puede hacer la correlación de una pareja de puntos de control, `input_points` contendrá las mismas coordenadas que `input_points_in` para ese par.

`cpcorr` solo modifica la posición de los puntos de control en un máximo de 4 píxeles.
`cpcorr` es fiable en la selección y adquisición de los puntos de control pero las imágenes de entrada y referencia `input` y `base` deben tener la misma escala para que `cpcorr` sea efectiva.
`cpcorr` no puede ajustar un punto si ocurre cualquiera de los siguientes sucesos:
• Los puntos están demasiado cerca del borde de cada imagen.
• Hay regiones de la imagen alrededor de los puntos que contienen valores `Inf` o `NaN` (infinitos o indeterminados)
• La región alrededor de un punto en la imagen de entrada tiene desviación típica igual a cero.
• Hay regiones alrededor de los puntos que están correladas deficientemente.
Las imágenes de entrada y de referencia `input` y `base` pueden ser del tipo `numeric` y deben tener valores finitos. Los pares de puntos de control son de tipo `double`.

Haciendo click en cualquiera de las dos primeras opciones del menú, se pregunta al usuario que imagen quiere correlar y se analiza la imagen, una vez analizada, se pregunta al usuario si quiere mostrar los vectores de dirección en color, ya que al usar una función externa al matlab o sus toolbox, (Dano) el tiempo de análisis se incrementa notablemente. Posteriormente se muestran los resultados por pantalla (figuras 19, 20, 21, 22, 23, 24 y 25) y aparece de nuevo el menú con las opciones de análisis.

Si el análisis es correcto, y la desviación no es demasiado grande entre ambas imagen, los resultados serían parecidos a los siguientes.

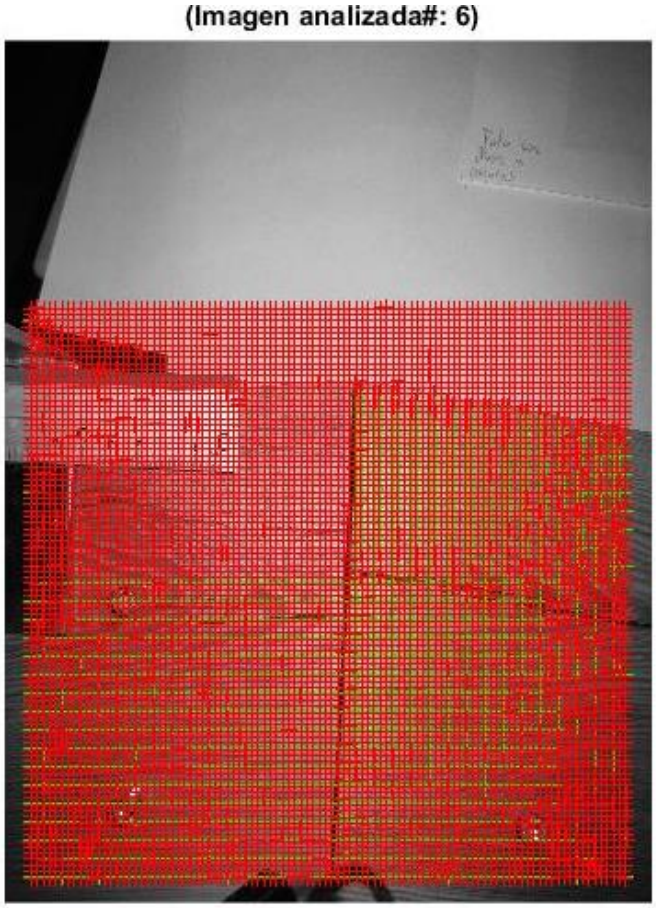


Figura 19. Imagen analizada, con las cruces antes y después de la correlación.

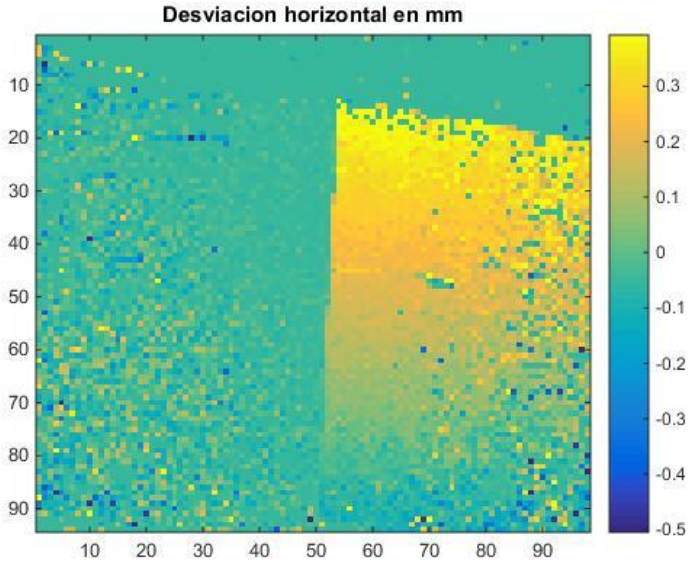


Figura 20. Desviación horizontal en milímetros.

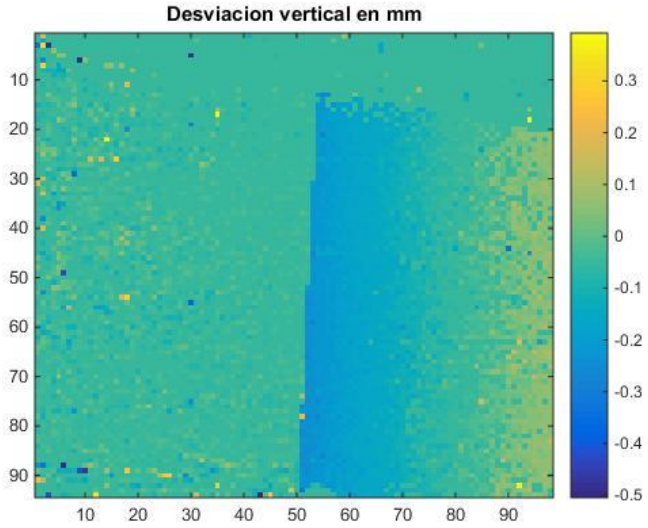


Figura 21. Desviación vertical en milímetros.

