UNIVERSIDAD MICHOACANA DE SAN NICOLÁS DE HIDALGO

Facultad de Ingeniería Eléctrica

División de Estudios de Posgrado

FUSIÓN DE IMÁGENES MULTI-FOCO

TESIS

Que para obtener el grado de

MAESTRO EN CIENCIAS EN INGENIERÍA ELÉCTRICA

Opción en Sistemas Computacionales

Presenta

Adan Garnica Carrillo

Dr. Félix Calderón Solorio

Director de Tesis

Morelia Michoacán Agosto 2015

FUSIÓN DE IMÁGENES MULTI-FOCO

Los Miembros del Jurado de Examen de Grado aprueban
la **Tesis de Maestría en Ciencias en Ingeniería Eléctrica** de ***Adán Garnica Carrillo***

Dr. Juan José Flores Romero
Presidente del Jurado

Dr. Félix Calderón Solorio
Director de Tesis

Dr. Leonardo Romero Muñoz
Vocal

Dr. José Antonio Camarena Ibarrola
Vocal

Dr. Sergio Bravo Solorio
Revisor Externo

Dr. Félix Calderón Solorio
Jefe de la División de Estudios de Posgrado
de la Facultad de Ingeniería Eléctrica. UMSNH
(Por reconocimiento de firmas).

UNIVERSIDAD MICHOACANA DE SAN NICOLÁS DE HIDALGO
Julio 2015

Quiero dedicar esta tesis a:

** A mi padre † Aristeo Garnica por haberme enseñado el valor del esfuerzo y la dedicación.*

Deseo expresar mi agradecimiento:

** A mi madre y hermanas por el apoyo que siempre me han dado.*

**De un modo muy especial a mi esposa e hijas que son quienes me apoyan e impulsan para no darme nunca por vencido.*

** A mi asesor Dr. Félix Calderón Solorio y a la institución por el tiempo y los recursos que pusieron a mi disposición para la realización de éste trabajo.*

** Los miembros de la mesa sinodal:*

- *Dr. Juan José Flores Romero, presidente del jurado.*
- *Dr. Leonardo Romero Muñoz, vocal*
- *Dr. José Antonio Camarena Ibarrola, vocal.*
- *Dr. Sergio Bravo Solorio, revisor externo.*

Por sus valiosas aportaciones a esta tesis.

Contenido

Lista de Figuras

Lista de Tablas

Lista de Símbolos

A	Matriz de restricciones en el proceso de optimización.
EOG	Energía de gradiente de una imagen.
E_L	Energía del Laplaciano de una imagen.
F_k	Respuesta de la k-ésima imagen al filtro de convolución H.
F_R	Frecuencia espacial por renglones.
F_C	Frecuencia espacial por columnas.
F_S	Frecuencia espacial de una imagen.
H	Kernel de convolución aplicado para medir la nitidez.
H_k	Kernels de convolución del Laplaciano discretizado.
I	Conjunto de imágenes $\{I_1, I_2, \cdots, I_{n-1}, I_n\}$.
I_F	Imagen fusionada.
I_{Fn}	Imagen resultante del n-ésimo proceso de fusión.
I_k	k-ésima imagen del conjunto de imágenes.
I_k^s	k-ésima imagen suavizada y submuestreada tomando 1 de s valores.
$I_m(x, y)$	Representa el pixel de la imagen I_m ubicado en la posición (x, y).
I_M	Imagen registrada resultado de alinear I_2 con I_1
I_0	Imagen original usada para la creación de imágenes sintéticas.
J_0	Versión emborronada de I_0, simulando el desenfoque.
L	Retícula de tamaño $N_r \times N_c$ en la que se representa la imagen.
LoG	Laplaciano de Gauss.
M	Número de restricciones en el proceso de optimización.
n	Número de imágenes en el conjunto I.
N_r	Número de renglones de la matriz que representa la imagen.
N_c	Número de columnas de la matriz que representa la imagen.
N	Número de variables a optimizar, $N = N_r \times N_c$
$N_v(x, y)$	Vecinos del pixel (x, y).
$N_w(x, y)$	Número de ventanas en las que aparece el pixel $p(x, y)$.
p	Matriz de pesos usada en la combinación lineal de las imágenes.
p_0	Matriz de pesos usada en la creación de imágenes multi-foco sintéticas.
p_k	Matriz de pesos resultado del k-ésimo proceso de fusión.
$p_{x,y}$	Matriz de pesos calculada para una ventana centrada en (x, y).

S	Entropía de una imagen.
u	Desplazamiento sufrido por la imagen en la dirección x
v	Desplazamiento sufrido por la imagen en la dirección y
w	Parámetro usado para establecer el tamaño de las ventanas
σ	Desviación estándar.
σ^2	Varianza.
μ_n	n-ésimo momento central absoluto.
μ_m	Media de los valores de los pixelse de la imagen I_m.
$\nabla^2 f$	Laplaciano de la función f.
ΔF	Diferencia entre F_1 y F_2.
$*$	Símbolo que representa la convolución.

Resumen

El problema de fusión de imágenes multi-foco consiste en dadas dos o más imágenes tomadas de la misma escena pero con distinto nivel de enfoque, extraer de dichas imágenes las regiones más nítidas, con el fin de formar una nueva imagen en la que aparezcan de forma nítida todos los objetos interesantes de la escena. El objetivo de la fusión de imágenes es obtener una nueva imagen con una calidad mayor que cualquiera de las imágenes fuente, la cual es más adecuada para la percepción humana o para el procesamiento digital. Donde el significado y la medición de la calidad dependerán de la aplicación.

En esta tesis se presenta una técnica para solucionar el problema de fusión de imágenes multi-foco, el cual es generado por la profundidad de campo limitada, que es una condición inherente de cualquier dispositivo que usa lentes ópticos. Se plantea solucionar el problema como un proceso de optimización, en el que se hace una combinación lineal de las imágenes multi-foco. Se establecen restricciones de coherencia espacial, que garantizan que la imagen fusionada además de tener la máxima nitidez, esté formada por regiones consistentes. El resultado del proceso es una imagen fusionada que parece haber sido capturada con una lente con profundidad de campo infinita. Se observó que la variación del enfoque genera una desalineación espacial entre las fotografías capturadas, lo que provoca que no haya una correspondencia exacta entre las coordenadas de un punto en una imagen y otra. Se presenta la implementación del método SSD-ARC, como técnica de registro de imágenes, con el fin de empatar las imágenes multi-foco para posteriormente realizar la segmentación y fusión de las mismas. El método propuesto ha demostrado tener un desempeño bastante aceptable al ser probado en un conjunto de imágenes que son comúnmente usadas para probar técnicas de fusión de imágenes. Se reportan resultados para imágenes sintéticas de tamaño 512×512, en las que se logró un porcentaje de acierto del 97.5 % en un tiempo de 4.5 minutos, mientras que en el estado del arte se reportan resultados, para las mismas imágenes, de 98 % de acierto pero en un tiempo de 94 minutos el cual es más de 20 veces mayor al tiempo consumido por nuestra propuesta. Aunque en las imágenes reales, no es posible medir cuantitativamente el desempeño del método, se presentan resultados en los que se puede observar de manera cualitativa, que el proceso de fusión se realiza de forma correcta.

Palabras clave: Fusión, Imágenes multi-foco, Registro de Imágenes, Ventanas deslizantes, Submuestreo.

Abstract

The multifocus image fusion is the process by which two or more partially focused images are combined into a new image in which all interested objects have the maximum sharpness. The image fusion process is applied when it is necessary to combine the source images into a new image with higher quality which cannot be achieved otherwise. The meaning and measurement of "quality"depend on the application. This thesis presents a technique to solve the multifocus image fusion problem, generated by the limited depth of field, which is a condition of any device with optical lenses. The problem is solved as an optimization process of a function that represents a linear combination of the multifocus images. A set of spatial coherence constraints is established to ensure that the resulting image has the maximum sharpness and it is formed by consistent regions. The result of the process is an image that appears to have been taken with a lens with an infinite depth of field. It was observed that the focusing variation generates a spatial misalignment between captured photos, which causes the coordinates of a point in the scene do not match in the multifocus images. The SSD-ARC method is presented as an image registration technique, in order to align the multifocus images before the fusion process. The proposed method has an acceptable performance when it is tested on a set of images that are used to test image fusion techniques. We report results for synthetic images of size 512×512, with an accuracy rate of 97.5 % in 4.5 minutes, while the state of the art are reported results with an accuracy rate of 98 % in 94 minutes for the same images, which is more than 20 times the time taken by our proposal. It is not possible to have a quantitative measurement of the performance of the method in the real images, however it is possible to observe that the resulting image is formed correctly.

Keywords: Image fusion, multifocus Images, Image Registration, Overlapping windows, subsampling.

Capítulo 1

Introducción

El problema de fusión de imágenes consiste en combinar la información de dos o más imágenes tomadas de la misma escena bajo distintas condiciones para crear una nueva imagen con mayor calidad que las imágenes fuente, donde la definición y la medición de la calidad dependen de la aplicación. Para el caso de la fusión de imágenes multi-foco, el objetivo es que la imagen fusionada sea más nítida que cualquiera de las imágenes multi-foco que se usan para la fusión. Malviya y Bhirud definen la fusión de imágenes multi-foco como el proceso de combinar dos o más imágenes parcialmente enfocadas o desenfocadas con el objetivo de crear una nueva imagen en la que aparezcan todos los objetos interesantes de manera nítida, [Malviya y Bhirud, 2009]. Goshtasby y Nikolov la definen como el proceso de combinar la información de dos o más imágenes de una escena en una imagen compuesta, la cual es más adecuada para la percepción humana o para el procesamiento digital, [Goshtasby y Nikolov, 2007].

Para resolver el problema de fusión de imágenes multi-foco debemos primero entender de manera clara, la representación de una imagen en forma digital. Una imagen digital es una representación abstracta de una escena y dado que no es posible almacenar toda la información que nos permita caracterizar totalmente la escena, se hace una discretización de la información. Las imágenes digitales I_m se representan generalmente en una retícula L, de tamaño $N_r \times N_c$ y en cada una de las posiciones de esa retícula se guarda por lo menos un valor, el cual nos proporciona información relacionada con una única posición específica

en la escena original. A cada una de las posiciones (x, y) de dicha retícula se le llama pixel, que puede ser representado como $I_m(x, y)$ con $0 \leq x < N_r$ y $0 \leq y < N_c$, siendo el pixel la unidad básica de una imagen digital. Cada pixel representa información concerniente al nivel de gris, color, temperatura, iluminación o alguna característica de un punto específico en la escena original, de acuerdo con el sensor utilizado para la captura de la imagen. Se puede representar una imagen digital como la retícula mostrada en la Figura 1.1.

$$
\overbrace{\left\{\begin{array}{|c|c|c|c|}
\hline
I_m(0,0) & I_m(0,1) & \cdots & I_m(0,N_c-1) \\ \hline
I_m(1,0) & I_m(1,1) & \cdots & I_m(1,N_c-1) \\ \hline
\vdots & \vdots & \ddots & \vdots \\ \hline
I_m(N_r-1,0) & I_m(N_r-1,1) & \cdots & I_m(N_r-1,N_c-1) \\ \hline
\end{array}\right.}^{N_c}
\Bigg\} N_r
$$

Figura 1.1: Representación de una imagen digital

1.1. Profundidad de campo

Una limitación de los dispositivos que usan lentes ópticos es la incapacidad de capturar los detalles de objetos que se encuentran a distancias diferentes de la lente. Nayar y Nakagawa en [Nayar y Nakagawa, 1994] y Potmesil y Chakravarty en [Potmesil y Chakravarty, 1981], muestran el modelo de una lente y explican que un objeto de la escena será reproducido con mayor o menor enfoque, en función de la distancia a la que se encuentra el objeto de la lente d_0, de la distancia focal f y de la distancia al plano de la imagen d_i. El nivel de enfoque también estará delimitado por la curvatura de la lente y por

la apertura del diafragma en el caso de las cámaras. En la Figura 1.2 se muestra el modelo de una lente, en donde se puede observar cómo ésta concentra la luz reflejada por un punto P de la escena, en una posición específica del sensor cuando se cumple la ecuación (1.1).

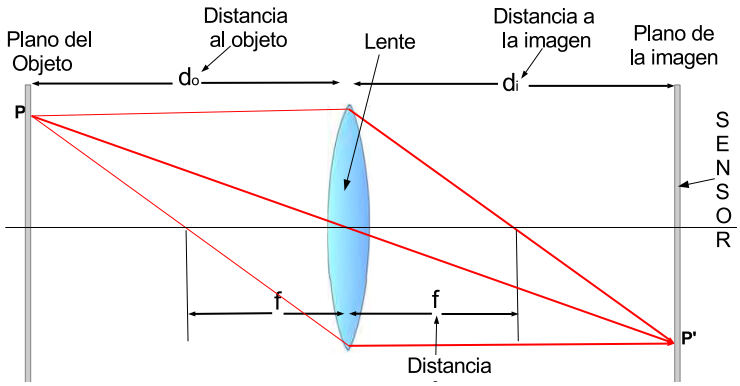


Figura 1.2: Modelo de una lente.

$$\frac{1}{d_o} + \frac{1}{d_i} = \frac{1}{f} \tag{1.1}$$

Si la ecuación (1.1) se cumple, un punto del objeto mapeará a un sólo punto en la imagen, como lo ilustra la Figura 1.3. En esta figura se puede ver que un punto A, en el objeto mapea a un punto A' en la imagen y un punto B en el objeto mapea a un punto B' en la imagen.

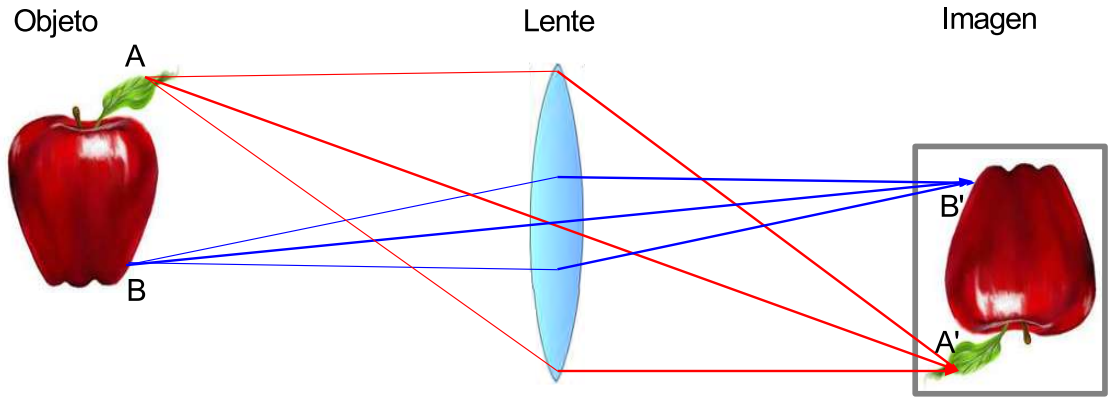


Figura 1.3: Correspondencia entre puntos en el objeto y en la imagen generada.

En la Figura 1.3, se muestra una situación ideal, en la que la luz reflejada por un punto en el objeto, es concentrada por la lente, sobre un sólo punto en el sensor de la cámara. Sin embargo esto no siempre es así, en la Figura 1.4 se muestra que si un punto está enfocado, la luz reflejada será concentrada por la lente en un punto igual en el sensor, pero si dicho punto está desenfocado (demasiado cerca o demasiado lejos), la luz incidirá sobre varias posiciones en el sensor, es decir, un sólo punto en la escena mapeará a varios puntos en la imagen generada. Dicha situación provoca que las imágenes se capturen con un efecto de escalamiento, además, dado que un sólo punto en la escena incide sobre varias posiciones en el sensor, se genera un círculo de confusión, que será de mayor o menor tamaño dependiendo de la distancia focal y de la distancia al objeto, [Sezan et al., 1991]. El hecho de que un punto en la escena incida sobre varias posiciones en el sensor, provoca que las regiones que están fuera de foco en la escena. sean capturadas por el sensor como una combinación de la luz reflejada por varios puntos en la escena original, lo que provoca regiones desenfocadas en las imágenes producidas por el sensor.

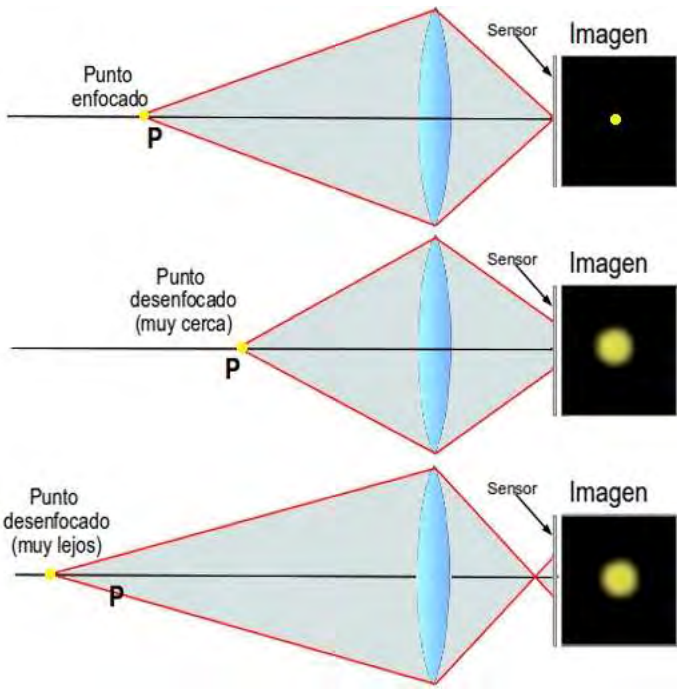


Figura 1.4: Círculo de confusión.

La profundidad de campo se define como el rango de profundidades que aparecen de manera enfocada en una imagen, [Kuthirummal et al., 2011]. Es decir el rango de distancias, medidas desde la lente, que es posible capturar con una nitidez aceptable, en una escena. En la Figura 1.5, se puede observar, que al modificar la distancia a la que se enfoca cuando se toma la fotografía, modificamos también el rango de distancias que aparecerán de manera nítida en la imagen.

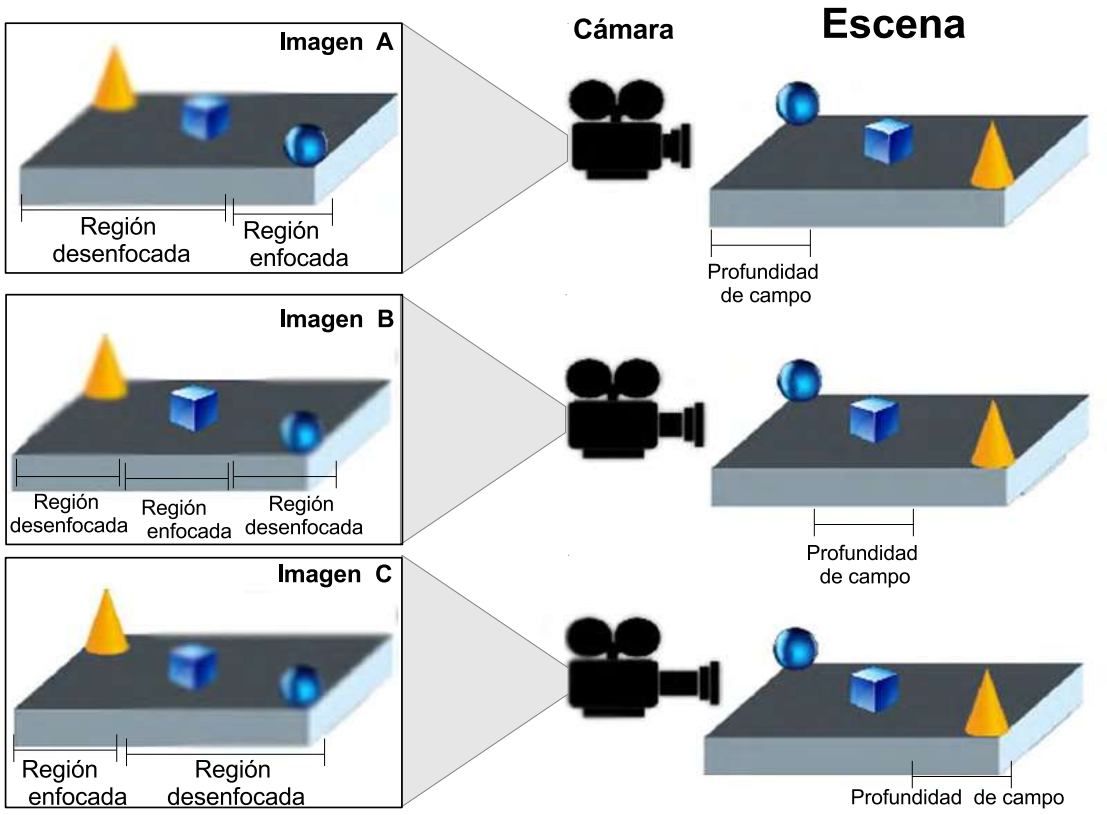


Figura 1.5: Profundidad de campo

El término nitidez hace alusión a algo que se distingue bien, que está bien definido o que es claro. Cuando una fotografía presenta una nitidez aceptable, decimos que está enfocada, por ello es común encontrar que en la literatura se usa casi indistintamente nítida, clara, bien definida o enfocada, para referirse a una imagen en la que se pueden apreciar los detalles fácilmente. Debido a que las condiciones físicas de las lentes no permiten tener profundidad de campo infinita, se requiere tomar la decisión de la distancia a la que se desea tener la

mayor resolución o nitidez. Dicha decisión nos lleva a tener imágenes con áreas que son nítidas y áreas que no lo son. En la Figura 1.6 se muestra una imagen capturada con una lente de poca profundidad de campo en la que se observa enfocado el plano de la flor en la escena y el fondo aparece borroso. Esto debido a que las condiciones físicas de la lente no permiten capturar de manera adecuada los detalles de objetos ubicados a una distancia mayor, por lo que perdemos los detalles del fondo.