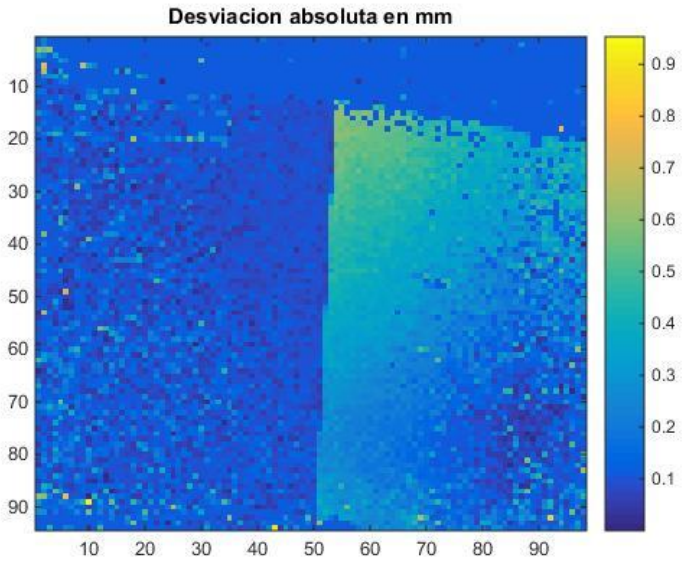


Figura 22. Desviación absoluta en milímetros.

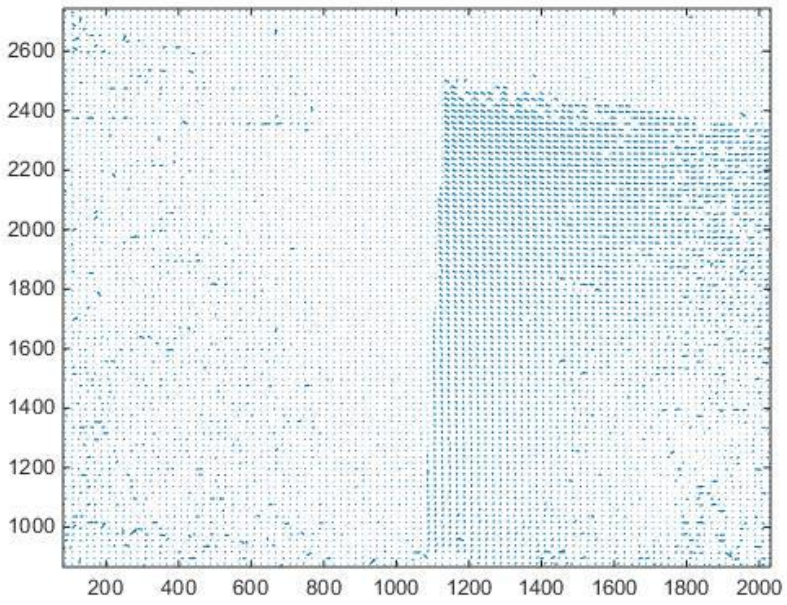


Figura 23. Vectores de movimiento de la imagen.

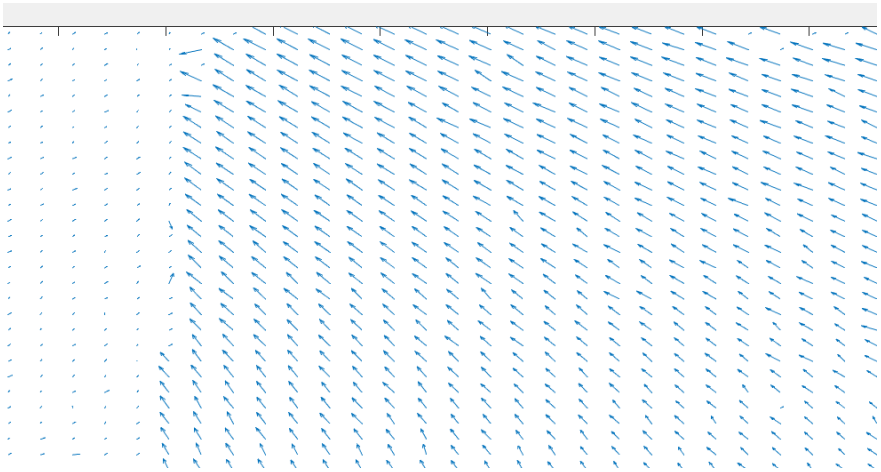


Figura 24. Detalle de los vectores de movimiento.

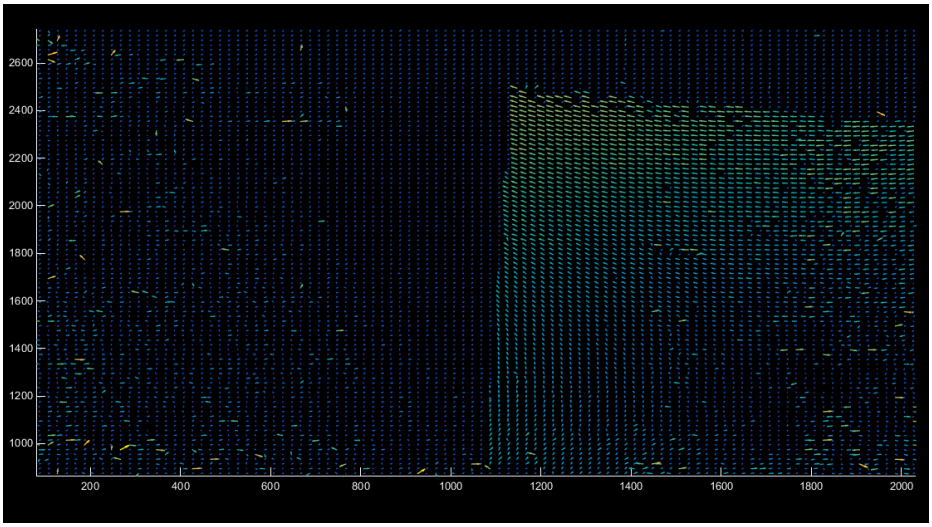


Figura 25. Vectores de movimiento en color.