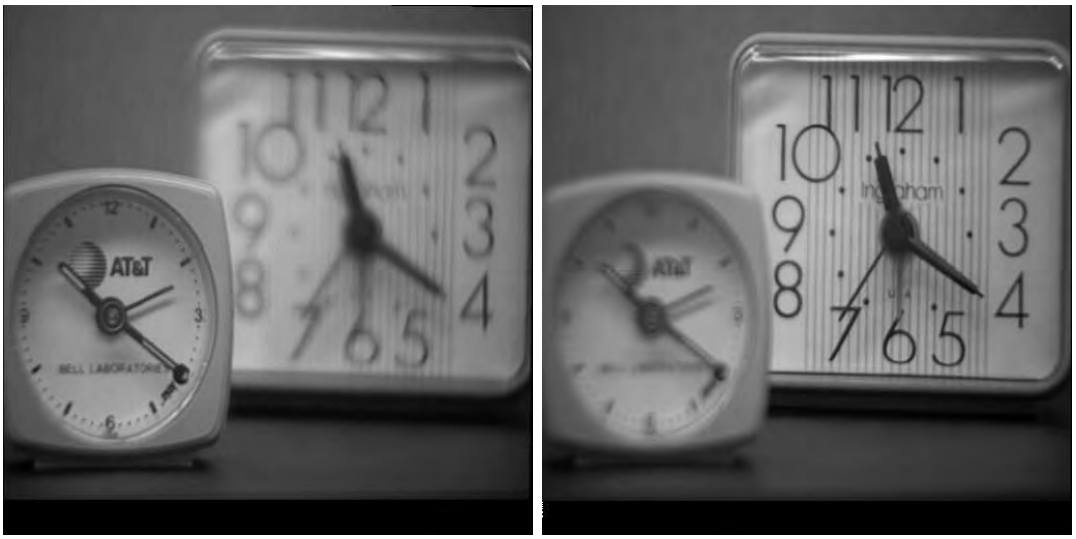
Figura 1.6: Fotografía capturada con una lente de poca profundidad de campo

Hasta donde hemos podido explorar del estado del arte, no existe una técnica probada que permita recuperar la información concerniente a los detalles de las regiones desenfocadas, basados en una sola imagen. Sin embargo una solución para lograr ver todos los objetos de interés de una escena, con una nitidez aceptable, es tomar varias fotografías, enfocadas a diferentes distancias, con el fin de capturar los detalles de cada uno de los objetos de la escena. Con esto tenemos un conjunto de imágenes I con el propósito de fusionarlas en una sola imagen que tenga mayor nitidez.

1.1.1. Imágenes multi-foco

Podemos definir las imágenes multi-foco como un conjunto de imágenes, correspondientes a la misma escena, pero que fueron capturadas bajo distintas condiciones de enfoque. Dicha situación la podemos ilustrar con un par de imágenes multi-foco, que son

muy conocidas en la literatura y que corresponden a dos relojes, las cuales se muestran en la Figura 1.7. En la Figura 1.7(a) se muestra una fotografía que fue tomada enfocando la cámara en el reloj que aparece enfrente en la escena, capturando los detalles de este reloj, pero sacrificando los detalles del fondo de la escena. Nótese que en consecuencia en esta fotografía se puede observar de manera más nítida la parte de enfrente de la escena y de manera poco nítida el fondo. En la Figura 1.7(b) se muestran las condiciones de enfoque de manera inversa a las de la Figura 1.7(a).



(a) ***Fotografía con enfoque en el reloj de enfrente*** **(b)** ***Fotografía con enfoque en el reloj de atrás***

Figura 1.7: Imágenes correspondientes a dos relojes capturadas con distinto nivel de enfoque.

Generalmente nos interesa tener una sola imagen en la que se pueda apreciar todos los objetos de interés de la escena con la mayor nitidez posible. Es por eso que se requiere extraer de cada una de las imágenes obtenidas aquellas regiones con mayor nitidez y combinar la información en una sola imagen donde simulemos que fue capturada con una lente con profundidad de campo infinita. Como ya se dijo, si se tiene una sola imagen con regiones desenfocadas, no se puede recuperar los detalles de dichas regiones pero si tenemos varias imágenes en las que las regiones desenfocadas en una imagen están enfocadas en otra podríamos extraer las regiones nítidas de cada imagen para crear una imagen final, en la

que toda el área aparezca enfocada o nítida.

1.2. Problema de fusión de imágenes multi-foco

Podemos decir que el problema de fusión de imágenes multi-foco consiste en, dado un conjunto de imágenes $I = \{I_1, I_2, \ldots, I_{n-1}, I_n\}$ donde $n \geq 2$, extraer de cada imagen I_k aquellas partes que son más nítidas en la imagen I_k que en cualquier otra imagen I_j del conjunto I con $k \neq j$ y unir las partes extraídas para crear una imagen fusionada I_F en la que toda el área tenga la mayor nitidez posible.

1.2.1. Ejemplo de fusión de imágenes multi-foco

Con el fin de ejemplificar mejor el objetivo de la fusión de imágenes multi-foco, planteamos un ejemplo realizando la fusión de las imágenes 1.7(a) y 1.7(b), de manera manual. El procedimiento realizado para fusionar las imágenes 1.7(a) y 1.7(b) es el siguiente:

1. Calcular una matriz de tamaño igual al de las imágenes, en la que se indique las regiones que son más nítidas en la imagen 1 y las que son más nítidas en la imagen 2, lo cual se muestra en la Figura 1.8.

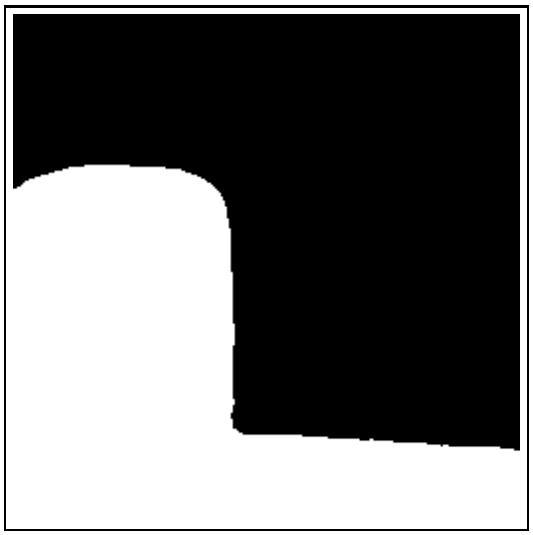
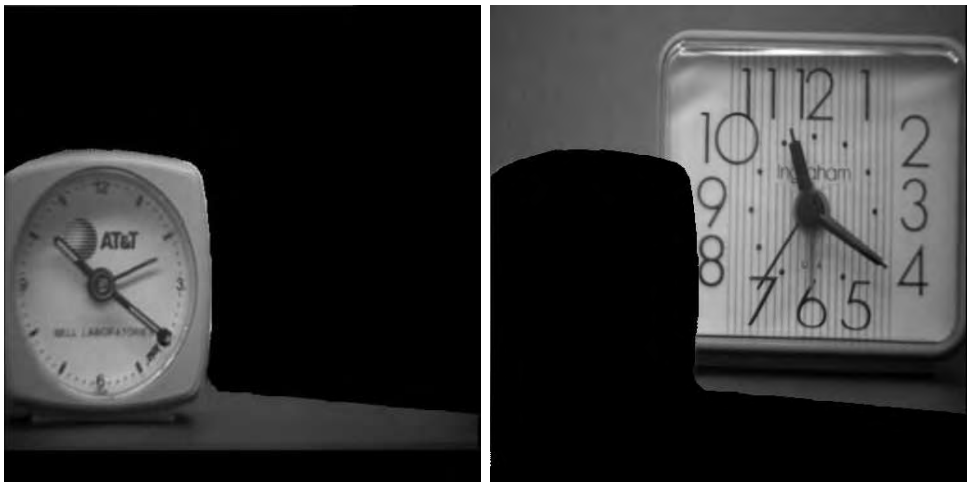


Figura 1.8: Regiones extraídas de cada imagen fuente

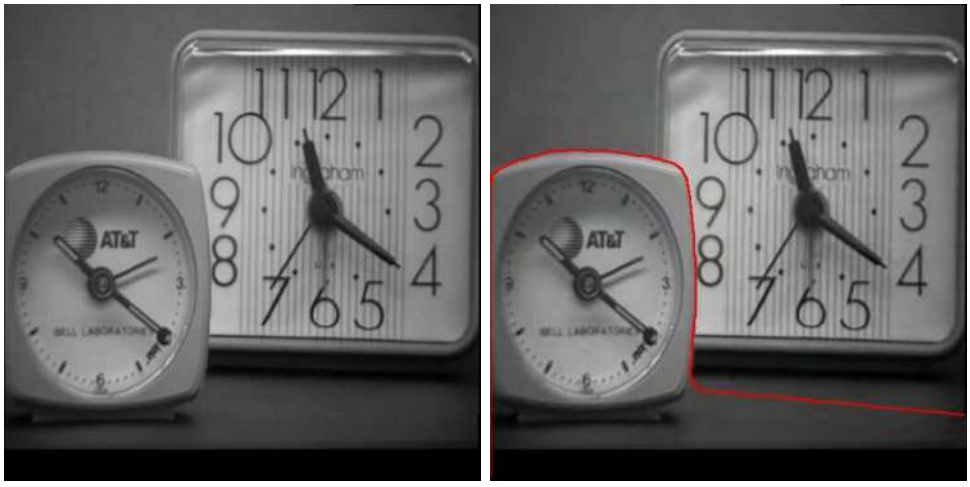
2. Basados en la información de la matriz calculada en el paso 1, extraer las regiones que aparecen más nítidas en 1.7(a) que en 1.7(b), como se muestra en la Figura 1.9(a). Así mismo las regiones que aparecen más nítidas en 1.7(b) que en 1.7(a), como se muestra en la Figura 1.9(b).



(a) ***Parte nítida de la imagen 1*** **(b)** ***Parte nítida de la imagen 2***

Figura 1.9: Proceso de fusión de imágenes multi-foco realizado de manera manual.

3. Fusionar la información extraída en una imagen final en la que se observe de manera nítida toda el área de la imagen, lo cual se muestra en las Figuras 1.10(a) y 1.10(b).



(a) ***Fusión de 1.9(a) y 1.9(b)*** **(b)** ***Segmentación realizada.***

Figura 1.10: Fusión de imágenes multi-foco realizado de manera manual.

Nótese en la Figura 1.10, que en la imagen final se muestra de manera nítida toda el área de la imagen. En la Figura 1.10(b), además de la fusión de las imágenes fuentes, se muestra la frontera de la segmentación con la que se hizo el corte de las imágenes multi-foco para lograr la imagen fusionada final.

El proceso de fusión requiere que las imágenes a fusionar estén perfectamente alineadas, lo cual en términos prácticos no es posible debido a movimientos en la cámara, cambios en la escena y a que el cambio de nivel de enfoque provoca efectos de escalamiento en los objetos de la escena.

1.3. Problema de registro de imágenes

Las técnicas planteadas en el estado del arte para resolver el problema de fusión de imágenes multi-foco, parten del supuesto de que existe una alineación perfecta entre las imágenes. Es decir, que un punto determinado de la escena, está representado siempre en las coordenadas (x, y), para todas las imágenes multi-foco. Sin embargo podemos mostrar un ejemplo de imágenes multi-foco en la que se puede observar que el mismo punto de la escena, no siempre está representado en las mismas coordenadas para todas las imágenes.

(a) ***Imagen 1*** **(b)** ***Imagen 2***

Figura 1.11: Problema de registro en imágenes multi-foco

En la Figura 1.11, el punto marcado con azul está ubicado en las coordenadas $(123, 324)$ en la imagen de la Figura 1.11(a) mientras que el mismo punto aparece en las

coordenadas $(119, 329)$ en la imagen de la Figura 1.11(b). En general podemos decir que un punto ubicado en (x, y) de la imagen 1 aparece en las coordenadas $(x+u, y+v)$ de la imagen 2. Para el caso del punto marcado con azul en la Figura 1.11 los valores para u y para v son: $u = -4$ y $v = 5$. Si analizamos el punto marcado con rojo en las imágenes 1.11(a) y 1.11(b) el cual está en las coordenadas $(69, 177)$ en la imagen 1 y en las coordenadas $(67, 180)$ en la imagen 2, tenemos que los valores de u y de v son $u = -2$ y $v = 3$, con lo que podemos ver que no todos los puntos de la escena sufren el mismo desplazamiento. Es por esa razón que si se desea comparar la nitidez de un punto en 2 imágenes multi-foco, es necesario alinear primero las imágenes o calcular los valores de u y de v, para todos los pixeles de la imagen a lo que se le conoce como el problema de registro de imágenes.

1.4. Motivación

La fusión de imágenes es muy necesaria cuando se requiere tener de manera nítida todos los objetos interesantes de una escena en una sola imagen, lo cual resulta imposible de lograr de otra manera cuando los objetos se encuentran a diferentes distancias de la lente y la profundidad de campo de la cámara no permite abarcar todas las distancias a las que se encuentran los objetos. Además de dicha aplicación la fusión de imágenes permite extraer de un conjunto de imágenes aquellas regiones que cumplen con determinados parámetros y unir las regiones extraídas en una imagen final en la que se puede analizar aquellas regiones que son de interés. Donde los parámetros varían dependiendo de la aplicación, por ejemplo para la fusión de imágenes multi-foco el parámetro a considerar es la nitidez.

1.5. Objetivos de la tesis

1.5.1. Objetivo general

Dado un conjunto de imágenes I tomadas con distinto nivel de enfoque, plantear un algoritmo que permita resolver el problema de fusión de imágenes multi-foco, para crear una imagen fusionada I_F que contenga de cada imagen I_k, aquellas regiones que son más nítidas en la imagen I_k que en cualquier otra imagen I_j del conjunto I. Simulando con esto

que la imagen I_F fue capturada con una lente con profundidad de campo infinito.

1.5.2. Objetivos particulares

- Detectar de manera automática las regiones nítidas de cada imagen.
- Garantizar la coherencia espacial en la imagen final
- Procesar imágenes de tamaños mayores o iguales a 512×512 en tiempos similares o mejores a los reportados en el estado del arte.

1.6. Propuesta de solución

En el contenido de esta tesis se presenta una técnica de solución al problema de fusión de imágenes multi-foco, extrayendo de manera automática las regiones más nítidas de cada imagen del conjunto I y uniendo las regiones extraídas en una imagen fusionada. Proponemos crear la imagen fusionada como una combinación lineal de las imágenes multi-foco, de tal manera que se maximiza la nitidez. Cabe hacer la aclaración de que el problema de registro de imágenes se resolverá utilizando el método SSD-ARC, que es un método de registro no paramétrico planteado en [Calderón y Marroquín, 2003]. Dicho método se aplicará con el fin de alinear los pixeles de las imágenes multi-foco antes de aplicar el proceso de fusión. La propuesta de solución al problema es la siguiente:

1. Aplicar una técnica de registro de imágenes con el propósito de que estén alineadas antes del proceso de fusión.
2. Calcular el nivel de nitidez de cada imagen multi-foco.
3. Calcular una matriz de pesos mediante un proceso de optimización, la cual se usará para combinar las imágenes maximizando la nitidez de la imagen fusionada.
4. Crear la imagen fusionada como una combinación lineal de las imágenes multi-foco.

1.7. Descripción de capítulos

El presente trabajo de tesis consta de 7 capítulos los cuales están distribuidos de la siguiente manera:

- En el capítulo 1 se hace una introducción al problema de fusión de imágenes multi-foco, se define el problema y se ofrece un ejemplo de cómo se resuelve de manera manual, esto con el fin de sentar las bases y coadyuvar a que el lector comprenda de la mejor manera los capítulos siguientes.

- En el capítulo 2 se hace una revisión de las distintas técnicas planteadas en la literatura para resolver el problema y de las diversas medidas de nivel de enfoque de una imagen que se han reportado. Así mismo se hace mención de los trabajos más prominentes reportados para la solución del problema de imágenes multi-foco.

- En el capítulo 3 se explica la necesidad de aplicar técnicas de registro de imágenes antes de resolver el problema de fusión de imágenes multi-foco y se plantea el método de SSD-ARC cómo técnica de registro no paramétrico que permite lograr una alineación espacial entre las imágenes multi-foco.

- En el capítulo 4 se explica el método que se usará para detectar las regiones nítidas en una imagen y se explica porque se ha decidido usar una versión discretizada del Laplaciano como detector de regiones de alto enfoque. Además se plantea la fusión de imágenes multi-foco como una combinación lineal de las imágenes fuente multiplicadas por un peso y se plantea una técnica para calcular los pesos que permiten hacer dicha combinación.

- En el capítulo 5 se plantea la propuesta para calcular los pesos requeridos para la ecuación lineal planteada en el capítulo 3, calculando dichos pesos mediante programación lineal y se explica el proceso a seguir para lograr los valores que permiten tener una imagen fusionada I_F con la nitidez óptima, garantizando la coherencia espacial de la imagen final.

- En el capítulo 6 se muestran los resultados obtenidos de aplicar el algoritmo planteado en el capítulo 5, en distintos pares de imágenes multi-foco.
- El capítulo 7 es dedicado para las conclusiones y para un análisis de las líneas a seguir en trabajos futuros.

1.8. Conclusiones del capítulo

La fusión de imágenes resulta de gran importancia cuando se tiene un conjunto de imágenes de las cuales se desea extraer sólo aquellas regiones que poseen determinadas características, como por ejemplo, las regiones más nítidas o las regiones con mayor iluminación y unir las regiones extraídas en una imagen final, la cual deberá contener todas las regiones que cumplen con las características establecidas. Hasta donde conocemos en el estado del arte las técnicas planteadas para resolver el problema de fusión de imágenes multi-foco resultan complejas y costosas en tiempo por lo que es necesario plantear una técnica que permita resolver el problema en de manera efectiva en el menor tiempo posible.

Capítulo 2

Antecedentes

En este capítulo se hace un análisis de las distintas técnicas presentadas en la literatura para medir el nivel de enfoque o nitidez de una imagen, se presenta el registro no paramétrico de imágenes y algunos de los trabajos más sobresalientes en ésta área.

2.1. Medidas de calidad de una imagen

Una técnica para resolver el problema de fusión de imágenes multi-foco es hacer un promedio entre los valores de cada pixel de las imágenes con las que se cuenta, sin embargo éste método tiene grandes deficiencias sobre todo cuando se aplica a imágenes que no están perfectamente registradas. Por lo que tenemos entonces que darnos a la tarea de decidir de qué imagen tomaremos la información que contendrá la imagen fusionada I_F. El objetivo es que la imagen I_F contenga las áreas de mayor calidad de las n imágenes del conjunto I. Entonces tenemos que encontrar la manera de medir la calidad o el nivel de enfoque de las imágenes. Pertuz et al., en [Pertuz et al., 2013], presentan una clasificación de las diferentes técnicas que hay para medir la calidad de una imagen. Dentro de las técnicas para medir el nivel de enfoque de una imagen podemos listar:

- **Medidas basadas en operadores de gradiente.** Estos operadores basan su análisis en primeras derivadas partiendo del supuesto de que las regiones que están enfocadas en la imagen, poseen bordes mejor definidos y en consecuencia las derivadas en las

áreas enfocadas tendrán una magnitud mayor que la derivada calculada en áreas que no están enfocadas, esto en función de que las derivadas son medidores de cambio y las regiones que están enfocadas son en la mayoría de los casos altamente cambiantes a diferencia de las áreas que no lo están, en donde el cambio entre el valor de un pixel y sus vecinos es generalmente poco. Zhou et al., [Zhou et al., 2014b] en plantean el uso de las primeras derivadas de la imagen en la dirección x y y para formar un tensor de estructura el cual es usado para calcular que tan sobresaliente es un pixel de una imagen. Esta medida es tomada como base para determinar las regiones enfocadas en cada una de las imágenes fuente. Así mismo se realiza el procedimiento a diferentes escalas con el fin de aproximar de mejor manera los límites de la segmentación.

Aunque existen diversos operadores de gradiente podemos mencionar la energía de gradiente como uno de los más usados y que mejor explica la idea del uso de primeras derivadas para medir el nivel de enfoque:

- **La energía de gradiente:** Es una medida de nitidez o nivel de enfoque, basada en primeras derivadas, que es definida como la suma de los cuadrados de las primeras derivadas direccionales [Sun et al., 2004, Huang y Jing, 2007, Chun y Kong, 2014]. Aunque la forma de calcular las derivadas puede ser planteada de distinta manera, este método plantea que el nivel de enfoque de una imagen se puede medir como:

$$EOG = \sum_{x=0}^{Nr-1} \sum_{y=0}^{Nc-1} D_x(x,y)^2 + D_y(x,y)^2 \tag{2.1}$$

  donde EOG es la energía del gradiente de la imagen, $D_x(x,y)$ es la derivada en la dirección x de la imagen en la posición (x,y) y $D_y(x,y)$ es la derivada de la imagen en la dirección y en la posición (x,y).

- **Medidas basadas en operadores Laplacianos:** La idea del uso de los operadores Laplacianos basa el trabajo en la premisa de que las segundas derivadas permiten identificar aquellas áreas de una imagen en las que la información es altamente variante por ejemplo bordes y que tienen en consecuencia poca suavidad. De acuerdo a lo planteado por Riaz et al., en [Riaz et al., 2008], pueden usarse operadores Laplacianos

como filtros pasa altas, lo que permite identificar las áreas en las que se encuentra la información de alta frecuencia. La idea del uso de los operadores Laplacianos como medida de nitidez es reforzada por lo planteado en algunos otros trabajos como [Sun et al., 2004, Huang y Jing, 2007, Chun y Kong, 2014], donde se presenta un análisis del uso de variantes del Laplaciano para medir el nivel de enfoque de una imagen. Aunque existen diversas variantes del Laplaciano así como de su implementación. La energía del Laplaciano consiste en calcular el cuadrado de la suma de las segundas derivadas en las direcciones x y y como lo mostrado en la ecuación (2.2).

$$E_L = \sum_{x=0}^{Nr-1} \sum_{y=0}^{Nc-1} (D_{xx}(x,y) + D_{yy}(x,y))^2 \tag{2.2}$$

donde E_L es la energía del Laplaciano de la imagen I_m, $D_{xx}(x,y)$ es la segunda derivada en la dirección x de la imagen en la posición (x,y) y $D_{yy}(x,y)$ es la segunda derivada de la imagen en la dirección y en la posición (x,y). Dentro de las variantes de kernels de convolución discretos de este operador podemos mencionar el uso de kernels de convolución como el dado por (2.3) y (2.4), propuestos por Riaz, et al., en [Riaz et al., 2008] y el dado por (2.5) propuesto por Subbarao et al., en [Subbarao et al., 1993].

$$H_1 = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{2.3}$$

$$H_2 = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \tag{2.4}$$

$$H_3 = \begin{bmatrix} -1 & -4 & -1 \\ -4 & 20 & -4 \\ -1 & -4 & -1 \end{bmatrix} \tag{2.5}$$

- **Medidas basadas en operadores estadísticos:** Estos métodos basan su trabajo en análisis estadísticos: conteos, media, varianza, entropía e histogramas. Por mencionar algunos de los más recurrentes en la literatura, podemos listar:

  - **Varianza:** Es común encontrar en la literatura autores que hacen referencia a la varianza como un medidor de calidad de una imagen, por ejemplo en [Sun et al., 2004, Huang y Jing, 2007, Chun y Kong, 2014, Shirvaikar, 2004, Shi et al., 2005, Pertuz et al., 2013] se menciona el uso de la varianza como una medida de calidad. Esta medida de calidad es definida para una imagen I_m como en la ecuación (2.6).

$$\sigma^2 = \frac{1}{N_r \times N_c} \sum_{x=0}^{Nr-1} \sum_{y=0}^{Nc-1} (I_m(x,y) - \mu_m)^2 \tag{2.6}$$

    donde μ_m es la media de la imagen I_m y se calcula con la ecuación (2.7).

$$\mu_m = \frac{1}{N_r \times N_c} \sum_{x=0}^{Nr-1} \sum_{y=0}^{Nc-1} I_m(x,y) \tag{2.7}$$

  - **Entropía:** Es una medida basada en histogramas o en probabilidades la cual es planteada en [Sun et al., 2004], [Chun y Kong, 2014], [Shirvaikar, 2004], [Shi et al., 2005] y [Pertuz et al., 2013], como una medida de calidad de una imagen y se define por la ecuación (2.8).

$$S = -\sum_{\forall i} p_i log_2(p_i) \tag{2.8}$$

    donde p_i es la probabilidad de la intensidad i en la imagen I_m y puede ser obtenida del histograma de la imagen, por ejemplo para una imagen en escala de grises con profundidad de 8 bits $0 \leq i < 256$.

  - **Momento central absoluto:** Shirvaikar., en [Shirvaikar, 2004] y Pertuz et al., en [Pertuz et al., 2013], establecen que se puede usar el momento central absoluto como medida de calidad de una imagen. El momento central absoluto lo define Shirvaikar en [Shirvaikar, 2004] como se muestra en la ecuación (2.9).

$$\mu_n = \sum_{\forall i} |i - \mu| p(i) \tag{2.9}$$

donde μ_n es el n-ésimo momento central, i es el i-ésimo nivel de gris, μ es el valor de la intensidad media del histograma y $p(i)$ es la frecuencia del i-ésimo nivel de gris de la imagen I_m.

- **Eigenvalores:** Wee y Paramesran en [Wee y Paramesran, 2007], proponen el uso de eigenvalores como una medida de nivel de enfoque robusta bajo condiciones de emborronamiento o de ruido. Dicho planteamiento lo basan en que una de las deficiencias de la mayoría de los métodos que se usan para medir el nivel de enfoque, es que la gran mayoría de ellos basan su análisis en premisa de que las regiones nítidas de una imagen presentan una respuesta alta a los filtros pasa altas, lo cual es correcto, sin embargo no se toma en cuenta que el ruido, también tiene una respuesta alta. Wee y Paramesran reportan muy buenos resultados de la aplicación de dicha técnica, sin embargo plantean que la complejidad computacional es muy alta.

- **Frecuencia espacial:** Una de las medidas de claridad más comúnmente utilizada es la propuesta por Fisher y Eskicioglu en [Eskicioglu y Fisher, 1995]; aunque ésta técnica implica el cálculo de primeras derivadas, sus autores la plantean como una definición de la frecuencia en el dominio del espacio, la cual indica el nivel de actividad global de una imagen I_m. Esta medición se logra aplicando las ecuaciones (2.11) para calcular la frecuencia por columnas, (2.10) para calcular la frecuencia por renglones y 2.12 para calcular la frecuencia espacial de la imagen y en consecuencia conocer el nivel de actividad global de la misma.

$$F_R = \sqrt{\frac{1}{N_r N_c} \sum_{x=0}^{N_r-1} \sum_{y=0}^{N_c-1} [I_m(x,y) - I_m(x,y-1)]^2} \tag{2.10}$$

$$F_C = \sqrt{\frac{1}{N_r N_c} \sum_{x=0}^{N_r-1} \sum_{y=0}^{N_c-1} [I_m(x,y) - I_m(x-1,y)]^2} \tag{2.11}$$

$$F_S = \sqrt{F_R^2 + F_C^2} \tag{2.12}$$

La idea de Fisher y Eskicioglu, fue retomada por Li et al., en [Li et al., 2001], donde se propone que la frecuencia espacial puede ser utilizada para medir el nivel de claridad de una imagen. Además Li et al., establecen que la frecuencia espacial puede ser calculada no sólo para medir el nivel de actividad global de la imagen sino que también se puede usar para calcular el nivel de actividad de un bloque o ventana de la imagen I_k de tamaño $M \times N$ usando las ecuaciones (2.13), (2.14) y (2.15).

$$RF = \sqrt{\frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [I_k(x,y) - I_k(x,y-1)]^2} \tag{2.13}$$

$$CF = \sqrt{\frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [I_k(x,y) - I_k(x-1,y)]^2} \tag{2.14}$$

$$SF = \sqrt{RF^2 + CF^2} \tag{2.15}$$

2.1.1. Uso de derivadas en análisis de imágenes

González y Woods en [González y Woods, 2008], explican que las derivadas de funciones digitales son una medida de diferencias y que aunque se pueden implementar de distintas formas, se requiere que cualquier implementación de derivada garantice que el resultado:

- Para primeras derivadas:
  - Debe ser cero en áreas planas (con nivel de gris constante).
  - Debe ser distinto de cero en el inicio de cambio de nivel de gris.
  - Debe ser distinto de cero a lo largo de una rampa.
- Para segundas derivadas

- Debe ser cero en áreas planas.
- Debe ser distinto de cero en inicio y cambio de nivel de gris o en rampas con pendiente que no es constante.
- Debe ser cero en rampas con pendiente constante.

González y Woods en [González y Woods, 2008], plantean también que el cálculo del Laplaciano de una función de dos dimensiones f puede hacerse con la ecuación (2.16).

$$\nabla^2 f = [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)] - 4f(x,y) \tag{2.16}$$

La frecuencia espacial de una imagen tiene cierta similitud con las derivadas, pues también en áreas planas el valor es cero y en las áreas que son cambiantes arroja un valor distinto de cero, por lo que se puede deducir que también se pueden usar las derivadas para medir el nivel de nitidez de una imagen.

2.2. Métodos para resolver el problema de fusión de imágenes multi-foco

En la literatura son presentadas distintas técnicas para resolver el problema de fusión de imágenes multi-foco, desde aquellas que basan su análisis en promediar los valores de cada pixel de las imágenes hasta aquellas que hacen uso de redes neuronales para determinar cuál es la fusión óptima. En esta sección describiremos algunas de las técnicas más utilizadas para resolver el problema de fusión de imágenes multi-foco.

2.2.1. Promedio

La forma más simple de resolver el problema es mediante el promedio entre los valores de cada pixel (x, y) de las n imágenes fuente (generalmente dos), dicha técnica es sencilla de aplicar y no requiere de gran cantidad de recursos. La técnica consiste en calcular cada pixel de la imagen fusionada como un promedio de los valores del conjunto de imágenes

en dicho pixel, como se muestra en la ecuación (2.17), en la que se suman los valores de cada imagen I_k en el pixel $I_k(x, y)$ y se divide por el número de imágenes n.

$$I_F(x,y) = \frac{I_1(x,y) + I_2(x,y) + \ldots + I_n(x,y)}{n} \qquad 0 \leq x < N_c \;\text{ y }\; 0 \leq y < N_c \tag{2.17}$$

Cabe mencionar que aunque éste método es muy simple, su desempeño no es aceptable en imágenes con problemas de registro, por lo que no resulta viable su aplicación en imágenes reales.

2.2.2. División por bloques y medición de nitidez con frecuencia espacial

Li et al., [Li et al., 2001] proponen dividir cada imagen fuente en bloques y crear la imagen fusionada I_F como una combinación de bloques de las imágenes fuente, discriminando de acuerdo a la frecuencia espacial de cada bloque. Así dadas dos imágenes fuente A y B se descomponen en bloques de tamaño $M \times N$, decidiendo entre el i-ésimo bloque A_i y B_i comparando la frecuencia espacial SF_i^A y SF_i^B de los dos bloques usando las ecuaciones (2.13),(2.14) y (2.15) y aplicando (2.18).

$$I_{F_i} = \begin{cases} A_i & SF_i^A > SF_i^A + TH \\ B_i & SF_i^B > SF_i^B + TH \\ \frac{A_i + B_i}{2} & \text{En otro caso.} \end{cases} \tag{2.18}$$

Cabe mencionar que ésta técnica es equivalente a medir el nivel de actividad de la imagen en las direcciones vertical y horizontal de cada bloque. Li et al., establecen que para aplicar el método se asume que las imágenes han sido previamente registradas, por lo que no se garantiza un adecuado funcionamiento del método en imágenes que tienen problema de registro.

2.2.3. Redes neuronales artificiales

Las redes neuronales también han sido utilizadas para resolver el problema de fusión de imágenes multi-foco, por ejemplo Li et al., [Li et al., 2002] usan medidas de frecuencia espacial para entrenar una red neuronal, la cual, dados dos bloques (uno de cada

imagen fuente), selecciona el bloque con mayor claridad (frecuencia espacial calculada con la ecuación (2.15)) y la imagen final es creada con el conjunto de bloques seleccionados.

Zhou et al., en [Zhou et al., 2006], dividen las imágenes fuente en bloques y se hace una evaluación del nivel de enfoque de cada bloque calculando la frecuencia espacial, el gradiente medio y la entropía, para crear un vector de características que constituye la entrada para entrenar una red neuronal pre-alimentada, mediante la cual se hace la discriminación de que bloques formarán la imagen fusionada. Sin embargo se explica que dicha técnica no es recomendable para ser usada con imágenes que no han sido previamente registradas.

Ma et al., en [Ma et al., 2011], plantean el uso de le varianza, la energía del gradiente, la energía del Laplaciano y la frecuencia espacial como medidas de nitidez de una las imágenes fuentes. Dichas medidas de nitidez se usan para alimentar una red neuronal de pulsos con dos canales, en la cual cada red neuronal puede pesar la diferencia de claridad de las imágenes y en base a ello seleccionar la mejor opción para crear la imagen fusionada.

2.2.4. Transformada discreta Wavelet

La Transformada Discreta Wavelet (DWT) ha sido utilizada en algunos trabajos, por ejemplo en [Yang et al., 2014, Tian y Chen, 2010, Malviya y Bhirud, 2009, Yang, 2011, Shi et al., 2005, Li y Yang, 2008]. Ha sido demostrado que dicha técnica es bastante robusta y que permite obtener muy buenos resultados. La técnica consiste básicamente en crear sub-imágenes, aplicando a las imágenes originales un filtro pasa altas y pasa bajas y después haciendo un sub-muestreo del resultado de aplicar los filtros. Finalmente, las sub-imágenes son combinadas usando reglas de fusión por pixeles o por ventanas como es mencionado en [Li y Yang, 2008], la imagen fusionada final es construida aplicando la transformada inversa.

2.3. Trabajos recientes

Aunque existe gran variedad de técnicas propuestas en la literatura para solucionar el problema de fusión de imágenes multi-foco, podemos listar algunos de los trabajos más prominentes publicados en ésta área:

- Gou et al., en [Guo et al., 2015], plantean un método usando regiones adaptativas. El cual consisten en parches cuadrados con traslape, planteados en [Qu et al., 2014] y calculando la claridad por cada una de las regiones adaptativas, se hace una votación y el pixel es seleccionado de la imagen con más votos, después se estima la información de profundidad de acuerdo a lo planteado en [Zhuo y Sim, 2011], dicha estimación combinada con el gradiente de la imagen se usa para medir la claridad en regiones adaptativas y la imagen fusionada se logra combinando las imágenes originales con un peso, basado en la información de profundidad, gradiente de la imagen y los votos calculados en base a la claridad y las regiones adaptativas.

- Liu et al., en [Liu et al., 2015], calculan descriptores DSIFT (Dense Scale Invariant Feature Transform) y con ellos usando ventanas deslizantes estiman mapas de nivel de actividad de cada imagen fuente, que combinado con la imagen normalizada de los descriptores DSIFT permiten crear un mapa de decisión inicial, el cual es refinado calculando la frecuencia espacial de manera local en las regiones en las que el mapa inicial de decisión marca como inciertas. De acuerdo a lo planteado por los autores éste método logra una mejor calificación que los métodos convencionales de fusión de imágenes ante varios medidores de calidad de imágenes sin embargo tiene la desventaja de tener una complejidad computacional alta.

- Li et al., en [Li et al., 2013] plantean un método de fusión de imágenes el cual consiste en hacer una medición del enfoque de cada imagen fuente usando un filtro morfológico, [De et al., 2006] , para obtener un mapa de nivel de enfoque para cada imagen, después se usa un filtro de medianas y esqueletización para obtener el mapa de nivel de enfoque definitivo para cada imagen, cada imagen es segmentada en tres regiones formando un trimapa que permite calcular los valores de α_n usado para calcular la imagen fusionada de acuerdo a la ecuación (2.19).

$$I_{n,N} = \alpha_n(x,y)I_n(x,y) + (1 - \alpha_n(x,y))I_{n-1,N}(x,y) \tag{2.19}$$

- Cao et al., en [Cao et al., 2015] se muestra el uso de la transformada discreta coseno (DCT) para lograr la fusión de imágenes multi-foco en formato *JPEG*. El método

consiste en decodificar y decuantizar las imágenes fuente para después dividirlas en bloques de 8×8, posteriormente se calcula la frecuencia espacial de cada bloque para después comparar las frecuencias calculadas de cada bloque y crear un mapa de decisión, el cual es refinado aplicando una verificación de consistencia que mejora la calidad de la imagen final usando un filtro de tamaño 3×3, finalmente se cuantizan los coeficientes de la DCT con una tabla de cuantización estándar del estándar JPEG [Wallace, 1991] después se usa la codificación de entropía para producir el flujo de bits de salida. Cao et al reportan un desempeño sobresaliente del método, pues arroja mejores resultados que otros métodos ante diferentes medidas de calidad de imágenes sin embargo sólo se menciona que el método es efectivo en imágenes codificadas en formato JPEG.

- Orozco en [Orozco, 2013] plantea un método de fusión de imágenes multi-foco en el que se calcula un mapa de decisión inicial en base al filtrado de alta y baja frecuencia de las imágenes multi-foco para lo cual se usa la diferencia de Gaussinas como una aproximación del Laplaciano y en base a ello se hace el filtrado de frecuencias. El mapa de decisión inicial es mejorado usando un método de segmentación basado en modelos de Markov planteado en [Rivera et al., 2007]. El método planteado por Orozco arroja resultados muy buenos, sin embargo al implementar los campos aleatorios de Markov para mejorar el mapa de decisión no se toma en cuenta la información de las imágenes multi-foco sino únicamente el mapa de decisión inicial. En [Orozco, 2013], no se reporta el tiempo requerido para el proceso de fusión.