Donde se puede ver que los resultados son los esperados, ya que en este caso aproximábamos las dos piezas de madera, y al tener el punto de rotación abajo, la desviación es mayor en la esquina superior izquierda, como se aprecia tanto en la gráfica de la desviación horizontal (figura 20) como en la de los vectores de movimiento (figura 23), apreciando como el modulo del vector es mayor en las zonas superiores (figura 24).

Sin embargo si analizamos desviaciones demasiado grandes, los resultados serían más aproximados a los siguientes (figura 26):

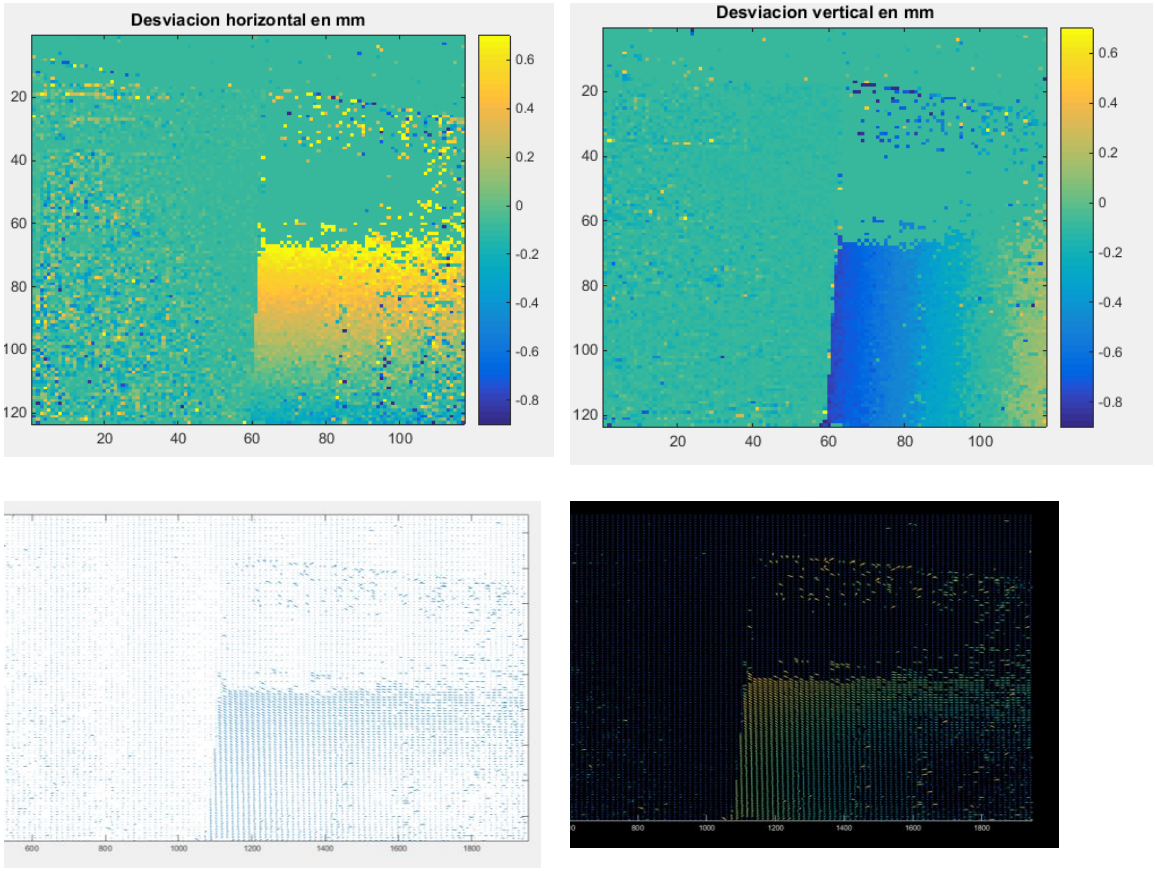


Figura 26. Resultados para un gran desplazamiento.

Donde se puede apreciar lo que hemos comentado anteriormente, que en las zonas donde ha habido una mayor desviación, los puntos no se han analizado y se muestran los puntos originales.

Las últimas opciones del menú (correlar todas con la anterior, y correlar todas con la primera imagen) realizan el mismo análisis, pero para todas las imágenes almacenadas, analizando cada una con su inmediatamente posterior en el primer caso, o con la primera imagen introducida en el segundo, pero mostrando únicamente la figura con los vectores de dirección en azul, guardando en una carpeta éstas figuras como jpg y creando un gif animado con la variación entre las imágenes.