- En [Calderón y Garnica, 2014] se plantea un método de fusión de imágenes basado en combinación lineal de imágenes. Con dicho método se calcula una matriz de pesos p, la cual se usa para hacer la combinación de imágenes. La matriz de pesos se calcula con ayuda del software de Wolfram Mathematica usando el código mostrado en la Figura 2.1, donde F_1 y F_2 son las respuestas a un filtro pasa altas de las imágenes I_1 e I_2 respectivamente y la función NMaximize es una función propia del software, la cual es utilizada en el proceso de optimización de la matriz de pesos.

```
RF = Sum[Sum[(p[x,y] (F1[[x, y]] - F2[[x, y]])), {y, 1, M}], {x, 1, M}];
var = Flatten[Table[p[x,y], {x, 1, M}, {y, 1, M}]];
D1 = Flatten[{Table[p[x,y] ≥ 0, {x, 1, M}, {y, 1, M}],
    Table[p[x,y] ≤ 1, {x, 1, M}, {y, 1, M}],
    Table[p[x,y] - p[x,y-1] ≤ e, {x, 1, M}, {y, 2, M}],
    Table[p[x,y-1] - p[x,y] ≤ e, {x, 1, M}, {y, 2, M}],
    Table[p[x,y] - p[x-1,y-1] ≤ e, {x, 2, M}, {y, 2, M}],
    Table[p[x-1,y-1] - p[x,y] ≤ e, {x, 2, M}, {y, 2, M}],
    Table[p[x,y] - p[x-1,y] ≤ e, {x, 2, M}, {y, 1, M}],
    Table[p[x-1,y] - p[x,y] ≤ e, {x, 2, M}, {y, 1, M}]
   }];
{t, sol} = NMaximize[{RF, D1}, var] // Timing;
t
v = Table[p[x,y] /. Last[sol], {x, 1, M}, {y, 1, M}];
```

Figura 2.1: Código implementado en Wolfram Mathematica

Los resultados arrojados por el método planteado en [Calderón y Garnica, 2014] son muy buenos sin embargo el tiempo necesario para el cálculo de la matriz de pesos es muy elevado, por ejemplo para calcular una matriz de pesos de tamaño 512×512, el método tardó 94 minutos.

2.4. Conclusiones del capítulo

El problema de fusión de imágenes multi-foco, ha sido resuelto utilizando distintas técnicas, sin embargo, hasta donde tenemos conocimiento, no existe una técnica que garantice un desempeño eficiente y que además se pueda aplicar en tiempo real, es por esta razón, que es necesaria la investigación y el desarrollo de técnicas que permitan mejorar los tiempos y resultados reportados en la literatura. Muchas de las técnicas presentadas en la literatura parten del supuesto de que las imágenes están perfectamente alineadas, lo cual no siempre es así, por lo que resulta necesario aplicar un método de registro de imágenes antes hacer el proceso de fusión para garantizar un óptimo desempeño del método. En general, los métodos planteados en la literatura establecen que es necesario calcular de alguna manera la nitidez de la imagen, usando para ello distintas técnicas. Muchas de las cuales están basadas en derivadas y en filtros pasa-altas.

Capítulo 3

Registro de imágenes

De la observación de las imágenes multi-foco se ha llegado a la conclusión de que el cambio en el nivel de enfoque provoca en dichas imágenes una desalineación espacial entre sí. Dicha situación nos genera un grave problema si aplicamos alguna técnica de fusión de imágenes multi-foco como la planteada por Li et al., en [Li et al., 2001] y que consiste en crear la imagen fusionada aplicando la ecuación (2.18), pues al hacer la comparación entre las dos imágenes para determinar cual posee mejor enfoque, podríamos estar comparando dos posiciones distintas de la escena original e incluso si hacemos el promedio de cada pixel en las imágenes multi-foco aplicando la ecuación (2.17) podríamos estar promediando valores concernientes a posiciones diferentes de la escena original, lo que nos generaría una imagen fusionada que no necesariamente es la mejor. Es por esta razón que antes de pensar en realizar el proceso de fusión de imágenes, debemos garantizar que las imágenes estén perfectamente alineadas. Es decir, que el pixel $I_k(x, y)$ debe corresponder a la misma posición en la escena que el pixel $I_j(x, y)$. Para ello necesitamos implementar un método de registro de imágenes que nos permita lograr la alineación de las imágenes multi-foco. Debemos mencionar que al modificar el nivel de enfoque de una lente se genera una transformación que parece no ser la misma para todos los pixeles de la imagen, es por eso que se ha pensado en el uso de un método de registro no-paramétrico que nos permita calcular la transformación que ha sufrido cada pixel de manera independiente.

Liu y Yu en [Liu y Yu, 2015], plantean que la mayoría de los métodos propuestos

para las solución del problema de fusión de imágenes multi-foco parten del supuesto de que las imágenes están perfectamente alienadas (no tienen problema de registro), lo cual no siempre es cierto. Por ende plantean la importancia de aplicar una técnica de registro que permita alinear las imágenes. Liu y Yu usan un detector de BRISK (Binary Robust Invariant Scalable Keypoints) y un descriptor de características en el proceso de alineación para lograr la alineación de las múltiples imágenes fuente y posteriormente se usa la Transformada Wavelet estacionaria para lograr la fusión de las imágenes. Finalmente los autores reportan imágenes comparativas del resultado de la fusión con métodos que no usan técnicas de registro y el método propuesto que si las usa. Al observar las imágenes reportadas en [Liu y Yu, 2015] se evidencía la importancia de registrar las imágenes antes de realizar la fusión.

Fischer y Modersittzki en [Fischer y Modersitzki, 2008] y Brown en [Brown, 1992], definen el registro de imágenes como el proceso de alinear dos o más imágenes tomadas de la misma escena, en diferente momento, con diferente enfoque o con diferente sensor. En [Brown, 1992] se menciona que las transformaciones más comunes son, rígidas, afines, proyectivas, perspectivas y polinomiales. En [Fischer y Modersitzki, 2008] también se menciona que el registro rígido no es lo suficientemente descriptivo y que es limitado.

De acuerdo a lo planteado por Calderón et al., [Calderón et al., 2006b] el problema de registro paramétrico de imágenes consiste en el proceso de encontrar un conjunto de parámetros el cual permite alinear una imagen origen con una imagen destino. En el registro paramétrico se aplica la misma transformación a todos los pixeles y dado que al cambiar el nivel de enfoque de la cámara cada objeto en la escena sufre una transformación diferente en función de la distancia a la que se encuentra de la lente, no es viable aplicar un método de registro paramétrico para alinear las imágenes multi-foco.

3.1. Registro no paramétrico

El registro no-paramétrico es definido en [Pennec et al., 2007], como un proceso de optimización que tiene como objetivo encontrar el desplazamiento de cada pixel con el fin de alinear de manera razonable dos imágenes. La diferencia entre el registro paramétrico y el no-paramétrico radica en que en el registro paramétrico se busca un único conjunto de parámetros que son aplicados a todas las coordenadas de la imagen origen con el fin de

alinearla con la imagen destino, mientras que en el registro no-paramétrico se obtiene un desplazamiento para cada pixel de manera independiente, lo que permite encontrar transformaciones aún y cuando éstas no sean transformaciones afines. Dado que se encuentra un desplazamiento para cada pixel de manera independiente, se puede lograr hacer un empatamiento razonable en imágenes que han sufrido transformaciones no-rígidas o deformaciones.

3.1.1. Método SSD-ARC

Los métodos basados en suma de diferencias al cuadrado (SSD por sus siglas en inglés), permiten hacer una medición de proximidad. Dicha situación ha sido aprovechada para plantear varios métodos que se aplican al procesamiento de imágenes. Por mencionar algunos podemos citar [Lucas et al., 1981] donde se aplica SSD para el registro de imágenes y [Tomasi y Kanade, 1991] donde es aplicado para el cálculo de flujo óptico.

En [Rivera y Marroquin, 2000] se propone el uso de la condición de resorte adaptable (ARC por sus siglas en inglés), para la preservación de bordes en una técnica de regularización. Calderón y Marroquín en [Calderón y Marroquín, 2003], hacen uso de SSD, ARC y el espacio de escalas usado en [Romeny, 1994], para plantear un algoritmo para el cálculo de flujo óptico. Basados en la premisa de que el cálculo del flujo óptico es similar al problema de registro, se aplica dicho algoritmo en el registro de imágenes, produciendo muy buenos resultados. Otros trabajos en esta dirección son los desarrollados en [Calderón et al., 2006a], [Calderón et al., 2007], [Calderón y Flores, 2010] y [Calderón y Romero, 2004], en los que se ha aplicado el método SSD-ARC, demostrando un muy buen desempeño.

El planteamiento del método SSD-ARC para el registro de dos imágenes I_1 e I_2 está dado en la ecuación (3.1), en la que el objetivo es encontrar aquellos valores de V_i y l_i que minimizan el error entre las dos imágenes, asumiendo que el error mínimo, significa que las imágenes se encuentran completamente alineadas.

$$U_{SSD-ARC}(V,l) = \sum_{i\in L}[E(V_i) - l_i H(V_i)]^2 + \frac{\lambda\mu}{4}\sum_{i\in L}|\nabla V_i|^2 + \mu\sum_{i\in L} l_i^2 \tag{3.1}$$

con

$$H(V_i) = |\nabla I_1(r_i + V_i)| \tag{3.2}$$

$$E(V_i) = I_1(ri + V_i) - I_2(r_i) \tag{3.3}$$

donde $E(V_i)$ es el error entre las imágenes I_1 e I_2 y depende del vector de desplazamiento V_i, el cual para dos dimensiones estará formado por dos valores $V_i = [u_i, v_i]$, los cuales representan el desplazamiento que ha sufrido I_2 con respecto a I_1, en la dirección x y y, respectivamente. El vector r_i indica la posición en la que se está calculando el error y para dos dimensiones consta de dos valores $r_i = [x, y]$. El producto de $H(V_i)$ y l_i representa la condición de reposo del resorte y es controlada por el parámetro μ. Dado que en el caso de imágenes las variables u y v son matrices, no se puede permitir que haya grandes cambios entre el valor en las coordenadas (x, y) y los vecinos en u y v, por lo que se usa el parámetro λ el cual permite controlar la suavidad que debe existir en las matrices u y v.

La implementación del método SSD-ARC en el registro de imágenes consiste básicamente en hacer uso de las pirámides Gaussianas para llevar las imágenes a un tamaño menor con el objetivo de que el desplazamiento también sea menor. Una vez que se tienen las imágenes en un tamaño reducido, se hace un cálculo de las matrices U^k y V^k, en las cuales se guarda el desplazamiento calculado para cada pixel de las imágenes en la escala k. Las matrices U^k y V^k son escaladas y multiplicadas por 2, para usarlas como base en el cálculo de las matrices U^{k-1} y V^{k-1}, dicho procedimiento se sigue hasta obtener las matrices U^0 y V^0, las cuales representan la solución.

Como ejemplo del uso del método SSD-ARC mostramos las imágenes de una escultura y un edificio en el fondo, las cuales ya fueron analizadas en el Capítulo 1 y se muestran en las Figuras 3.1(a) y 3.1(b).

(a) ***Imagen 1*** **(b)** ***Imagen 2***

Figura 3.1: Problema de registro en las imágenes de la escultura.

Nótese en las Figuras 3.1(a) y 3.1(b) que los puntos marcados no se encuentran en la misma posición en ambas imágenes, por lo que es necesario registrar las imágenes para que los puntos se encuentren ubicados en las mimas coordenadas. Si calculamos la diferencia entre las imágenes de las Figuras 3.1(a) y 3.1(b) se obtiene como resultado la imagen mostrada en la Figura 3.2, en la cual se marca con blanco los valores altos de diferencia y con negro los valores cercanos a cero. Se puede notar en esta figura que hay gran diferencia entre las imágenes, esto se debe a la transformación que sufre la imagen capturada, al variar el nivel de enfoque.

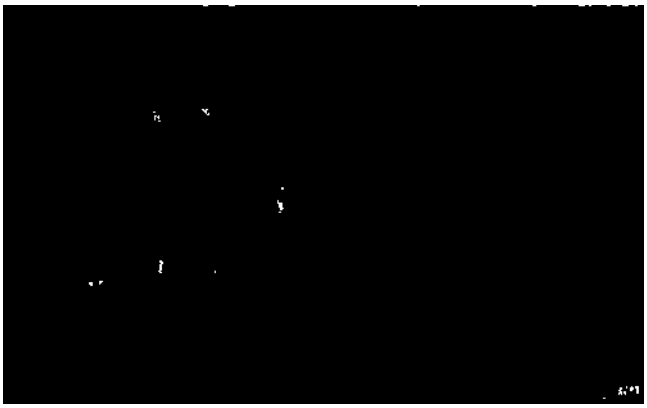


Figura 3.2: Diferencia entre las Figuras 3.1(a) y 3.1(b).

Para aplicar un método de registro se debe seleccionar una imagen de referencia, la cual quedará fija y la otra será la que sufra la transformación. En este caso aplicaremos SSD-ARC con la imagen I_1 mostrada en la Figura 3.1(a) como referencia y calculando los valores de las matrices u y v que se aplicarán a la imagen I_2 de la Figura 3.1(b), para obtener una imagen registrada a la que denominamos I_M aplicando la ecuación (3.4) para calcularla.

$$I_M(x, y) = I_2(x + u(x, y), y + v(x, y)) \tag{3.4}$$

El procedimiento aplicado para registrar una imagen I_2 y alinearla con I_1 se muestra en la Figura 3.3, en donde se ilustra que usando SSD-ARC, se pueden calcular las matrices u y v y en base a ellas y aplicando la ecuación (3.4) se calcula la imagen registrada I_M.

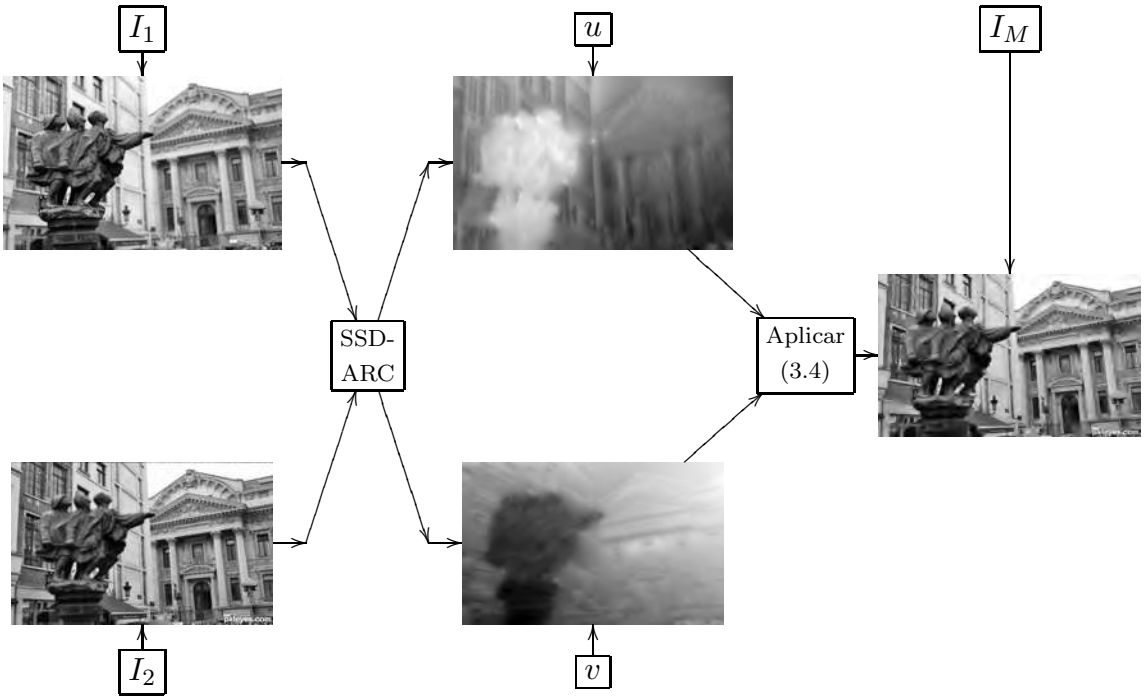


Figura 3.3: Esquema del proceso a seguir para alinear I_2 con I_1.

Al aplicar el proceso de registro obtenemos las matrices u y v que indican el desplazamiento que se debe aplicar a I_2 para alinearla con I_1. En la Figura 3.4(a) se muestra la dirección del desplazamiento que se ha aplicado a los pixeles de I_2 para obtener la imagen registrada I_M.

(a) ***Imagen*** I_M **(b)** ***Vectores de desplazamiento***

Figura 3.4: I_M calculada con SSD-ARC consumiendo 32 seg. para el cálculo.

En la imagen mostrada en la Figura 3.4(a) se puede observar que el desplazamiento de los pixeles, no sigue el mismo patrón en toda el área de la imagen, sino que es diferente de acuerdo a la región. Dicha situación sustenta nuestra hipótesis, de que es necesario aplicar un método de registro no paramétrico para calcular la desalineación sufrida por las imágenes al variar enfoque. En la Figura 3.5 se muestra la diferencia entre I_1 e I_M, se puede notar que gracias a la alineación que se aplicó, la diferencia entre ambas imágenes prácticamente desapareció.

Figura 3.5: Diferencia entre las imágenes de la escultura después del registro.

Para corroborar que el proceso de registro se haya realizado correctamente, revisamos los puntos marcados y podemos notar en la Figura 3.6 que ahora están perfectamente alineados, es decir, se encuentran en las mismas coordenadas tanto en I_1 como en I_M.

(a) I_1 **(b)** I_M

Figura 3.6: Resultado del proceso de registro en las imágenes de la escultura.

Presentamos otro ejemplo del problema registro de imágenes, para el cual tenemos las imágenes multi-foco mostradas en la Figura 3.7 en la que aparece una imagen en la que se ha enfocado en una flor, 3.7(a) y otra en la que se ha enfocado en el fondo de la escena, 3.7(b). Nótese que el tamaño de la flor cambia de una imagen a otra además de que también cambia la posición. Si se aplica un algoritmo para fusionar las imágenes, el resultado seguramente se será erróneo, debido a que al comparar la nitidez del pixel $I_1(x, y)$ con la nitidez de $I_2(x, y)$ estaremos comparando la nitidez de dos puntos diferentes de la escena. Por lo anterior, es necesario primero aplicar el proceso de registro para posteriormente poder hacer la comparación de pixeles que tienen correspondencia exacta.

(a) I_1 **(b)** I_2

Figura 3.7: Problema de registro en imágenes multi-foco en las imágenes del diente de león

Se ha aplicado el método SSD-ARC a las imágenes multi-foco mostradas en 3.7 y se obtiene la imagen resultante mostrada en la Figura 3.8(a). En la Figura 3.8(b) se muestra el desplazamiento que se ha aplicado a los pixeles de la imagen 3.7(b) para obtener la imagen 3.8(a).

(a) ***Imagen I_M calculada con el uso de SSD-ARC aplicado a las Figuras 3.7(a) y 3.7(b).***

(b) ***Vectores de desplazamiento calculados con el método SSD-ARC para las imágenes de las Figuras 3.7(a) y 3.7(b).***

Figura 3.8: Resultado del registro de las imágenes 3.7(a) y 3.7(b), en un tiempo de 80 seg.

Al observar los desplazamientos ilustrados en la Figura 3.8(b) se puede notar que el desplazamiento es mayor en la región de la flor, con lo que se logra que el tamaño y posición de la flor en I_M coincida con el tamaño y posición de la flor en la imagen I_1.

3.2. Conclusiones del capítulo

En éste capítulo se ha analizado la necesidad de aplicar técnicas de registro a las imágenes multi-foco, con el fin de alinear las imágenes antes de aplicar algún algoritmo de fusión. En nuestro caso utilizaremos el método SSD-ARC para el registro de las imágenes multi-foco. En las pruebas mostradas en éste capítulo, se ha podido observar que el método SSD-ARC funciona de muy buena manera para registrar las imágenes multi-foco, pues se logra identificar el desplazamiento que sufre cada pixel de manera independiente, con lo que se logra una mejor alineación de las imágenes.

El tiempo consumido por el método SSD-ARC es relativamente bajo, pues para calcular las matrices u y v en las imágenes de la escultura que tienen un tamaño de 373×600 tarda 32 seg., mientras que para las imágenes del diente de león que tienen un tamaño de 89×600 se consumieron 80 seg.

Capítulo 4

Detección de regiones de alto enfoque para combinación lineal de imágenes

Una de las tareas principales para resolver el problema de fusión de imágenes multi-foco es encontrar, de manera automática, las regiones que presentan la mayor nitidez o enfoque en la imagen. En este capítulo presentamos el proceso a seguir para simular el desenfoque de una imagen y discutiremos algunas de las técnicas que se presentan en la literatura para medir el nivel de enfoque de una imagen. Se presenta una técnica para generar imágenes desenfocadas sintéticamente porque de este modo conocemos las regiones que fueron desenfocadas y podemos medir el desempeño de los métodos planteados para estimar la nitidez y también podemos medir el desempeño de los algoritmos presentados para la fusión de imágenes multi-foco, lo cual nos servirá como base para determinar el mejor algoritmo y aplicarlo en imágenes reales.

4.1. Simulación del desenfoque

De acuerdo a lo planteado por Elder y Zucker, el problema de desenfoque puede modelarse mediante la convolución con un kernel Gaussiano (4.1), [Elder y Zucker, 1998,

Bae y Durand, 2007, Riaz et al., 2008].

$$G(x, y; \mu, \sigma) = g(x; \mu, \sigma) \times g(y; \mu, \sigma) \tag{4.1}$$

donde $g(x; \mu, \sigma)$ es la ecuación de una Gaussina, en una dimensión, con media μ y desviación estándar σ representada en la ecuación (4.2).

$$g(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{4.2}$$

Así dada una imagen original I_0 y un kernel Gaussiano, podemos calcular una imagen desenfocada J_0 con (4.3).

$$J_0(x, y) = I_0(x, y) * G(x, y; \mu, \sigma) \tag{4.3}$$

donde la convolución es representada por el símbolo $*$. La convolución de una señal Y con un kernel X está dada por la ecuación (4.4).

$$Z(n) = \sum_{k=0}^{N-1} X(k)Y(n-k) \tag{4.4}$$

lo cual se puede reescribir en una dimensión como:

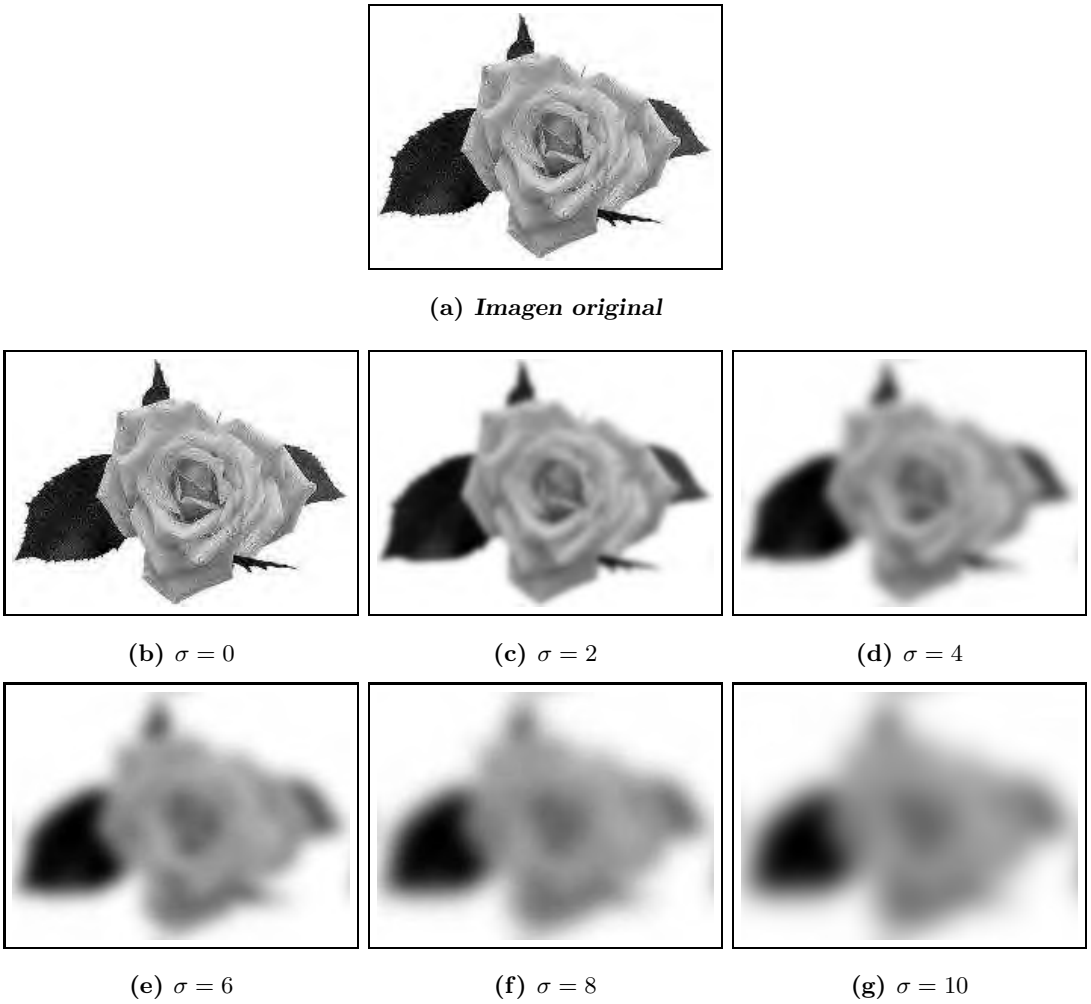
$$X * Y(n) = \sum_{k=0}^{N-1} X(k)Y(n-k) \tag{4.5}$$

donde N es el número de muestras de la señal Y y en dos dimensiones se puede escribir como:

$$X * Y(n, m) = \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} X(k, l)Y(n-k, m-l) \tag{4.6}$$

Según Riaz et al, al aplicar el kernel Gaussiano, se atenúan las frecuencias altas mientras que las frecuencias bajas no sufren cambio, [Riaz et al., 2008], lo que nos permite plantear que el hecho de aplicar el kernel obtenido al aplicar la ecuación (4.1) es equivalente a aplicar un filtro pasa bajas y es un modelo del círculo de confusión de una lente, definido en [Sezan et al., 1991]. De acuerdo con esto, el problema de desenfoque o emborronamiento puede plantearse como la pérdida de las altas frecuencias.

Para mostrar el efecto del desenfoque consideremos la imagen mostrada en la Figura 4.1(a), la cual convolucionamos con un conjunto de Gaussianas con diferente valor de desviación estándar σ, los resultados de esto se muestran en las Figuras 4.1(b)-4.1(g).



(a) ***Imagen original***

(b) $\sigma = 0$ **(c)** $\sigma = 2$ **(d)** $\sigma = 4$

(e) $\sigma = 6$ **(f)** $\sigma = 8$ **(g)** $\sigma = 10$

Figura 4.1: Simulación del desenfoque mediante una convolución con un kernel Gaussiano

Dado que el valor de la desviación estándar σ determina el diámetro de la Gaussiana, a medida que el valor de σ disminuye el diámetro de la Gaussiana disminuye también. Debido a que no es posible calcular una Gaussiana con la ecuación (4.1) utilizando $\sigma = 0$, se calcula el límite el cual es un impulso tal como se indica en la ecuación (4.7). La convolución de la imagen con un impulso, da como resultado la misma imagen. Nótese en la Figura 4.1, que a medida que el valor de la desviación estándar σ aumenta, se pierden los detalles en la imagen, es decir perdemos la componente de alta frecuencia y en consecuencia la imagen se observa borrosa o desenfocada.

$$\lim_{\sigma \to 0} g(x, \mu, \sigma) = \delta(x - \mu) \tag{4.7}$$

El proceso de convolucionar la imagen con una Gaussiana, puede interpretarse como el hecho de crear círculos de confusión cuyo diámetro varía en función del valor de la desviación estándar σ, por lo que cada pixel es recalculado como una combinación de los valores de los pixeles que están dentro del círculo de confusión hipotético.

4.2. Creación de un par de imágenes multi-foco sintéticas

Dadas las consideraciones anteriores y con el fin de ejemplificar el problema de desenfoque en imágenes multi-foco, explicaremos el proceso para crear un par de imágenes desenfocadas de manera sintética a partir de una imagen original I_0 y una imagen borrosa J_0. Como ejemplo utilizaremos las imágenes de un par de lobos las cuales se muestran en las Figuras 4.2(a) y 4.2(b).

(a) ***Imagen de dos lobos*** I_0 **(b)** ***Imagen sintética*** J_0***, fuera de foco.***

Figura 4.2: Generación de una imagen borrosa sintéticamente

Para lograr el efecto de desenfoque sólo sobre algunas partes de la imagen definiremos la matriz binaria p_0 mostrada en la Figura 4.3, la cual nos servirá para crear el par de imágenes sintéticas a partir de las imágenes I_0 y J_0. En la Figura 4.3, se muestran en blanco aquellos pixeles donde $p_0(x, y) = 1$ y en negro aquellos donde $p_0(x, y) = 0$.

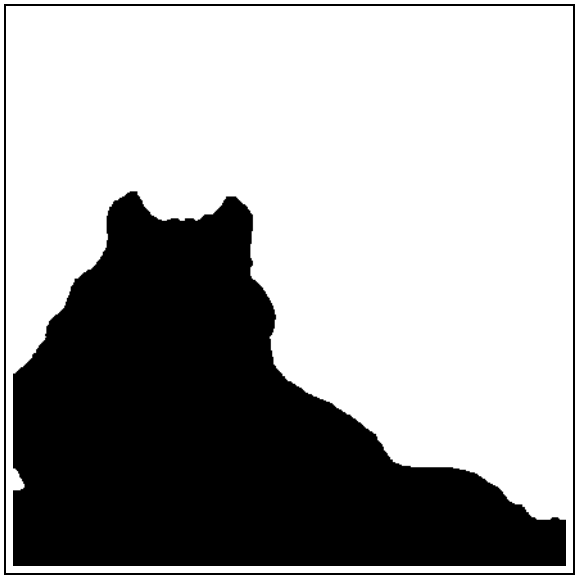


Figura 4.3: Matriz de pesos p_0 binaria

Basados en la matriz de pesos p_0 y con el uso de las ecuaciones (4.8), (4.9) y (4.10) se puede crear un par de imágenes multi-foco, I_1 e I_2. Las imágenes I_1 e I_2 estarán desenfocadas/enfocadas parcialmente de manera complementaria, es decir, las partes nítidas de I_1 aparecerán desenfocadas en I_2 y viceversa. En la regla (4.8) se indica que en los pixeles donde $p_0(x, y) = 0$, la imagen I_1 aparecerá nítida mientras que los pixeles donde $p_0(x, y) = 1$ dicha imagen aparecerá desenfocada, caso contrario a lo que pasa con I_2, la cual se forma usando la regla (4.9).

$$I_1(x,y) = \begin{cases} J_0(x,y) & \text{Si } p_0(x,y) = 1 \\ I_0(x,y) & \text{en otro caso.} \end{cases} \tag{4.8}$$

$$I_2(x,y) = \begin{cases} J_0(x,y) & \text{Si } p_0(x,y) = 0 \\ I_0(x,y) & \text{en otro caso.} \end{cases} \tag{4.9}$$

Las reglas para calcular las imágenes I_1 e I_2 dadas por (4.8) y (4.9) las podemos expresar en forma matemática con la ecuación (4.10).

$$\begin{aligned} I_1(x,y) &= J_0(x,y)p_0(x,y) + I_0(x,y)(1 - p_0(x,y)) \\ I_2(x,y) &= J_0(x,y)(1 - p_0(x,y)) + I_0(x,y)p_0(x,y) \end{aligned} \tag{4.10}$$

En la Figura 4.4 se muestra el resultado de aplicar la ecuación (4.10) sobre las imágenes de los lobos mostradas en 4.2.

(a) ***Imagen sintética 1*** **(b)** ***Imagen sintética 2***

Figura 4.4: Creación de un par de imágenes multi-foco sintéticas.

Nótese que la matriz binaria p_0, mostrada en la Figura 4.3, ha sido definida tratando de aproximar lo mejor posible los bordes del lobo que aparece enfrente en la imagen de la Figura 4.2(a). Con el fin de simular que para capturar 4.4(a) se han establecido las condiciones en la cámara para obtener el mayor nivel de enfoque en el lobo de enfrente y en consecuencia el desenfoque es sobre el lobo de atrás, mientras que en la Figura 4.4(b) muestra la misma imagen pero con las condiciones de enfoque/desenfoque de manera inversa. Con estas imágenes se pretende clarificar el procedimiento para generar dos imágenes multi-foco sintéticas a partir de una origen.

En algunos trabajos se presenta la deconvolución como una técnica para mejoramiento de imágenes, por ejemplo Yuan et al presentan el uso de técnicas de deconvolución para la recuperación de una imagen original desde una imagen que ha sido degradada, [Yuan et al., 2008]. Starck y Murtagh plantean el uso de técnicas de deconvolución, acompañadas de técnicas de regularización y procesamiento de las imágenes en diferentes escalas, para mejorar imágenes astronómicas, controlando por un lado, la suavidad y por otro lado el

ruido, [Starck y Murtagh, 2006]. Zhou et al presentan la aplicación de la deconvolución con diversas condiciones de contorno, para recuperar una imagen original desde una imagen que ha sufrido emborronamiento y adición de ruido, [Zhou et al., 2014a]. Parecería entonces que basta con averiguar el valor de la desviación estándar σ y aplicar técnicas de deconvolución sobre las imágenes para mejorar la nitidez, sin embargo debemos de considerar que el desenfoque es sólo sobre algunas regiones de la imagen y que no tenemos la menor idea de los valores de desviación estándar σ adecuados. Más aún, el desenfoque no es el mismo en todos los pixeles de la imagen, por lo que aun conociendo un valor de desviación estándar σ, no podríamos resolver el problema de fusión de imágenes multi-foco de esa manera.

Para resolver el problema de fusión, nuestra hipótesis será que las imágenes multi-foco contienen regiones de una supuesta imagen original que está completamente enfocada y otra borrosa. Podemos entonces, considerar que el proceso de fusión de imágenes multi-foco puede ser visto como un proceso inverso al proceso de calcular dos imágenes parcialmente desenfocadas y consiste en calcular los valores de una matriz binaria de pesos p (inversa a p_0), que nos permita determinar, cómo unir dos imágenes multi-foco para obtener una imagen final que debe ser igual a la supuesta imagen original.

Suponiendo que tenemos dos imágenes multi-foco, podemos decir que para calcular la matriz binaria de pesos p, requerimos conocer aquellos pixeles son más nítidos en la imagen I_1 que en la imagen I_2, así como aquellos pixeles que son más nítidos en la imagen I_2 que en I_1, de tal forma que la imagen fusionada I_F sea creada de usando la regla de fusión dada en (4.11).

$$I_F(x,y) = \begin{cases} I_1(x,y) & \text{Si } F(I_1(x,y)) > F(I_2(x,y)) \\ I_2(x,y) & \text{en otro caso.} \end{cases} \tag{4.11}$$

donde $F(I_k(x,y))$ es una función que nos permite medir la nitidez de la imagen I_k en las coordenadas (x,y). Lo anterior nos lleva a la necesidad definir la función F, que permita medir el nivel de enfoque de una imagen y en base a ello discriminar de manera automática, las regiones de cada imagen que tienen mayor enfoque o nitidez así como aquellas que deberán ser descartadas por estar desenfocadas.

4.3. Medición del nivel de enfoque

Aunque en el estado del arte se presentan varias propuestas de solución para el problema de fusión de imágenes multi-foco, es recurrente encontrar métodos basados en el análisis de frecuencias para decidir qué áreas extraer de cada una de las imágenes para crear la imagen fusionada. Dichos trabajos basan su investigación en el hecho de que las regiones que presentan mayor nitidez en una imagen, tienen también una respuesta a la frecuencia mayor que aquellas que aparecen de manera borrosa o desenfocada.

Riaz et al plantean que existen distintas formas de medir el nivel de enfoque de una imagen y dentro de los operadores que existen mencionan que los filtros pasa altas son muy efectivos para este fin. También se explica que una técnica para filtrar las frecuencias altas es usar las segundas derivadas, en particular el operador Laplaciano, [Riaz et al., 2008]. El Laplaciano se define como la suma de las segundas derivadas en la dirección x y y de una función f en dos dimensiones como se muestra en la ecuación (4.12).

$$\nabla^2 f = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} \tag{4.12}$$

donde $\frac{\partial^2 f(x,y)}{\partial x^2}$ y $\frac{\partial^2 f(x,y)}{\partial y^2}$ son las segundas derivadas de la función f en la dirección x y y respectivamente.

Haciendo un análisis de las diversas maneras de medir la calidad de una imagen que fueron listadas en la Sección 2.1, podemos ver que prácticamente todas estas medidas están basadas en el análisis de que tan variante es la imagen, ya sea calculando la diferencia entre un pixel y sus vecinos (primeras o segundas derivadas), diferencia entre cada pixel y la media o midiendo de alguna manera que tan variante es la imagen. Por otro lado, sabemos que un filtro pasa altas nos permite medir también (aunque de manera indirecta), las variaciones en la imagen, pues un valor de respuesta a un filtro pasa altas con una magnitud alta, indica una imagen altamente variante mientras que si la magnitud de la respuesta al filtro pasa altas se acerca a cero será un indicador de que la imagen es poco variante y en consecuencia posee poca información de alta frecuencia.

Fisher y Eskicioglu en [Eskicioglu y Fisher, 1995] y Li et al en [Li et al., 2001] presentan una técnica para medir el nivel de actividad de una imagen o bloque mediante el uso de la frecuencia espacial, para lo cual se hace el cálculo de la diferencia entre cada pixel (x, y) y sus

dos pixeles vecinos $(x-1, y)$ y $(x, y-1)$. Si definimos las primeras derivadas de la imagen I_m en las direcciones x y y como lo mostrado en las ecuaciones (4.13) y (4.14) respectivamente, podemos notar que dichas fórmulas se encuentran implícitas en las fórmulas para el cálculo de la frecuencia por renglón y por columna mostradas en las ecuaciones (2.13) y (2.14).

$$d_x(x, y) = I_m(x, y) - I_m(x - 1, y) \tag{4.13}$$

$$d_y(x, y) = I_m(x, y) - I_m(x, y - 1) \tag{4.14}$$

De lo anterior podemos decir que para el cálculo de la frecuencia espacial se hace una medición del nivel de actividad en las direcciones x y y. Lo que se propone en este trabajo de tesis es medir el nivel de actividad en todas las direcciones posibles. Esto se puede hacer calculando todas las primeras derivadas direccionales posibles que involucran a los pixeles que se encuentran en la vecindad mostrada en la Figura 4.5.

$(x-1, y-1)$	$(x-1, y)$	$(x-1, y+1)$
$(x, y-1)$	(x, y)	$(x, y+1)$
$(x+1, y-1)$	$(x+1, y)$	$(x+1, y+1)$

Figura 4.5: Vecindad del pixel ubicado en la coordenada (x, y)

A las suma de las diferencias del pixel $I_m(x, y)$ con sus vecinos lo llamaremos E y lo calculamos con la ecuación (4.15), en la que sumamos las diferencias del pixel $I_m(x, y)$ con cada uno de sus ocho vecinos más cercanos.

$$E(x,y) = \sum_{k=-1}^{1} \sum_{l=-1}^{1} (I_m(x,y) - I_m(x-k, y-l)) \tag{4.15}$$

Si expandimos la sumatoria de la ecuación (4.15) tenemos la ecuación (4.16), lo que podemos plantear como la suma de los valores de una retícula como la mostrada en la Figura 4.6.

$$\begin{aligned} E(x,y) = 8I_m(x,y) - I_m(x,y-1) - I_m(x,y+1) - I_m(x-1,y) - I_m(x+1,y) - \\ I_m(x-1,y-1) - I_m(x-1,y+1) - I_m(x+1,y-1) - I_m(x+1,y+1) \end{aligned} \tag{4.16}$$

$-I_m(x-1,y-1)$	$-I_m(x-1,y)$	$-I_m(x-1,y+1)$
$-I_m(x,y-1)$	$8I_m(x,y)$	$-I-m(x,y+1)$
$-I_m(x+1,y-1)$	$-I_m(x+1,y)$	$-I_m(x+1,y+1)$

Figura 4.6: Representación de le ecuación (4.16) como la suma de los valores de una retícula.