5.5 Script de Matlab

```
% Pedimos las imagenes que nos van a servir para hacer la correlación,
las
% imagenes tienen que tener todas de la misma resolución
% todas las imágenes a analizar se tienen que encontrar en el mismo
% directorio y con el formato las imagenes terminadas en 10XX en
formato:
%*.bmp;*.jpg;*.pcx;*.tif
% donde XX son las imagenes que vamos a analizar, pudiendo analizar
desde
01 a 99
function PFC
close all
```

```
%Pedimos la primera imagen a procesar

pregunta = menu(sprintf('Correlación de imagenes\n \n Debe introducir
las imagenes terminadas en 10XX en formato:  *.bmp;*.jpg;*.pcx;*.tif
\n y con numeracion correlativa entre la primera y la última
'),'Ok','Cancelar');

if pregunta ==2
    error('Salir');
end
[fichero, path] = uigetfile(('*.bmp;*.jpg;*.pcx;*.tif'),
'Primera imagen a procesar');

ficheroconpath = strcat(path, fichero);
[a,F]=size (ficheroconpath);
%Comprobamos que los ficheros tienen el nombre correcto
if ficheroconpath (F-7)~= '1' || ficheroconpath (F-6)~= '0'
    error('Los números de la imágen tienen que terminar en 10XX');
end
%guardamos la ruta completa en esa variable
ima = imread(ficheroconpath);

[alto, ancho, componentes] = size(ima);
%Comprobamos que está en escala de grises
if componentes > 1
    error('La imagen no esta en escala de grises');
end
%Miramos la longitud del fichero para despues utilizarlo para leer las
%demás imágenes


[fichero2, path2] = uigetfile(('*.bmp;*.jpg;*.pcx;*.tif'),
'Última imagen a procesar');

ficheroconpath2 = strcat(path2, fichero2);
[a2,F2]=size (ficheroconpath2);

if ficheroconpath2 (F2-7)~= '1' || ficheroconpath2 (F2-6)~= '0'
    error('Los números de la imágen tienen que terminar en 10XX');
end
ima2 = imread(ficheroconpath2);

[alto2, ancho2, componentes2] = size(ima2);

if componentes2 > 1
    error('La imagen no esta en escala de grises');
end


%Control de errores
if F ~= F2
    error('La longitud de los nombres de fichero tiene que ser la
misma');
end
if (F2-F) >= 100 %Comprobamos que no haya mas de 100 imágenes
   error('No puede introducir más de 100 imágenes');
end
%Guardamos los numeros de las imágenes y lo pasamos a una variable
numérica
```

```
%para poder operar con ella
numchar= ficheroconpath((F-7):(F-4));
numchar2= ficheroconpath2((F2-7):(F2-4));
 num=str2num(numchar);
num2=str2num(numchar2);
numimagenes=(num2-num)+1;
%Creamos los numeros de todas las imagenes que vamos a analizar.
numseguidos=((num):(num2))';
%creamos la lista con las rutas de los ficheros que vamos a ir
completando
%con los números de uno en uno
for j=1:numimagenes%Copiamos el path hasta llegar a los numeros de la
imagen
listanombrefichero(j,:)=ficheroconpath (1,1:(F-8));
end

str=num2str(numseguidos);%Copiamos los numeros
listanombrefichero(:,(F-7):(F-4))=str(:,:);

for j=1:numimagenes%Copiamos la extensión y terminamos la lista.
listanombrefichero(j,(F-3):F)=ficheroconpath (1,(F-3):F);
end
[a,b,c]=size(listanombrefichero);
%Guardamos en una matriz de tantas dimensiones como imagenes vamos a
%analizar todas las imagenes.
wbar = waitbar(0, 'Trabajando, por favor espere...');
 for j=1:numimagenes

    imamatrix(:,:,j) =imread(listanombrefichero(j,:));

subplot((ceil(a/3)), 3, j);
imshow  (imamatrix(:,:,j),  [0  255]);  title(['',sprintf('  (Imagen#:
%1g)',j)]);
waitbar((j/(numimagenes)),wbar)
 end
close (wbar)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Vamos a pedir la relación entre pixeles y milímetros, para poder
%%%analizar posteriormente los resultados.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure;
imshow (imamatrix(:,:,1), [0 255]); title('Imagen Patrón');

menumm = menu(sprintf('Introducir relacion pixeles-mm'),
'Introducir manualmente','Seleccionar en la imágen');

if menumm==1
   %En el caso manual, el usuario debe de mirar en la fotografia
analizada
   %cuantos pixeles hay en un milimetro, y guardamos la relacion en la
   %variable varmm que utilizaremos despues para analizar los
resultados
   %graficamente
   respuestamm    =    inputdlg('¿Cuantos    pixeles    hay    en    un
mm?','Manualmente');

   pixelsmm = str2num(cell2mat(respuestamm));
   varmm= (1/pixelsmm);
end
```

```
if menumm==2
    %De igual manera que en el caso anterior, guardamos la relacion en
la
    %variable varmm pero haciendo los calculos con los parametros que
se
    %obtienen por pantalla
    cmrespuesta = inputdlg('¿Cuantos cm se ven de la regla?',
    'Seleccionar en la imágen');
    cmregla = str2num(cell2mat(cmrespuesta));
    %se tienen en cuenta diferentes longitudes de la regla
    title(['',sprintf('Haga click en el 0 y el %1g de la
regla',cmregla)]);

[xmm(1,1),ymm(1,1)]=ginput(1);% Entrada de parámetro con el cursor de
forma
%gráfica.
hold on
plot(xmm(1,1),ymm(1,1),'+b')

[xmm(2,1),ymm(2,1)]=ginput(1);
hold on
plot(xmm(2,1),ymm(2,1),'+b')
xminmm = min(xmm);
xmaxmm = max(xmm);
totalpixels= (xmaxmm-xminmm);

        %Ponemos 10*cmregla(mm) porque hacemos el calculo en mm no en
cm
        varmm=((10*cmregla)/totalpixels);

end


%%ELEGIR POR ENTRADA CON PUNTERO LA REJILLA CUADRADA DE LA REGIÓN DE
INTERES%%

compro=1;
while compro==1
    clearvars x y % Necesitamos limpiar las variables x e y ya que al
    %utilizar min y max de x e y, si cambiamos la rejilla coge los
    %valores de la anterior si no borramos esas variables.
    figure;
imshow (imamatrix(:,:,1), [0 255]); title('Imagen Patrón');
title(sprintf('Difinir región de interes. \n Hacer click en los
vértices superior izquierdo \n e inferior derecho de la region de
interes de la imagen'))

[x(1,1),y(1,1)]=ginput(1);% Entrada de parámetro con el cursor de
forma
%gráfica.
hold on
plot(x(1,1),y(1,1),'+b')

[x(2,1),y(2,1)]=ginput(1);
hold on
plot(x(2,1),y(2,1),'+b')

drawnow
```

```
xmin = min(x);
xmax = max(x);
ymin = min(y);
ymax = max(y);

lineabajo=[xmin ymin; xmax ymin];
linearriba=[xmin ymax; xmax ymax];
lineizq=[xmin ymin; xmin ymax];
linedrch=[xmax ymin; xmax ymax];
%Creamos las lineas para mostrar por pantalla la region que vamos a
usar
%para analizar posteriormente
plot(lineabajo(:,1),lineabajo(:,2),'-b')
plot(linearriba(:,1),linearriba(:,2),'-b')
plot(lineizq(:,1),lineizq(:,2),'-b')
plot(linedrch(:,1),linedrch(:,2),'-b')

% Pedimos la resolucion en pixeles de la separación de los puntos de
trama
textomenu2 = {'Resolución horizontal (x) [pixels]:', ...
        'Resolución vertical (y) [pixels]:'};
titulodlg2 = 'Creación de la rejilla';
numlineas2= 1;
def2     = {'10','10'};
respuesta2 = inputdlg(textomenu2,titulodlg2,numlineas2,def2);
xespaciado = str2double(cell2mat(respuesta2(1,1)));
yespaciado = str2double(cell2mat(respuesta2(2,1)));

% Redondeamos hacia arriba xmin,xmax and ymin,ymax y dividimos entre
el
% espaciado para sacar el numero de elementos (cruces) en cada eje
numXelem = ceil((xmax-xmin)/xespaciado)-1;
numYelem = ceil((ymax-ymin)/yespaciado)-1;

xmin_nuevo = (xmax+xmin)/2-((numXelem/2)*xespaciado);
xmax_nuevo = (xmax+xmin)/2+((numXelem/2)*xespaciado);
ymin_nuevo = (ymax+ymin)/2-((numYelem/2)*yespaciado);
ymax_nuevo = (ymax+ymin)/2+((numYelem/2)*yespaciado);

%Creamos la rejilla de analisis para mostrar por pantalla con cruces
verdes
[x,y]                                                                 =
meshgrid(xmin_nuevo:xespaciado:xmax_nuevo,ymin_nuevo:yespaciado:ymax_n
uevo);
%meshgrid nos reproduce un rejilla completa con las coordenadas de
rejilla que
%hemos introducido por pantalla
[filas,columnas] = size(x);

imshow (imamatrix(:,:,1), [0 255]); title('Imagen Patrón');
title(['La rejilla seleccionada tiene ',num2str(filas*columnas),
    ' puntos de trama'])
hold on;
plot(x,y,'+g')
hold off

compro = menu(sprintf('¿Quiere introducir otra rejilla?'),'Sí','No');
close all
```

```
end
xyreform(:,1)=reshape(x,[],1);          %reformamos la rejilla para que
la función cpcorr la acepte
    xyreform(:,2)=reshape(y,[],1);



menucorrelacion = menu(sprintf('Como quieres correlar'),
'Correlar con la anterior','Correlar con la primera imagen',