Al sumar los valores de los pixeles representados en la retícula mostrada en la Figura 4.6 podemos extraer el kernel de convolución H mostrado en (4.17).

$$H = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \tag{4.17}$$

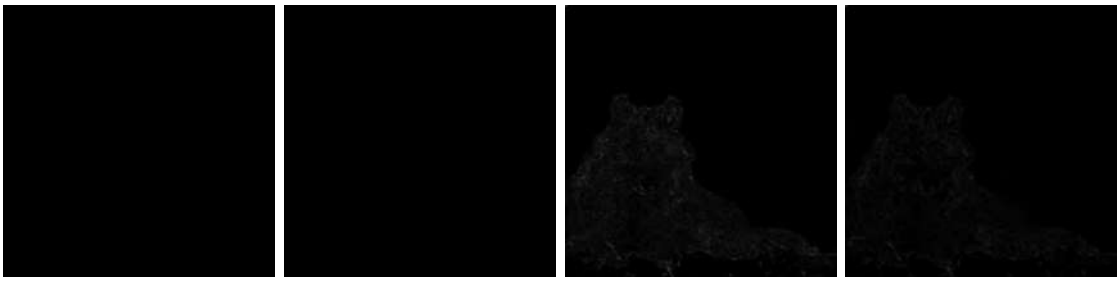
El aplicar el kernel de convolución H puede ser visto como la medición del nivel

de actividad en las 8 direcciones y además dicho kernel es una discretización del Laplaciano que ya ha sido planteada como un medidor de calidad de la imagen en [Riaz et al., 2008].

Existen diversas maneras de aproximar el Laplaciano de manera discreta, la forma común es calcular el Laplaciano de Gauss, que consiste en calcular las segundas derivadas Gaussianas. Otra manera de aproximar el Laplaciano es usando kernels de convolución como el mostrado en la ecuación (4.17) que de acuerdo con lo planteado en [Riaz et al., 2008] es una discretización del Laplaciano.

Como ejemplo de la medición del nivel de enfoque usando el operador Laplaciano, se ha aplicado dicho operador sobre la imagen de los lobos mostrada en la Figura 4.4(a). Dado que para el cálculo de las derivadas Gaussianas se requiere establecer el valor de la desviación estándar σ, hemos realizado pruebas para establecer el valor de σ que ofrece mejor resultado para efectos de detectar las partes nítidas de la imagen. En la figura 4.7 se muestra el valor absoluto de la respuesta de aplicar el Laplaciano de Gauss sobre la imagen 4.4(a) a medida que se varía el valor de σ usado para el cálculo de las derivadas Gaussianas.

(a) $\sigma = 0.3$ **(b)** $\sigma = 0.4$ **(c)** $\sigma = 0.5$ **(d)** $\sigma = 0.6$



(e) $\sigma = 0.7$ **(f)** $\sigma = 0.8$ **(g)** $\sigma = 0.9$ **(h)** $\sigma = 1.0$

Figura 4.7: Imágenes de estimación del nivel de enfoque usando el Laplaciano de Gauss (LoG) variando el valor de σ.

La Figura 4.8 muestra el valor absoluto de la respuesta de aplicar el kernel del Laplaciano discretizado H mostrado en (4.17) a la imagen mostrada en la Figura 4.4(a).

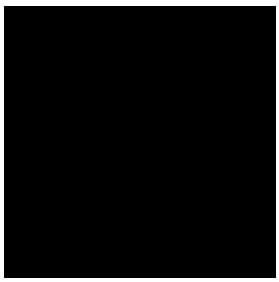
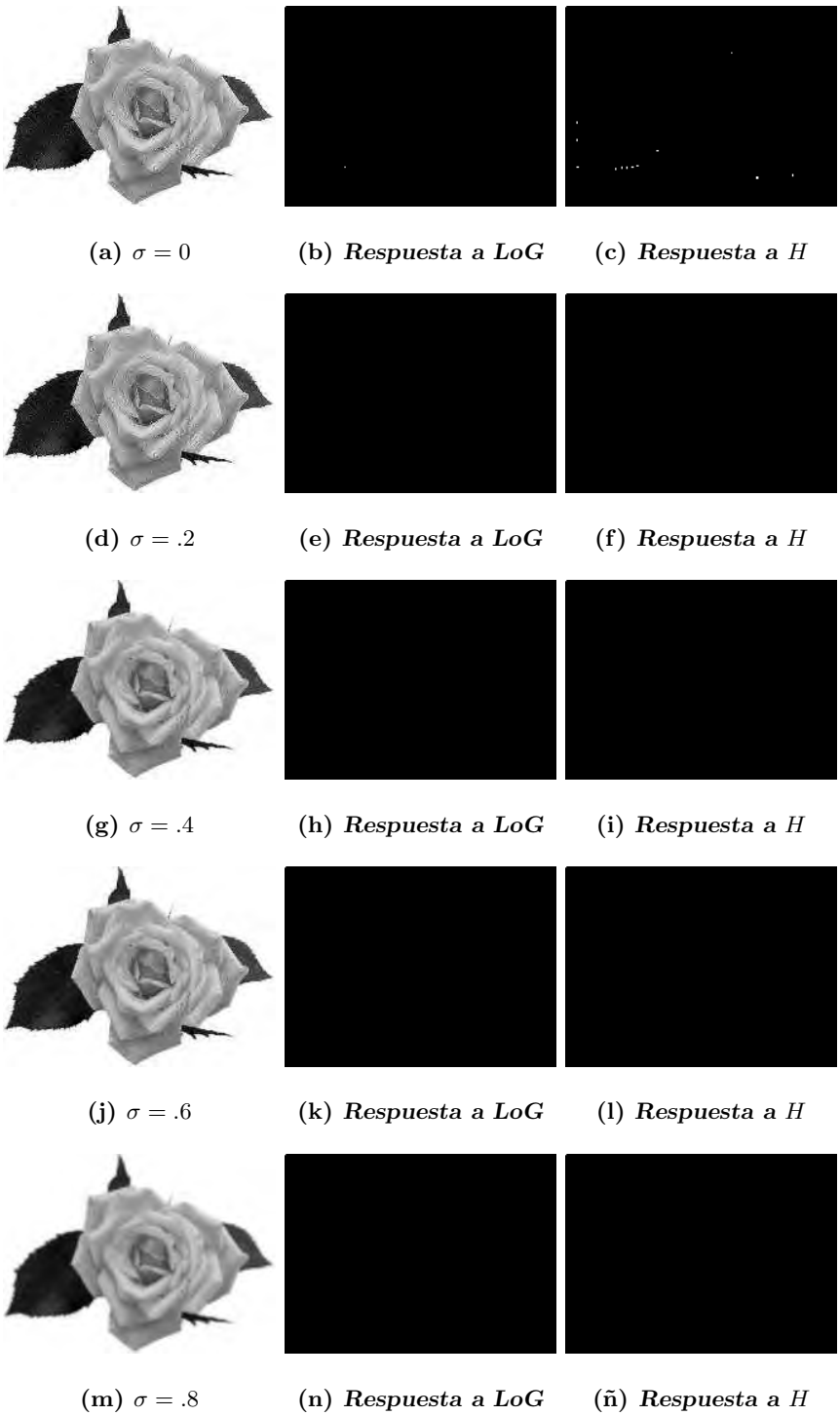


Figura 4.8: Medición del nivel de enfoque usando el kernel discretizado del Laplaciano.

De la observación de las Figuras 4.7 y 4.8 podemos notar tres cosas:

1. Se puede identificar las regiones nítidas usando el operador Laplaciano.
2. La medición del nivel de enfoque usando el Laplaciano de Gauss, requiere establecer un valor de σ adecuado para el cálculo de las derivadas Gaussianas.
3. La convolución con el kernel del Laplaciano discretizado H, también permite identificar las regiones nítidas de una imagen, con la ventaja de que al aplicar dicho kernel, no es necesario calcular o establecer ningún parámetro.

En la figura: 4.9, se muestran tres columnas, en la columna de la izquierda se muestra la imagen de una flor correspondiente a la mostrada en la Figura 4.1(a), en la que se ha simulado el desenfoque mediante la convolución con una Gaussiana, variando el valor de desviación estándar σ usado para el desenfoque. En la segunda columna se muestra el valor absoluto de la respuesta al Laplaciano de Gauss (LoG) aplicado a la imagen de la primera columna, usando $\sigma = 0.8$ para el cálculo de las derivadas Gaussianas y en la tercera columna se muestra el valor absoluto de la respuesta de aplicar el kernel de convolución del Laplaciano discretizado H a la imagen de la primera columna, se representan los valores altos en blanco y los valores cercanos a cero con negro. Se puede observar que a medida que la imagen de la flor es más borrosa, la magnitud de los valores de la respuesta al filtro Laplaciano es menor.



(a) $\sigma = 0$ **(b)** ***Respuesta a LoG*** **(c)** ***Respuesta a*** H

(d) $\sigma = .2$ **(e)** ***Respuesta a LoG*** **(f)** ***Respuesta a*** H

(g) $\sigma = .4$ **(h)** ***Respuesta a LoG*** **(i)** ***Respuesta a*** H

(j) $\sigma = .6$ **(k)** ***Respuesta a LoG*** **(l)** ***Respuesta a*** H

(m) $\sigma = .8$ **(n)** ***Respuesta a LoG*** **(ñ)** ***Respuesta a*** H

Figura 4.9: Respuesta al Laplaciano de Gauss (LoG) y al Laplaciano Discreto.

Al observar la Figura 4.9, podemos notar que el kernel de convolución del Laplaciano discretizado también permite detectar las regiones nítidas en una imagen además de que dicho kernel no requiere de establecer ningún parámetro para su aplicación y que debido a su tamaño, la convolución es muy rápida mientras que el tiempo requerido para cálculo del Lapalciano de Gauss se incrementa a medida que aumenta el valor de σ utilizado en el cálculo de las derivadas Gaussianas.

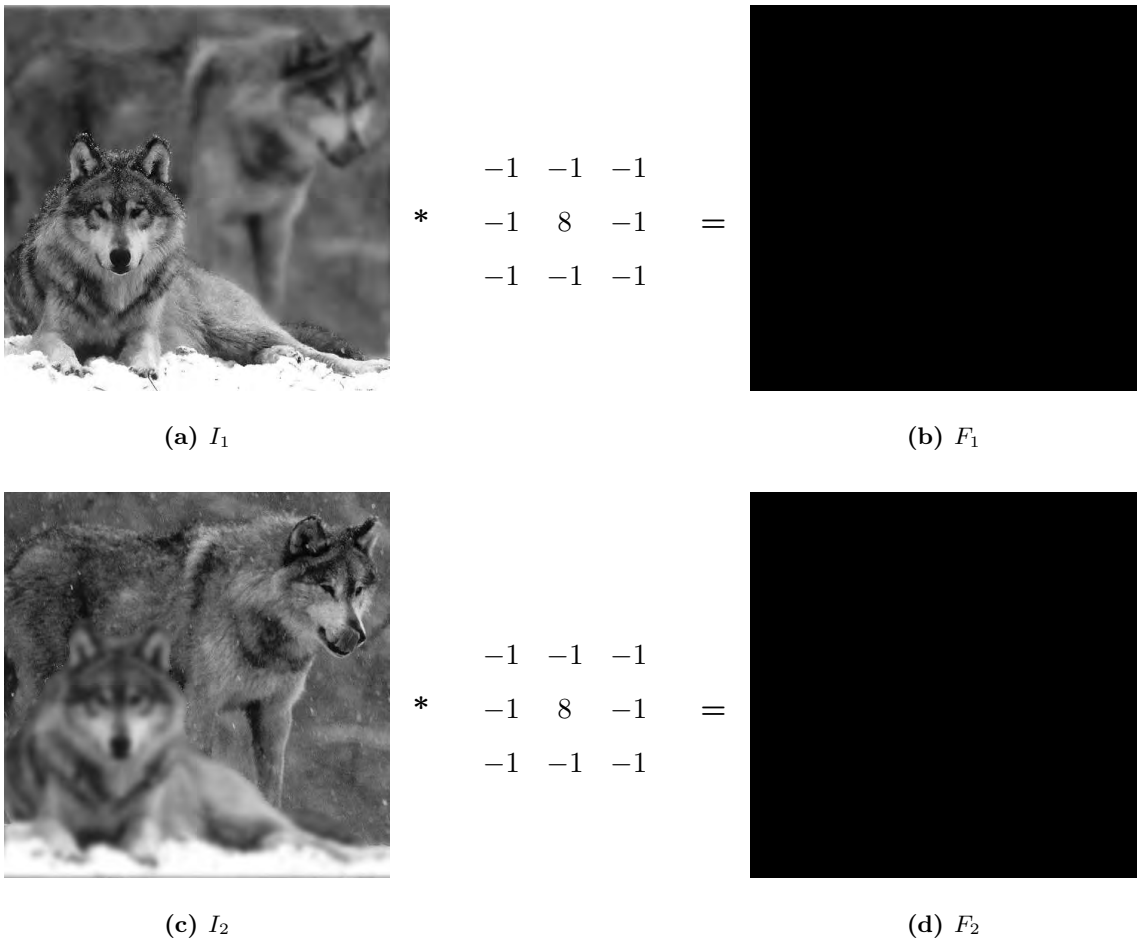
En función del análisis que se ha hecho, nuestra propuesta es usar la convolución de la imagen con el kernel discretizado del Laplaciano H como medidor del nivel de enfoque. Entonces, planteamos medir la nitidez de una imagen I_k como el valor absoluto de la convolución de I_k con H a lo que llamaremos F_k y para cada pixel $I_k(x, y)$ calcularemos su nivel de enfoque usando la ecuación (4.18).

$$F_k(x, y) = |I_k(x, y) * H| \tag{4.18}$$

Cabe hacer la aclaración de que se toma el valor absoluto del resultado, esto en virtud de que lo que nos interesa saber es la magnitud del resultado. Un resultado con magnitud alta significa una variación alta en la imagen, aún si el resultado es negativo. Mientras que un resultado con magnitud cercana a cero significa que la imagen posee poca información de alta frecuencia en esa posición. Al aplicar la ecuación (4.18) a todos los pares $< x, y >$ de la retícula L tenemos como resultado una matriz bidimensional que nos da información de las regiones de la imagen que tienen mayor nitidez.

Dadas las imágenes mostradas en las Figuras 4.4(a) y 4.4(b), a las que llamaremos I_1 e I_2 respectivamente, aplicamos el kernel de convolución H y calculamos F_1 y F_2 usando la ecuación (4.18). En la Figura 4.10, se aprecia que al aplicar el kernel Laplaciano discretizado a I_1, este detecta las regiones enfocadas, las cuales se muestran en blanco en la Figura 4.10(b) y del mismo modo al aplicar la convolución con el Laplaciano discretizado a I_2 se puede detectar las regiones enfocadas de dicha imagen, lo cual se muestra en 4.10(d).

Por lo anterior podemos hacer notar que el kernel de convolución del Laplaciano discretizado puede ser usado para detectar las regiones de alto enfoque en imágenes multifoco.



(a) I_1 **(b)** F_1

(c) I_2 **(d)** F_2

Figura 4.10: Respuesta de aplicar el kernel discretizado del Laplaciano a imágenes multi-foco

Una vez que se han detectado las regiones enfocadas y desenfocadas de cada imagen multi-foco, tenemos que plantear una estrategia que nos permita combinar I_1 e I_2 para crear la imagen I_F.

4.4. Combinación lineal de imágenes

Dadas las imágenes I_1 e I_2, el objetivo de la fusión de imágenes multi-foco es crear una imagen I_F que contenga las regiones más claras de nuestras imágenes multi-foco. Como se planteó y se mostró con la Sección 4.2, podemos llegar a dos imágenes parcialmente

desenfocadas/enfocadas a partir una imagen original que está completamente enfocada. Sin embargo nuestro interés es: dadas dos imágenes multi-foco, que están parcialmente enfocadas, llegar a una que esté lo más enfocada posible. Como se puede entender, el proceso debe ser inverso a lo planteado en la ecuación (4.10). Podemos entonces definir una ecuación que nos permita crear una imagen I_F que esté completamente enfocada, como una combinación lineal de las imágenes I_1 e I_2, en base a una matriz binaria de pesos p como lo planteado en la ecuación (4.19).

$$I_F(x,y) = I_1(x,y)p(x,y) + I_2(x,y)(1 - p(x,y)) \tag{4.19}$$

Como las imágenes I_1 e I_2 son dadas, el reto entonces, es encontrar aquellos valores de la matriz p que nos permita crear la imagen I_F con la mayor nitidez posible. Adicionalmente, si analizamos las ecuaciones (4.10) y (4.19), podemos notar que I_0 e I_F deben ser equivalentes si queremos que I_F esté completamente enfocada. Si sustituimos I_F por I_0 y en lugar de I_1 e I_2 escribimos la ecuación para calcularlas llegamos a lo siguiente:

$$\begin{aligned} I_0(x,y) = J_0(x,y)p_0(x,y)p(x,y) + I_0(x,y)p(x,y) - I_0(x,y)p_0(x,y)p(x,y) + \\ (J_0(x,y) - J_0(x,y)p_0(x,y) + I_0(x,y)p_0(x,y))(1 - p(x,y)) \end{aligned}$$

$$p_0(x,y) - 1 = (2p_0(x,y) - 1)p(x,y)$$

De lo anterior podemos llegar a la ecuación (4.20):

$$p(x,y) = \frac{p_0(x,y) - 1}{2p_0(x,y) - 1} \tag{4.20}$$

donde si $p_0(x,y) = 0$ entonces la solución es $p(x,y) = 1$ y si $p_0(x,y) = 1$ entonces la solución es $p(x,y) = 0$, es decir los valores son inversos y dado que p_0 y p tienen valores binarios, podemos llegar a la ecuación (4.21).

$$p(x,y) = 1 - p_0(x,y) \tag{4.21}$$

Para las imágenes de las Figuras 4.4(a) y 4.4(b), dado que ya tenemos la matriz p_0, podemos calcular fácilmente el valor de p aplicando la ecuación (4.21), lo cual lo mostramos en la Figura 4.11.

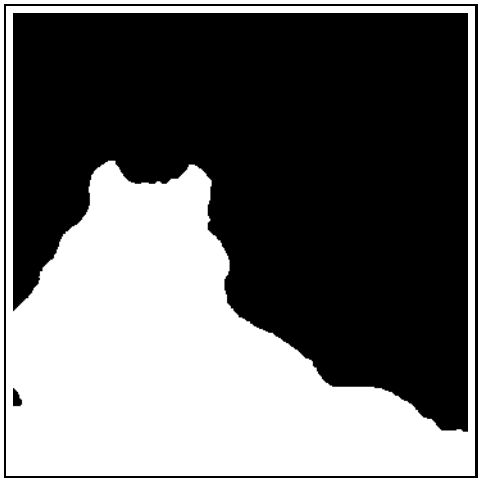


Figura 4.11: Cálculo del valor de p en base a p_0

Si aplicamos la regla de fusión de imágenes dada por (4.19) a las imágenes I_1 e I_2 mostradas en las figuras 4.4(a) y 4.4(b) usando la función de pesos p mostrada en la Figura 4.11 obtenemos la imagen mostrada en la Figura 4.12, con lo que demostramos que teniendo dos imágenes parcialmente desenfocadas y con los valores correctos de p podemos obtener una imagen completamente enfocada.

Figura 4.12: I_F calculada usando la ecuación (4.19) y los valores de p de la Figura 4.11

Se puede ver en la Figura 4.12 que la imagen luce completamente enfocada. Dado p_0, el proceso de fusión sería muy sencillo, sin embargo no es el caso, por lo que tenemos que calcular la matriz de pesos p dado I_1 e I_2. Esto se puede plantear como un problema de optimización lineal, donde el objetivo es que la nitidez de la imagen I_F sea la máxima.

Entonces tenemos que definir una función que nos permita medir la claridad de nuestra imagen fusionada. Basados en la ecuación (4.18) podemos medir la nitidez del pixel $I_F(x, y)$ de la imagen fusionada como:

$$F_F(x, y) = |I_F(x, y) * H| \tag{4.22}$$

Dado que la imagen I_F es creada como una combinación de las imágenes I_1 e I_2, podemos entonces reescribir la ecuación (4.22) como:

$$F_F(x, y) = |(I_1(x, y)p(x, y) + I_2(1 - p(x, y))) * H| \tag{4.23}$$

podemos reescribir la ecuación (4.23) como (4.24), esto en función de que la convolución es asociativa y distributiva y de que los valores de p, I_1 e I_2 son siempre positivos.

$$F_F(x, y) = |I_1(x, y) * H|p(x, y) - |I_2(x, y) * H|p(x, y) + |I_2(x, y) * H| \tag{4.24}$$

Basados en la ecuación (4.18), se definen dos arreglos bidimensionales F_1 y F_2, con lo que podemos reescribir la ecuación (4.24) como (4.25).

$$F_F(x, y) = F_1(x, y)p(x, y) - F_2(x, y)p(x, y) + F_2(x, y) \tag{4.25}$$

que se puede reescribir como (4.26).

$$F_F(x, y) = \Delta F(x, y)p(x, y) + F_2(x, y) \tag{4.26}$$

donde $\Delta F(x, y) = F_1(x, y) - F_2(x, y)$.

Entonces podemos decir que el proceso consiste en calcular los valores de $p^\star$ que hacen que la ecuación (4.26) arroje el valor máximo.

$$\begin{aligned} p^\star(x, y) = \underset{p(x,y)}{máx} \quad & \Delta F(x, y)p(x, y) \\ & \text{Sujeta a} \\ & p(x, y) \in \{0, 1\} \end{aligned} \tag{4.27}$$

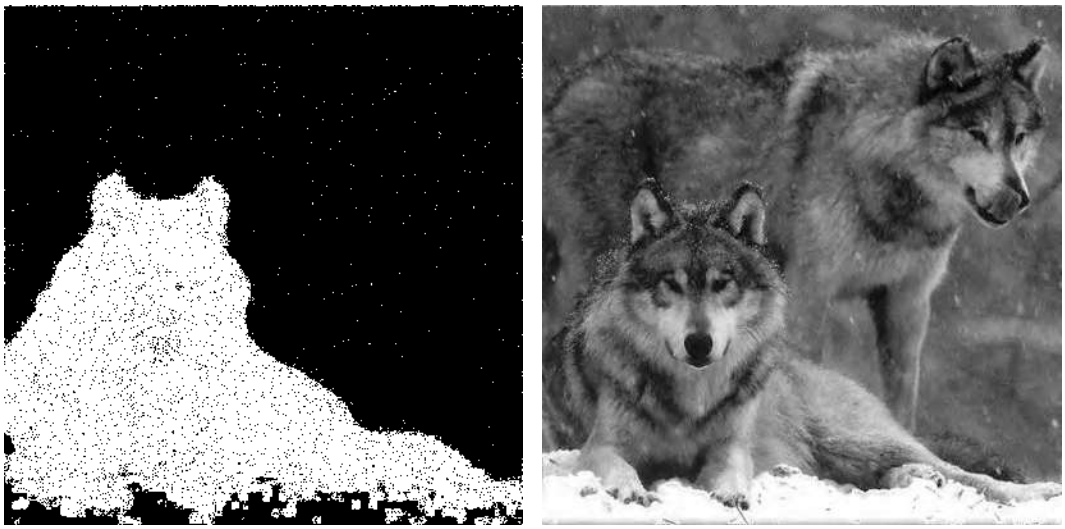
Cabe hacer mención de que el término $F_2(x, y)$ se ha omitido debido a que es constante para el proceso de optimización. El objetivo es encontrar los valores de $p(x, y)$ que hacen que (4.27) arroje el valor máximo posible. Podemos observar de la ecuación (4.27) que si $\Delta F(x, y) \geq 0$ (lo cual implica que $F_1(x, y) > F_2(x, y)$) el valor de $p(x, y)$

que maximiza el resultado de la ecuación (4.27) es $p(x, y) = 1$, indicando con esto que el valor de $I_F(x, y) = I_1(x, y)$ y si por el contrario $\Delta F(x, y) < 0$ (lo cual implica que si $F_2(x, y) > F_1(x, y)$) el valor de $p(x, y)$ que maximiza (4.27) es $p(x, y) = 0$, indicando con esto que el valor de $I_F(x, y) = I_2(x, y)$. Esto garantiza que para formar la imagen fusionada I_F se tomarán de I_1 aquellos pixeles (x, y) dónde $F_1(x, y) \geq F_2(x, y)$ y de I_2 aquellos pixeles (x, y) dónde $F_2(x, y) > F_2(x, y)$.

De este modo tenemos que el reto es resolver el problema de optimización, el cual podemos plantear mediante algún método de programación lineal, por ejemplo el método Simplex. No obstante, dada la naturaleza del problema, la solución está dada por (4.28).

$$p(x, y) = \begin{cases} 1 & \text{Si } F_1(x, y) \geq F_2(x, y) \\ 0 & \text{En otro caso} \end{cases} \tag{4.28}$$

Con (4.28) podemos omitir el proceso de optimización con lo que el cálculo de p se hace mucho más sencillo. Así para el ejemplo de las imágenes de los lobos mostradas en las Figuras 4.4(a) y 4.4(b) aplicamos la regla (4.28) con lo que obtenemos lo mostrado en la Figura 4.13(a). Después de aplicar la regla de fusión (4.19) sobre los valores de I_1, I_2 y p mostrados en las Figuras 4.4(a), 4.4(b) y 4.13(a) respectivamente logramos el resultado mostrado en la Figura 4.13(b).



(a) ***Matriz de pesos*** p **(b)** ***Imagen fusionada***

Figura 4.13: Fusión de imágenes de los lobos usando (4.19) para el cálculo de p

Se puede observar que la función de pesos calculada es una muy buena aproximación a la real mostrada en la Figura 4.11. Más aún, la imagen fusionada mostrada en 4.13(b) parece estar completamente enfocada, sin embargo se puede ver que en 4.13(a) aparecen inconsistencias que se observan como puntos blancos en la región negra y/o puntos negros en la región blanca. Esto se debe a que no se han planteado restricciones que garanticen la coherencia espacial, es por ésta razón que el aplicar la regla dada por (4.28) para calcular la matriz de pesos no es suficiente, pues es necesario aplicar restricciones que nos permitan no sólo garantizar una nitidez aceptable en la imagen fusionada sino que también garantizar la coherencia espacial, tema que se aborda en el siguiente capítulo.

(a) **(b)**

Figura 4.14: Partes nítidas de I_1 e I_2

4.5. Conclusiones del capítulo

En este capítulo se ha planteado la propuesta de utilizar un kernel de convolución para la medición del nivel de nitidez de las imágenes multi-foco, el kernel de convolución que se ha seleccionado es una discretización del Laplaciano el cual es mostrado en (4.17). Dicho kernel es el resultado del cálculo de las primeras derivadas direccionales en todas las posibles direcciones con los 8 vecinos más cercanos. Si lo comparamos con la propuesta

hecha por Li et al también puede ser visto como la medida del nivel de actividad en todas las direcciones posibles. En la Figura 4.10, se puede observar que el kernel propuesto permite identificar las regiones que están enfocadas en cada una de las imágenes sintéticas de los lobos. En la Figura 4.9 se puede observar que tanto el kernel de convolución del Laplaciano discretizado H como el Laplaciano de Gauss LOG permiten identificar las regiones con mayor nitidez de una imagen y que a medida que la imagen es más borrosa la respuesta ante cualquiera de las dos versiones del Laplaciano también disminuye. Para aplicar el kernel de convolución del Laplaciano discretizado H no se requiere calcular ningún parámetro que dependa de la imagen a tratar. Además el cálculo de la convolución de la imagen con el kernel H es muy rápido, a diferencia del cálculo del Laplaciano de Gauss que requiere de más tiempo a medida que se incrementa el valor de σ que se usa para el cálculo de las derivadas Gaussianas, por lo que el kernel de convolución del Laplaciano discretizado H, será el utilizado para medir el nivel de nitidez o enfoque de las imágenes multi-foco. Podemos concluir que el proceso de fusión de imágenes multi-foco puede ser planteado como una combinación lineal de imágenes para lo que es necesario calcular una matriz binaria de pesos p que permita determinar la forma óptima para realizar la combinación. Dicha situación puede ser planteada como un proceso de optimización y aunque al aplicar (4.28), se logra calcular de manera muy aproximada la matriz de pesos p, se puede notar en la Figura 4.13(a) que la matriz p calculada, no es del todo exacta, pues al usarla en (4.19), algunas regiones que deberían ser tomadas de I_1, son tomadas de I_2 y viceversa. Dicha situación se puede corregir aplicando restricciones de coherencia espacial, tema que se aborda en el siguiente capítulo.

Capítulo 5

Solución con restricciones de coherencia espacial

El uso de la regla para calcular la matriz de pesos p mostrada en (4.28) no es suficiente, pues el resultado muestra regiones inconsistentes como las que se pueden observar en la Figura 4.13(a). Esto es debido a que la falta de textura y la falta de nitidez causada por el desenfoque hacen que al calcular la matriz de pesos aplicando la regla (4.28), pixeles vecinos correspondientes a la misma región sean tomados de una imagen o de la otra indistintamente, cuando deberían ser tomados de la misma.

La coherencia espacial restringe a que pixeles vecinos pertenecientes a una misma región tengan un comportamiento de enfoque/desenfoque similar. Se propone hacer el cálculo de la matriz p con valores reales $0 \leq p(x, y) \leq 1$ y el uso de restricciones de coherencia espacial con el fin de garantizar que los pixeles vecinos que pertenecen a un mismo objeto, sean tomados de la misma imagen. Específicamente proponemos que los valores de $p(x, y)$ deben ser parecidos a los valores de sus vecinos. Para lograr que los valores de pixeles vecinos en p sean similares se propone que el valor absoluto de las diferencias entre un valor $p(x, y)$ y sus vecinos sea menor que un valor ϵ, de acuerdo con (5.1). El valor de ϵ controla la semejanza entre dos vecinos, es decir, con el valor de ϵ se establece la diferencia máxima permitida entre un pixel $p(x, y)$ y sus vecinos $N_v(x, y) = \{p(x-1, y), p(x, y-1)\}$.

$$\begin{aligned} |p(x,y) - p(x-1,y)| &< \epsilon \\ |p(x,y) - p(x,y-1)| &< \epsilon \end{aligned} \tag{5.1}$$

Las restricciones de coherencia espacial planteadas en (5.1) pueden ser reformuladas como restricciones lineales dadas por (5.2).

$$\begin{aligned} p(x,y) - p(x-1,y) &< \epsilon \\ p(x-1,y) - p(x,y) &< \epsilon \\ p(x,y) - p(x,y-1) &< \epsilon \\ p(x,y-1) - p(x,y) &< \epsilon \end{aligned} \tag{5.2}$$

5.1. Solución por el método Simplex

Si reunimos los elementos dados, tenemos que el problema de fusión de imágenes multi-foco lo podemos representar como la maximización de un sistema lineal dado por (5.3)

$$\begin{aligned} p^\star = \max_p \quad & \textstyle\sum_{x=0}^{N_r-1} \sum_{y=0}^{N_c-1} \Delta F(x,y) p(x,y) \\ & \text{Sujeta a} \\ & p(x,y) - p(x-1,y) < \epsilon \\ & p(x-1,y) - p(x,y) < \epsilon \\ & p(x,y) - p(x,y-1) < \epsilon \\ & p(x,y-1) - p(x,y) < \epsilon \\ & p(x,y) \leq 1 \\ & p(x,y) \geq 0 \end{aligned} \tag{5.3}$$

$$\forall < x, y > \in L$$

Dado que es un proceso de optimización lineal, lo podemos resolver mediante alguna técnica de programación lineal, por ejemplo el método Simplex. Dado que los pesos $p(x,y)$ se definen binarios para realizar la fusión utilizando (4.19), el resultado obtenido en el proceso de optimización debe ser binarizado aplicando una umbralización la cual está dada

por (5.4).

$$p(x,y) = round(p^{\star}(x,y)) \tag{5.4}$$

Al aplicar la ecuación para formar la imagen fusionada I_F mostrada en (4.19), con la solución binarizada del proceso de optimización dado por el sistema mostrado en (5.3), se garantiza que la imagen fusionada I_F además de tener una nitidez superior a I_1 e I_2, esté formada cumpliendo las restricciones de coherencia espacial que se plantearon en (5.2), sin embargo solucionar el problema utilizando el método Simplex, implica algunos desafíos en memoria y tiempo de cómputo.

Para solucionar el problema usando el método Simplex, tenemos que la forma canónica para el método Simplex está dada por (5.5).

$$\begin{gathered} \text{Máx } z = c_1x_1 + c_2x_2 + \cdots + c_Nx_N \\ \text{Sujeto a} \\ Ax + \mathcal{I}x_h = b \end{gathered} \tag{5.5}$$

donde $\mathcal{I}$ es una matriz que inicialmente es la identidad, pero que durante las iteraciones del método pierde ésta característica y x_h es el vector de variables de holgura para trasformar el problema de desigualdades en igualdades. Los recursos de memoria necesarios para resolver el problema usando el método Simplex hacen imposible su aplicación en términos prácticos. El total de variables N será igual al número de renglones (N_r) por el número de columnas (N_c) de las imágenes, entonces $N = (N_r \times N_c)$. La matriz de restricciones o matriz aumentada A tendrá M renglones y N columnas, donde M es el número de restricciones dado que para cada pixel tenemos las 5 restricciones mostradas en (5.2), tendremos $M = 5 \times N_r \times N_c$, así que la matriz de restricciones A será de tamaño $N \times M = 5N_r^2N_c^2$. Adicionalmente la matriz identidad será de tamaño $M \times M = 25N_r^2N_c^2$. Dadas las consideraciones anteriores, si por ejemplo, tenemos una imagen de tamaño 256×256 y suponiendo que los valores sean almacenados en flotantes de 4 bytes, los recursos necesarios para resolver el problema serán de 80 GB para la matriz de restricciones y 400 GB para la matriz $\mathcal{I}$. Más aún, dichas matrices son altamente dispersas, pues menos del 0.01 % de los valores en ambas matrices son diferentes de cero. Se podría plantear la propuesta de implementar el problema con ayuda de matrices ralas, sin embargo el método Simplex tiene la desventaja de que durante

el proceso iterativo la matriz deja de ser dispersa. Adicionalmente el método Simplex tiene una complejidad computacional $O(2^M)$, donde M es el número de vértices de la región de factibilidad. Dado que no tenemos a nuestro alcance una computadora con los recursos en memoria necesarios, nos es imposible en términos prácticos resolver (5.3) para imágenes de tamaños mayores a 256×256 usando Simplex. En [Calderón y Garnica, 2014] se plantea una solución utilizando la función NMaximize del software de Wolfram Mathemamatica con lo que se logra resolver el problema para imágenes de tamaño 512×512 con muy alto porcentaje de acierto en el resultado, sin embargo el tiempo requerido para la solución fue de 94 minutos, por lo que se hace muy costosa la aplicación usando dicho software.

5.1.1. Ventanas deslizantes

Por las razones expuestas planteamos una propuesta de solución que está basada en la misma premisa, pero haciendo una optimización sobre una región o ventana de las imágenes originales con lo que reducimos los recursos requeridos para solucionar el problema. Entonces, basados en la ecuación (5.3) la optimización se hará sobre una ventana W de tamaño $(2w+1) \times (2w+1)$, centrada en las coordenadas (x, y) y que contendrá todos los pares (k, l) tales que $k \in 0, \ldots, 2w$ y $l \in 0, \ldots, 2w$, como se muestra en (5.6).

$$
\begin{aligned}
p^{\star}_{x,y} = \max_{p_{x,y}} \quad & \textstyle\sum_{k=0}^{2w} \sum_{l=0}^{2w} \Delta F(x-w+k, y-w+l) p_{x,y}(k,l) \\
& \text{Sujeta a} \\
& p_{x,y}(k,l) - p_{x,y}(k-1,l) < \epsilon \\
& p_{x,y}(k-1,l) - p_{x,y}(k,l) < \epsilon \\
& p_{x,y}(k,l) - p_{x,y}(k,l-1) < \epsilon \\
& p_{x,y}(k,l-1) - p_{x,y}(k,l) < \epsilon \\
& p_{x,y}(k,l) \leq 1 \\
& p_{x,y}(k,l) \geq 0
\end{aligned}
\tag{5.6}
$$

Dado que $p_{x,y}$ es una matriz de tamaño $(2w+1) \times (2w+1)$, de cada solución para la matriz $p_{x,y}$, solamente tomaremos el valor central, es decir, cada valor de $p(x, y)$ será calculado usando la información concerniente a los vecinos que se encuentran a una distancia máxima de w pixeles de las coordenadas (x, y). Entonces, la relación que guarda

$p_{x,y}$ con la matriz de pesos p es la mostrada en (5.7), lo que significa que cada valor $p(x, y)$ será sustituido por el valor del centro del arreglo $p_{x,y}$ binarizado.

$$p(x, y) = round(p^{\star}_{x,y}(w, w)) \tag{5.7}$$

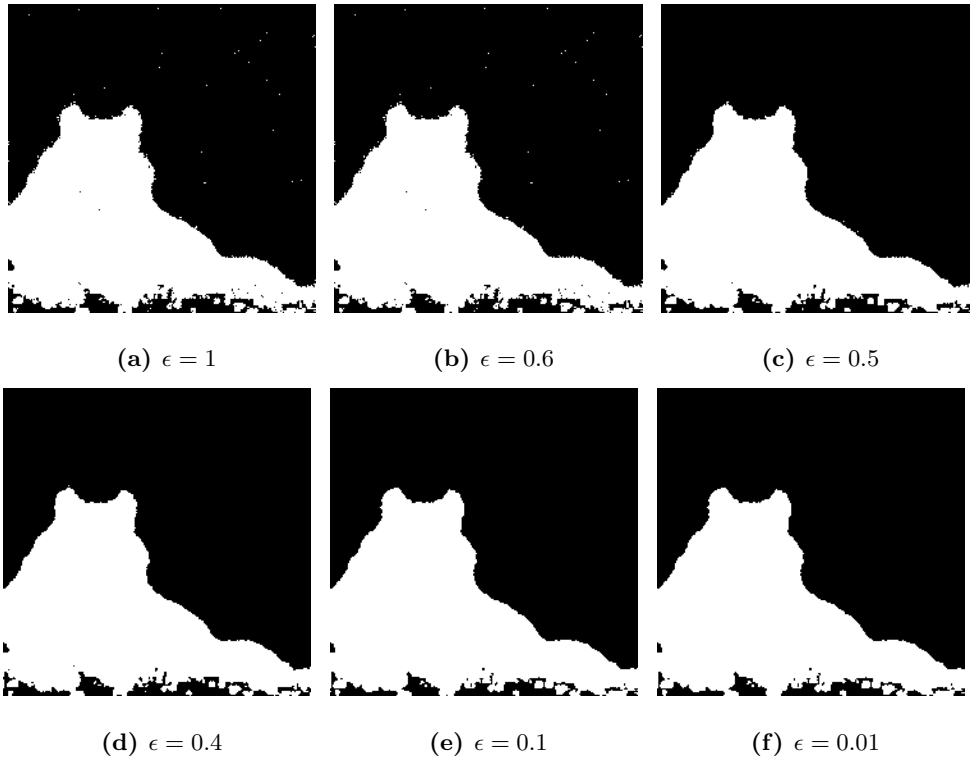
Basados en la ecuación (5.6) podemos plantear el Algoritmo 1 para calcular p por ventanas, al cual denominaremos **CPV** y recibe como parámetros I_1, I_2, H, ϵ y w. en el que se aplica el proceso de optimización por ventanas y en base a ello se calcula la matriz de pesos p, la cual será utilizada para calcular la imagen fusionada I_F utilizando la regla de fusión dada por (4.19).

Algoritmo 1: **CPV(I_1, I_2, H, ϵ, w).**

1: Calcular F_1 y F_2 aplicando la ecuación (4.18) usando H..
2: Calcular $\Delta F = F_1 - F_2$
3: **para todo** $< x, y > \in L$. **hacer**
4: Maximizar (5.6) para calcular p^{*}_{xy} usando ϵ y w.
5: Calcular $p(x, y)$ usando (5.7).
6: **fin para**
7: **devolver** p.