'Correlar todas con la anterior','Correlar todas con la primera
imagen','Finalizar');

%creamos un bucle para correlar tantas imágenes como se quiera y
mostrar
%los resultados posteriormente por pantalla


while menucorrelacion ~=5
    close all
    if menucorrelacion ==1 || menucorrelacion ==2
            numacorre = inputdlg('¿Qué imagén quieres correlar?');

            numacorre =str2double(cell2mat(numacorre));
            while (numacorre > numimagenes || numacorre==1)

                numacorre = inputdlg('¿Qué imagén quieres correlar
(introduce numero valido)?',
                '!!! ERROR Introduzca un número valido !!!');
                numacorre =str2double(cell2mat(numacorre));
            end
            if menucorrelacion ==1

           hbar = waitbar(0, 'Correlando, por favor espere...');
           waitbar(0.3,hbar)
            correlacioncpcorr(:,:)=cpcorr(round(xyreform),
round(xyreform),
            imamatrix(:,:,(numacorre)), imamatrix(:,:,numacorre-1));
            waitbar (0.8,hbar)
            close (hbar);
        end

        if menucorrelacion ==2
            gbar = waitbar(0, 'Correlando, por favor espere...');
           waitbar(0.3,gbar)

            correlacioncpcorr(:,:)=cpcorr(round(xyreform),
round(xyreform),
            imamatrix(:,:,(numacorre)), imamatrix(:,:,1));
            waitbar (0.8,gbar)
            close (gbar);
        end
        menucolor = menu(sprintf('¿Quieres mostar los vectores de
direccion por colores?\n Incrementa notablemente el tiempo de
ejecución'),'Sí','No');
            %Le damos todos los  marcadores y las imagenes a procesar
a la función
          %cpcorr, que es una función que se encuentra en image
processing toolbox
```

```
            correlacionenx=correlacioncpcorr(:,1);%    Guardamos    los
resultados de cpcorr en la dirección x
            correlacioneny=correlacioncpcorr(:,2);%    Guardamos    los
resultados de cpcorr en la dirección y
        figure;
        % subplot(1,3,1)
        imshow (imamatrix(:,:,2), [0 255]);
        title(['',sprintf(' (Imagen analizada#: %1g)',numacorre)]);

            hold on
            plot(x,y,'+g')% mostramos en verde los puntos de trama
originales

            plot(correlacionenx,correlacioneny,'+r')
            %mostramos en rojo los puntos después de la correlación
            hold off


            %Creamos una matriz con las mismas dimensiones que la
region de
            % interes con las coordenadas de las cruces después de
correlar
        cpcorregidox=reshape((correlacionenx),filas,columnas);
        cpcorregidoy=reshape((correlacioneny),filas,columnas);


%        Restamos los puntos originales con los puntos despues de la
%         correlacion para obtener la desviacion en pixeles que existe
para
%        cada punto
        difx = (x-cpcorregidox);
        dify = (y-cpcorregidoy);

%
         %%%%%%%Con la variable varmm que creamos anteriormente vamos
         %a mostrar en mm en vez de pixeles%%%

         xmmplot=(difx*varmm);
         ymmplot=(dify*varmm);
        absx= abs(xmmplot);
        absy= abs(ymmplot);
        abstotal = absx+absy;

         figure;
        %Mostramos por pantalla los resultados
        imagesc(xmmplot),colorbar
       title('Desviacion horizontal en mm')

        figure;

        imagesc (ymmplot),colorbar
        title('Desviacion vertical en mm')

        figure;

        imagesc(abstotal),colorbar
        title('Desviacion absoluta en mm')
        figure;
```

```
        %Mostramos los vectores de direccion

        xquiver= flipud(x);
        yquiver= flipud (y);
        quiver(xquiver,yquiver,(cpcorregidox-x),(cpcorregidoy-
y));axis([xmin xmax ymin ymax])

        if menucolor== 1
        %Mostramos los vectores de direccion con colores
        % Lo mostramos con la funcion descargada de
        %http://www.mathworks.com/matlabcentral/fileexchange/3225-
quiverc
        figure;
        quiverc(xquiver,yquiver,(cpcorregidox-x),(cpcorregidoy-
y));axis([xmin xmax ymin ymax])
        end
    end

    if menucorrelacion ==3 %Correlar todas con la anterior
        wbar3 = waitbar(0, 'Correlando, por favor espere...');
        %Correlamos ahora todas las imágenes de una en una, guardando
las
        %figuras de los vectores de direccion en la carpeta
video_todas_con_anterior
        %que creamos, tambien se crea un gif animado donde se puede
ver la
        %variacion entre las imágenes.
        mkdir('video_todas_con_anterior')
    cd('video_todas_con_anterior');
    Vid='Vid';
        for i=2:numimagenes
        correlacioncpcorr(:,:)=cpcorr(round(xyreform),
round(xyreform),
        imamatrix(:,:,(i)), imamatrix(:,:,i-1));
        waitbar((i/numimagenes),wbar3)
        correlacionenx=correlacioncpcorr(:,1);%      Guardamos      los
resultados de cpcorr en la dirección x
        correlacioneny=correlacioncpcorr(:,2);%      Guardamos      los
resultados de cpcorr en la dirección y
        cpcorregidox=reshape((correlacionenx),filas,columnas);
        cpcorregidoy=reshape((correlacioneny),filas,columnas);

    u=i+999;
        ustr=num2str(u);
        videoname=[Vid ustr '.jpg']
        h=figure;
        xquiver= flipud(x);
        yquiver= flipud (y);
        quiver(xquiver,yquiver,(cpcorregidox-x),(cpcorregidoy-y));
        axis([xmin xmax ymin ymax]);
        title(['',sprintf(' (Imagen#: %1g)',(i-1))]);

        saveas(h,videoname,'jpg')

        %crear                          un                          gif
https://waterprogramming.wordpress.com/2013/07/31/generating-gifs-
from-matlab-figures/
        drawnow;
    frame = getframe(1);
    im = frame2im(frame);
```

```
    [imind,cm] = rgb2ind(im,256);
    outfile = 'todas_con_la_anterior.gif';

    % En la primera iteraccion crea el gif y despues lo va
actualizando.
    if i==2
        imwrite(imind,cm,outfile,'gif','DelayTime',1,'loopcount',inf);
    else

imwrite(imind,cm,outfile,'gif','DelayTime',1,'writemode','append');
    end
     close (h)
        end
         close (wbar3)


    end

    if menucorrelacion ==4 %Correlar todas con la primera de manera
idéntica
        %a el caso anterior.
       wbar3 = waitbar(0, 'Correlando, por favor espere...');
        mkdir('video_todas_con_primera')
    cd('video_todas_con_primera');
    Vid='Vid';
        for i=2:numimagenes
        correlacioncpcorr(:,:)=cpcorr(round(xyreform),
round(xyreform),
        imamatrix(:,:,(i)), imamatrix(:,:,1));
        waitbar((i/numimagenes),wbar3)
        correlacionenx=correlacioncpcorr(:,1);
        % Guardamos los resultados de cpcorr en la dirección x
        correlacioneny=correlacioncpcorr(:,2);
        % Guardamos los resultados de cpcorr en la dirección y
        cpcorregidox=reshape((correlacionenx),filas,columnas);
        cpcorregidoy=reshape((correlacioneny),filas,columnas);

    u=i+999;
        ustr=num2str(u);
        videoname=[Vid ustr '.jpg']
        h=figure;
        xquiver= flipud(x);
        yquiver= flipud (y);
        quiver(xquiver,yquiver,(cpcorregidox-x),(cpcorregidoy-y));
        axis([xmin xmax ymin ymax]);
        title(['',sprintf(' (Imagen#: %1g)',(i-1))]);

        saveas(h,videoname,'jpg')


        %crear un gif
        drawnow;
    frame = getframe(1);
    im = frame2im(frame);
    [imind,cm] = rgb2ind(im,256);
    outfile = 'todas_con_la_primera.gif';

    % En la primera iteraccion crea el gif y despues lo va
actualizando.
```

```
    if i==2
        imwrite(imind,cm,outfile,'gif','DelayTime',1,'loopcount',inf);
    else

imwrite(imind,cm,outfile,'gif','DelayTime',1,'writemode','append');
    end
     close (h)
        end
         close (wbar3)


    end

         menucorrelacion    =    menu(sprintf('Como    quieres    seguir
correlando'),
         'Correlar con la anterior','Correlar con la primera imagen',
         'Correlar todas con la anterior',
         'Correlar todas con la primera imagen','Finalizar');
    end

end
```

6. Análisis de resultados y conclusiones

Como hemos visto, las desviaciones que obtenemos se ajustan a lo que esperábamos, ya que son del orden del milímetro, que es aproximadamente lo que movíamos la madera.

Hemos podido comprobar lo que en teoría ya sabíamos, que es que si la desviación es mayor, el análisis no se realiza, ya que usamos un método de correlación cruzada.

Los tiempos de análisis de las imágenes no son demasiado excesivos, tomando unos 20'' para un par de fotografías de resolución 1400x1050, con una separación del 10 píxeles entre los puntos de trama, dando un total de 7980 puntos de trama a analizar. Aumentando el tiempo hasta los 2'10'' si se muestran los vectores de dirección a color. Si se analizan las fotografías con su resolución original 2816x2112, el análisis aumenta a 1'15'' para un total de 31126 puntos de trama sin mostrar vectores de dirección a color y 7'40'' mostrándolos, de ahí el preguntar si se quieren mostrar o no.

Por lo tanto los tiempos son aceptables pero se podrían mejorar utilizando un programa compilado en vez de interpretado, pero el fin de este proyecto no es ese, si no hacer un script que se pueda usar de una manera más pedagógica en la universidad que con fines prácticos.

Para realizar estas medidas se ha utilizado un procesador AMD A10 5745M de 4 núcleos.

Como posibles mejoras a destacar, queremos proponer el poder ampliar el área de búsqueda de correlación, para desplazamientos mayores, ya que ese sería el mayor problema si se quiere dar una utilidad real al proyecto. Proponemos la implementación del método C_{LS} (Ecuación 202) ya que es capaz de analizar deformaciones más altas.

También sería interesante una manera de poder introducir las fotografías de una forma más cómoda, que no necesitase un tratamiento previo de nombre o conversión a escala de grises.

7. Bibliografía

Blaber, J. (2015). *Ncorr*. Recuperado el 12 de Abril de 2015, de www.ncorr.com

Cary. (2000). *"Deformation Measurements by Digital Image Correlation: Implementation of a Second-order Displacement Gradient."* .

Chris Eberl, R. T. (2010). *Digital Image Correlation and Tracking with Matlab.*

Dano, B. (s.f.). *Quiverc*. Recuperado el 2 de Abril de 2015, de http://www.mathworks.com/matlabcentral/fileexchange/3225-quiverc

Dante Dynamics. (s.f.). *Dante Dynamics*. Recuperado el 7 de Febrero de 2015, de http://www.dantecdynamics.com/measurement-principles-of-dic

DIAC. (2013). Laboratorio de Sistemas de Televisión. Madrid: EUITT.

DIAC. (s.f.). Apuntes asignatura Tratamiento Digital de la Imagen. *Departamento de Ingenieria Audivisual y Comunicaciones*.

E. López-Alba, F. A.-G. (2010). Análisis de deformaciones en probetas planas mediante correlación digital de imágenes. Jaén.

Fiter, E. L. (2012). *PFC: Descripción, comparación y ejemplos de uso de las funciones de la toolbox de procesadodigital de imágenes de MATLAB®.* Madrid: EUITT.

Mathworks. (s.f.). Recuperado el 9 de Marzo de 2015, de http://es.mathworks.com/

Mathworks. (s.f.). Ayuda de Matlab.

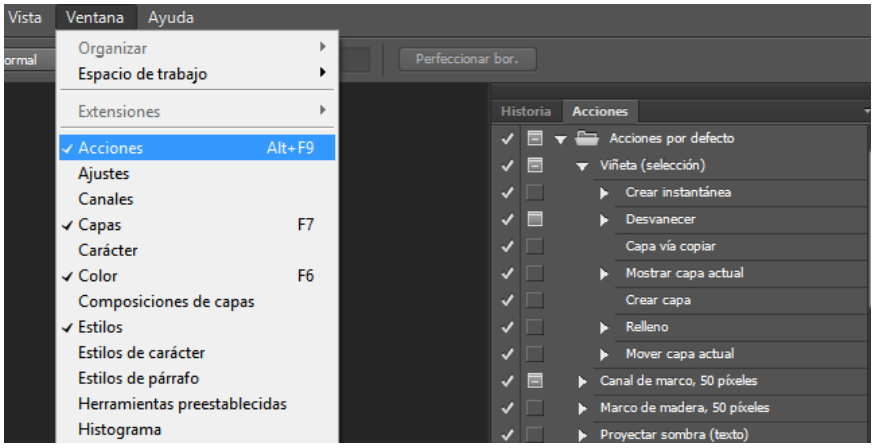
Pastor, E. M. (2013). *PFC: Análisis de imágenes basado en correlacion de imágenes.* Madrid: EUITT.

WaterProgramming. (s.f.). Recuperado el 23 de Marzo de 2015, de https://waterprogramming.wordpress.com/2013/07/31/generating-gifs-from-matlab-figures

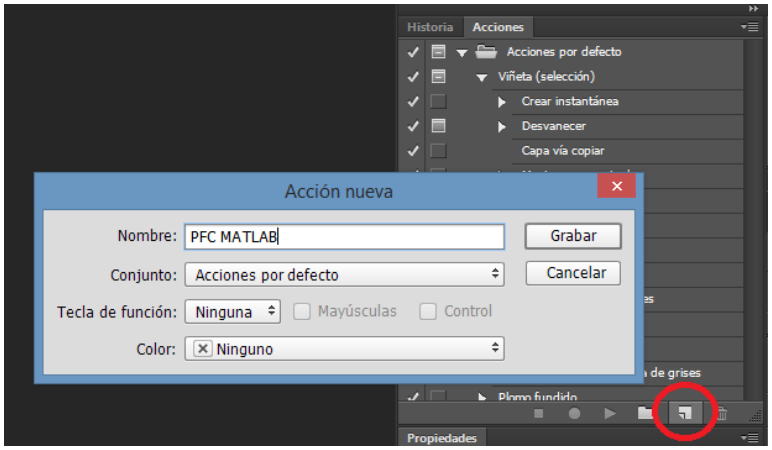
Wikipedia. (s.f.). *Wikipedia*. Recuperado el 3 de Mayo de 2015, de http://es.wikipedia.org/wiki/MATLAB

Anexo A. Tutorial práctico para aplicación en laboratorio: Pretratamiento de las imágenes a procesar con Adobe Photoshop mediante la automatización de lotes.

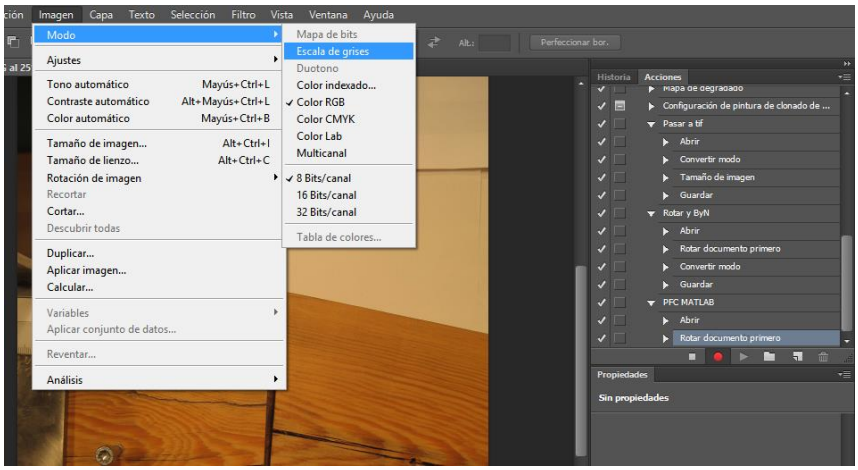
A continuación vamos a explicar el procedimiento a realizar para automatizar ciertas acciones por lotes, en nuestro caso rotación de la imagen, reducir resolución, pasar a escala de grises, y cambiar el nombre del archivo a uno con numeración correlativa y la extensión para que el script de Matlab lo acepte y sea una manera rápida de procesar muchas imágenes a la vez. Para la elaboración de éste tutorial se ha utilizado el software Adobe Photoshop CC versión de 2014.
Lo primero que debemos hacer es mostrar la pestaña de Acciones (Alt+F9)



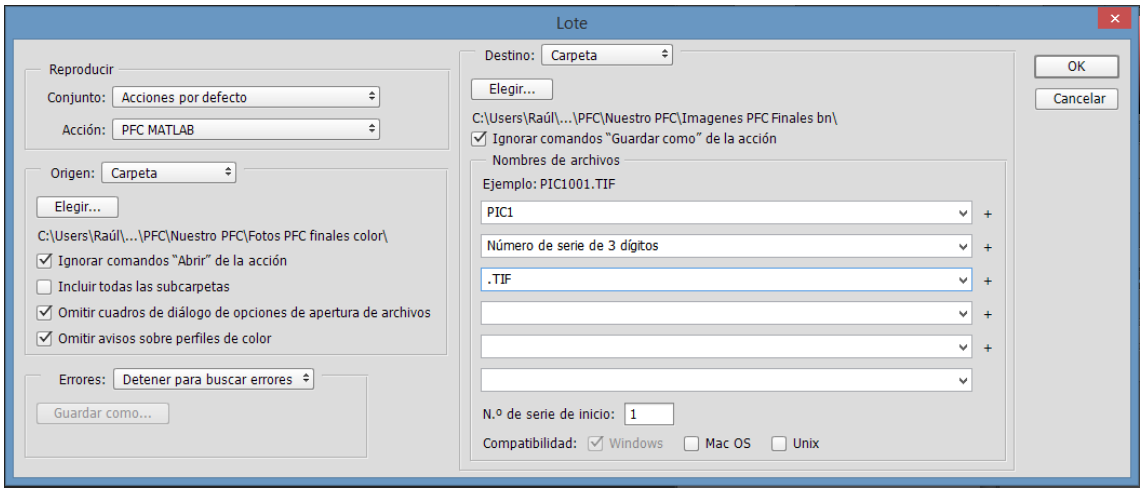
Clicaremos en crear nueva acción y pondremos un título y daremos a Grabar.



A continuación abriremos una fotografía haremos toda las transformaciones que consideremos oportunas, en nuestro caso rotar, cambiar de modo a escala de grises, ajustar la resolución (tamaño de imagen) y guardaremos. El origen y destino de la fotografía ahora es irrelevante.



Al finalizar con todos los tratamientos que queramos, haremos click al cuadrado a la izquierda del botón de *rec* redondo rojo.
Ahora vamos a aplicar todas estas transformaciones a todas las fotografías contenidas en una carpeta, cambiando el nombre y el formato. Para ello iremos al menú Archivo→Automatizar→Lote y nos aparecerá la siguiente ventana:



Donde seleccionamos las opciones siguientes: Ignorar comandos abrir de la acción, Omitir cuadros de dialogo, Omitir avisos sobre perfiles e Ignorar comandos "Guardar como" de la acción. Seleccionaremos las carpetas de Origen de las fotografías y el destino donde queremos que sean guardadas después de las transformaciones. En la parte Nombre de Archivo introducimos los siguientes parámetros para que el ejemplo tenga el formato PIC1001.TIF:

- PIC1 (o cualquier otro nombre acabado en 1 ya que es necesario para su tratamiento con Matlab).

- Número de serie de 3 dígitos (para la numeración correlativa).

- La extensión que queramos siempre que sea un punto y 3 letras como se indicaba en el script de Matlab, en nuestro caso hemos elegido .TIF.

Con todos esos parámetros seleccionados, daremos a OK y se empezaran a analizar todas las fotografías y se guardaran con el formato específico para que el script de Matlab lo acepte sin problemas.

Anexo B. Posible práctica de laboratorio.

1.- INTRODUCCIÓN

El objetivo de esta práctica es que el alumno consiga afianzar todas las ideas relativas a la correlación digital de imágenes, manejando los conceptos de región de interés, análisis mediante correlación cruzada normalizada y sus ventajas e inconvenientes.

2.- USO DEL SCRIPT

Los alumnos tomarán fotografías al elemento a analizar, intentando que entre ellas existan desviaciones muy pequeñas, deberán tomar las fotografías con una iluminación apropiada, haciendo uso de los fotos y mesa de mezcla situados en el laboratorio de televisión. Una vez tomadas las fotografías, deberán procesarlas creando una acción en el programa Photoshop como se indica en el anexo A.

Analizarán las fotografías con el script proporcionado haciendo uso de todas las opciones del menú de correlación, debiendo elegir una separación de los puntos de trama adecuada y contestando a las siguientes preguntas.

¿Al analizar una imagen con la anterior, los resultados son los que esperabas?

¿Qué sucede al analizar dos imágenes con una desviación muy alta, como por ejemplo al usar la opción correlar con la primera imagen?

3.- ANÁLISIS DEL SCRIPT

Los alumnos deben abrir el script de Matlab con el editor para comprender su funcionamiento línea a línea. Deben de separarlo en el mayor número de funciones posibles para manejar todas las variables posibles y las llamadas a esas variables entre funciones, reduciendo el script principal al menor número de líneas de código posible. Al menos se deberá dividir en las siguientes funciones:

- Función para pedir imágenes de la primera a la última y analizar su path.
- Función que lea la variable de la función anterior y guarde las imágenes como matriz multidimensional.
- Función para establecer relación entre ps y milímetros.
- Función para seleccionar región de interés por pantalla.
- Función con el menú de correlación.
- Función con el análisis de la correlación.
- Función para mostrar los resultados por pantalla.

- Función para crear un gif animado.

- Función para mostrar barras de espera por pantalla.

El alumno al finalizar la práctica debe conocer el funcionamiento de todas las funciones propias de Matlab contenidas en este script, haciendo uso del comando *help* propio de Matlab. Se confeccionará una lista con todas las funciones en orden alfabético que se usen a lo largo del script (con el nombre es suficiente).