5.2. Coherencia espacial

Para aplicar el Algoritmo 1 requerimos valores para w y para ϵ. Lo primero que haremos es analizar las el impacto que tienen en la coherencia espacial, el variar el valor de ϵ. Fijaremos el valor de $w = 2$, con lo que la ventana será de tamaño 5×5. En la Figura 5.1 se muestra el resultado de aplicar el Algoritmo 1 a las imágenes de los lobos, variando el valor de ϵ. En esta figura se puede observar que los valores de ϵ que mejor resultado ofrecen, para efectos de coherencia espacial, son valores cercanos a cero, por lo que el valor de ϵ que utilizaremos es $\epsilon = 0.01$.



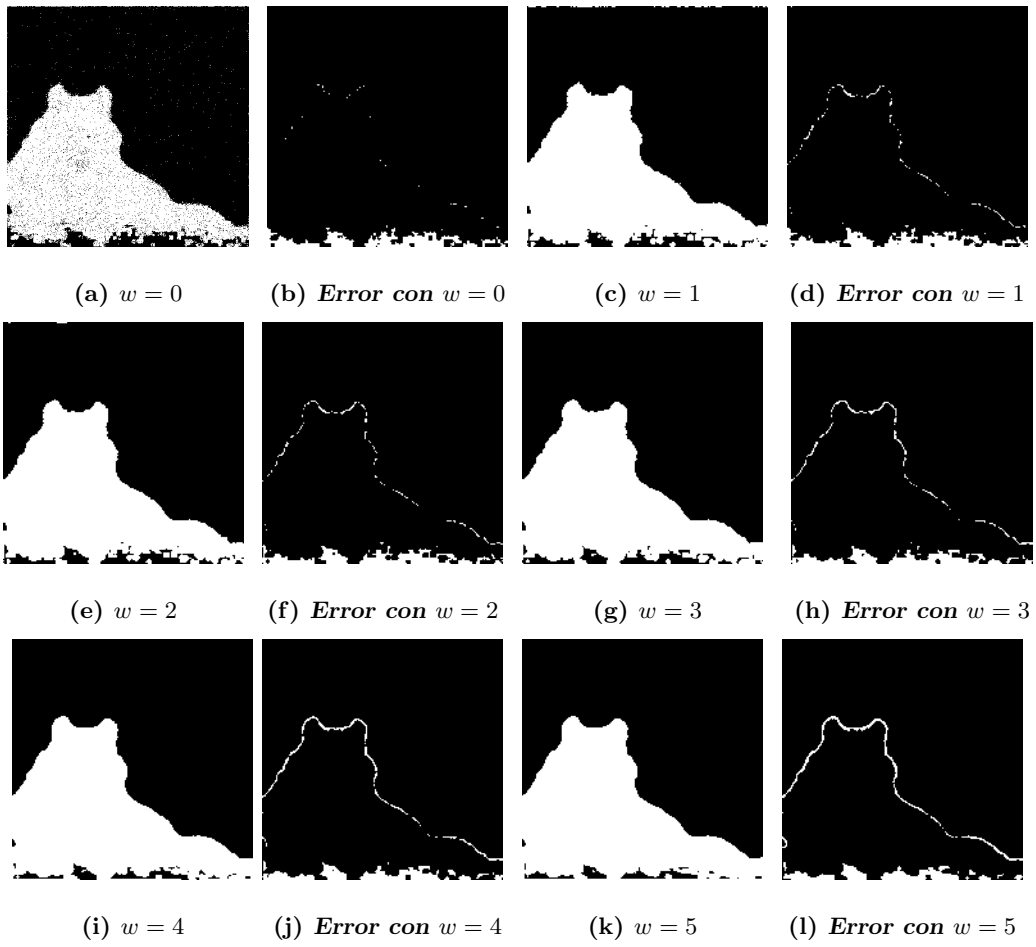
(a) $\epsilon = 1$ **(b)** $\epsilon = 0.6$ **(c)** $\epsilon = 0.5$

(d) $\epsilon = 0.4$ **(e)** $\epsilon = 0.1$ **(f)** $\epsilon = 0.01$

Figura 5.1: Efecto de variar el valor ϵ al aplicar el Algoritmo 1 con $w = 2$.

Con el fin de entender mejor las consecuencias de variar el valor de w, aplicamos el algoritmo CPV sobre las imágenes de los lobos mostradas en 4.4(a) y 4.4(b), variando el valor de $w = 0, 1, 2, 3, 4, 5$ y el resultado se muestra en la Figura 5.2.

En las Figuras 5.2(a), 5.2(c), 5.2(e), 5.2(g), 5.2(i) y 5.2(k), se muestra la matriz de pesos p obtenida aplicando el Algoritmo 1 con $w = 0$, $w = 1$, $w = 2$, $w = 3$, $w = 4$ y $w = 5$ respectivamente y en las Figuras 5.2(b), 5.2(d), 5.2(f), 5.2(h), 5.2(j) y 5.2(l) la diferencia de la función p obtenida con la exacta (que para éste caso es conocida). Podemos observar que cuando $w = 0$ el resultado es el mismo que al aplicar la regla para el cálculo de la matriz de pesos (4.28) en la que no se aplican restricciones de coherencia espacial. Esto es porque las restricciones de coherencia espacial no surten efecto al aplicarse sobre ventanas de un sólo pixel, pero se puede observar que al incrementar el valor de w las inconsistencias disminuyen y dejan de aparecer puntos blancos en las regiones que deben ser negras y viceversa.



(a) $w = 0$ **(b)** ***Error con*** $w = 0$ **(c)** $w = 1$ **(d)** ***Error con*** $w = 1$

(e) $w = 2$ **(f)** ***Error con*** $w = 2$ **(g)** $w = 3$ **(h)** ***Error con*** $w = 3$

(i) $w = 4$ **(j)** ***Error con*** $w = 4$ **(k)** $w = 5$ **(l)** ***Error con*** $w = 5$

Figura 5.2: Efecto de variar el valor w al usar el Algoritmo 1 sobre las imágenes de los lobos.

Dado que para este ejemplo ya conocemos la solución, la podemos comparar con los resultados de aplicar el algoritmo CPV. En la Tabla 5.1 se muestra el porcentaje de acierto y el tiempo consumido por el algoritmo a medida que se incrementa el valor de w.

w	**% de acierto**	**tiempo en seg**
0	92.04	0.271
1	96.13	11.6
2	96.43	371.1
3	96.63	3516.3
4	96.91	19751.9
5	97.07	56261.8

Tabla 5.1: Porcentaje de aciertos al variar el valor de w al aplicar el Algoritmo 1 en las imágenes de los lobos

Se puede notar en la Tabla 5.1 al incrementar el valor de w, se incrementa también el porcentaje de acierto, sin embargo se puede ver que el costo en tiempo, a medida que w crece, se eleva de forma exponencial como se muestra en la gráfica de la Figura 5.2.

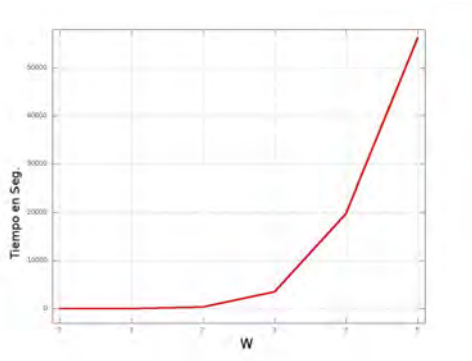


Figura 5.3: Incremento en el tiempo consumido por el algoritmo CPV al variar w.

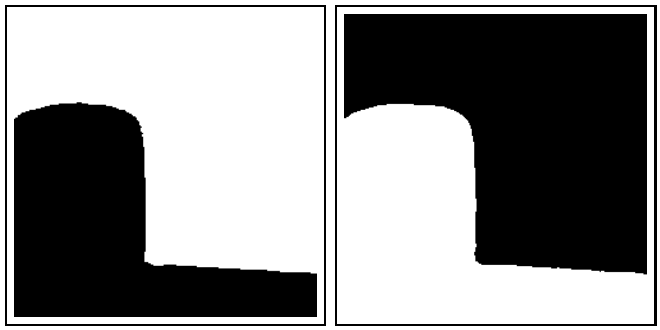
A pesar de que es claro que no se puede implementar el algoritmo CPV con tamaños de ventana cercanos al tamaño de la imagen original, si podemos concluir que a medida que el tamaño de la ventana es mayor, con valores adecuados de ϵ, el resultado será mejor, pues tendremos mayor coherencia espacial. Por lo anterior podemos decir que además del valor de ϵ, también el valor de w nos permite mejorar la coherencia espacial.

Se ha puede observar que para el ejemplo de las imágenes de los lobos que el Algoritmo 1 funciona muy bien para el cálculo de la matriz p. Probaremos el desempeño del Algoritmo 1 en las imágenes multi-foco de los relojes imágenes multi-foco reales, mostradas en la Figura 5.4 y que son muy conocidas en el estado del arte.

(a) I_1 **(b)** I_2

Figura 5.4: Imágenes de los relojes

El problema al que nos enfrentamos es el hecho de que en este caso no contamos con una matriz de pesos p exacta con la que podamos comparar los resultados por lo que nos hemos dado a la tarea de estimar matrices de pesos p_0 y p de forma manual, tratando de aproximar los bordes entre los objetos de la imagen. Dichas matrices p_0 y p se muestran en las Figuras 5.5(a) y 5.5(b) respectivamente. En la Figura 5.6 se muestra el resultado de aplicar el Algoritmo 1 sobre las imágenes de los relojes mostradas en 5.4(a) y 5.4(b).

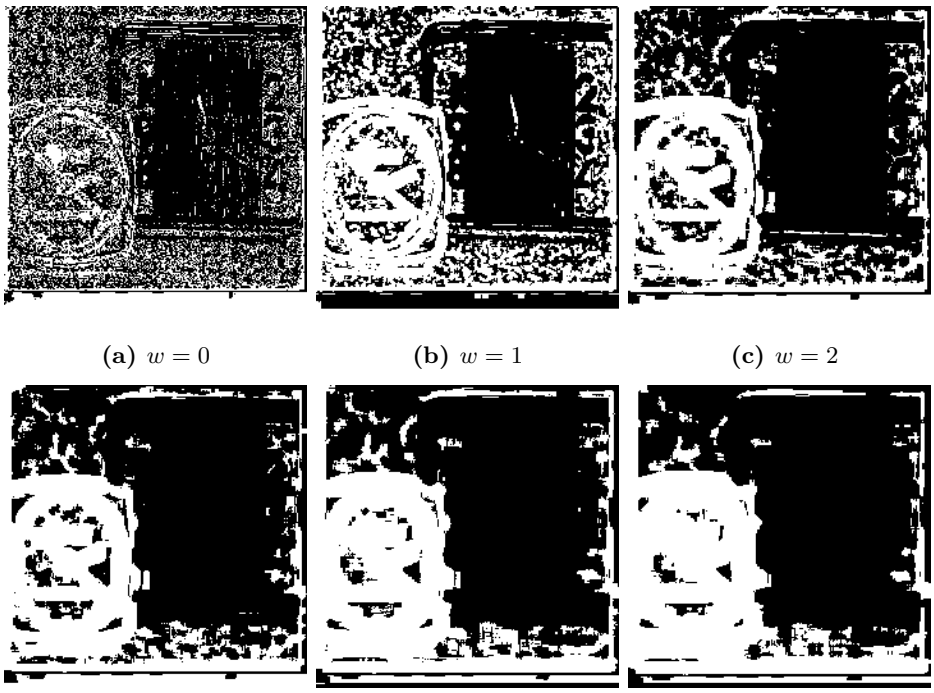


(a) ***Matriz de pesos*** p_0 **(b)** ***Matriz de pesos*** p

Figura 5.5: Matrices de pesos p y p_0 estimadas manualmente para las imágenes de los relojes

Nótese en la Figura 5.6, que al igual que en las imágenes de los lobos, a medida que el valor de w se incrementa, se logra tener una mayor coherencia espacial y las regiones lucen más definidas, puede apreciarse que con $w = 0$, no se logra distinguir una separación entre las dos regiones, sin embargo con $w = 5$ ya se logra notar la región donde está la parte nítida de I_1 (región blanca de p) y a región donde está la parte nítida de I_2 (región negra de p).

A medida que el valor de w es mayor, el resultado que arroja el Algoritmo 1 se acerca más a el resultado esperado, sin embargo el tiempo necesario para el cálculo de p se eleva enormemente a medida que el valor de w crece. Dicha situación se aprecia en la tabla 5.2 en la que se puede observar que el porcentaje de coincidencia entre la matriz p estimada manualmente y la calculada con el uso del Algoritmo 1 aumenta a medida que se incrementa el valor de w y en consecuencia el tamaño de la ventana. Se puede ver también en la misma tabla 5.2, que a medida que la ventana es más grande también el tiempo crece por lo que no resulta viable, la aplicación del Algoritmo 1 con ventanas de tamaño mayor a 11×11.



(a) $w = 0$ **(b)** $w = 1$ **(c)** $w = 2$

(d) $w = 3$ **(e)** $w = 4$ **(f)** $w = 5$

Figura 5.6: Cálculo de p con el Algoritmo 1 para el ejemplo de los relojes al variar w.

De acuerdo a lo planteado en el Algoritmo 1, por cada ventana W, se calcula una matriz $p_{x,y}$ que contiene $(2w+1) \times (2w+1)$ valores, de los cuales únicamente se conserva un valor y el resto de los valores son desechados. Por ejemplo para $w = 4$, se calculan 81 valores, de los cuales 80 son desechados y sólo uno es guardado. Más aún, dado que las ventanas tienen traslape, los valores son calculados y desechados varias veces.

Por las consideraciones anteriores proponemos un nuevo algoritmo en el que se conservan todos los valores calculados para cada $p_{x,y}$. Así para una posición (x, y) tendremos que la relación entre p y $p_{x,y}$ estará dada por la ecuación (5.8).

$$p(x - w + k, y - w + l) \leftarrow p^{\star}_{x,y}(k, l) \tag{5.8}$$

donde $0 \leq k \leq 2w$ y $0 \leq l \leq 2w$.

Al conservar todos los valores de las matrices $p_{x,y}$, tendremos varias soluciones para la matriz de pesos p. Por ejemplo para $p_{5,5}$ y $p_{5,6}$ con $w = 1$ tenemos las relaciones mostradas en la Tabla 5.3.

w	**% de acierto**	**tiempo en seg**
0	63.86	0.314
1	71.64	5.649
2	76.60	169.01
3	80.67	1278.47
4	83.56	8902.56
5	85.97	31225.6

Tabla 5.2: Porcentaje de acierto y tiempo consumido por el Algoritmo 1 sobre las imágenes de los relojes, variando el valor de w.

valores de k y l	p'	p''
$k=-1, l=-1$	$p'(4,4) = p^{\star}_{5,5}(0,0)$	$p''(4,5) = p^{\star}_{5,6}(0,0)$
$k=-1, l=0$	$p'(4,5) = p^{\star}_{5,5}(0,1)$	$p''(4,6) = p^{\star}_{5,6}(0,1)$
$k=-1, l=1$	$p'(4,6) = p^{\star}_{5,5}(0,2)$	$p''(4,7) = p^{\star}_{5,6}(0,2)$
$k=0, l=-1$	$p'(5,4) = p^{\star}_{5,5}(1,0)$	$p''(5,5) = p^{\star}_{5,6}(1,0)$
$k=0, l=0$	$p'(5,5) = p^{\star}_{5,5}(1,1)$	$p''(5,6) = p^{\star}_{5,6}(1,1)$
$k=0, l=1$	$p'(5,6) = p^{\star}_{5,5}(1,2)$	$p''(5,7) = p^{\star}_{5,6}(1,2)$
$k=1, l=-1$	$p'(6,4) = p^{\star}_{5,5}(2,0)$	$p''(6,5) = p^{\star}_{5,6}(2,0)$
$k=1, l=0$	$p'(6,5) = p^{\star}_{5,5}(2,1)$	$p''(6,6) = p^{\star}_{5,6}(2,1)$
$k=1, l=1$	$p'(6,6) = p^{\star}_{5,5}(2,2)$	$p''(6,7) = p^{\star}_{5,6}(2,2)$

Tabla 5.3: Relación de los valores de $p_{5,5}$ y $p_{5,6}$ con la matriz p

Nótese en la Tabla 5.3 que para las posiciones $(4,5)$, $(4,6)$, $(5,5)$, $(5,6)$ $(6,5)$ y $(6,6)$ de p, tenemos dos valores y a medida que calculamos la solución para todas las matrices $p_{x,y}$ tendremos aún más soluciones para cada $p(x,y)$, por lo que es necesario tomar la decisión de que valor asignaremos a cada posición $p(x,y)$. Proponemos calcular el valor de $p(x,y)$ como el promedio de todos los valores obtenidos para cada posición (x,y). Entonces la matriz de pesos p será calculada como lo mostrado en (5.9).

$$p(x,y) = \frac{1}{N_w(x,y)} \sum_{k=-w}^{k=w} \sum_{k=-w}^{k=w} p_{x+k,y+l}(w-k, w-l) \qquad \forall < x,y > \in L \tag{5.9}$$

donde $N_w(x,y)$ es el número total de ventanas en las que aparece $p(x,y)$.

Como se ha dicho, cada valor de la matriz de pesos p se calcula varias veces, esto

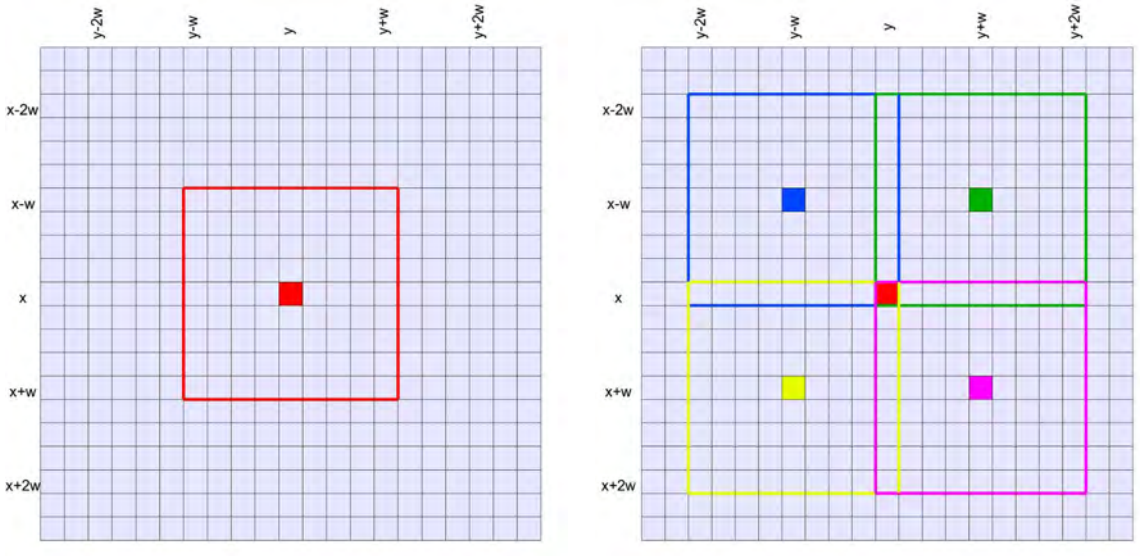
debido al traslape entre las ventanas. Modularemos esta situación mediante una variable a la que denominaremos Δ, la cual regula el desplazamiento que habrá entre una ventana y otra y que toma valores $1 \leq \Delta \leq (2w+1)$, con lo que si $\Delta = 1$ cada valor de la matriz de pesos $p(x,y)$ se calculará hasta $(2w+1) \times (2w+1)$ veces y si $\Delta = (2w+1)$, cada valor de la matriz de pesos se calculará sólo una vez. El proceso para calcular la imagen fusionada usando ventanas deslizantes y promediando los valores obtenidos en los procesos de optimización, lo denominaremos **CPV-P** y se muestra en el Algoritmo 2.

Algoritmo 2: **CPV-P**(I_1, I_2, H, ϵ, w, Δ)

1: Calcular la respuesta F_1 y F_2 aplicando (4.18) con H
2: Calcular $\Delta F = F_1 - F_2$
3: Hacer $x = 0$, $y = 0$, $p(x,y) \leftarrow 0$ y $N_w(x,y) \leftarrow 0 \; \forall (x,y) \in L$
4: **mientras** $x < N_r$ y $y < N_c$ **hacer**
5: Maximizar (5.6) para calcular p^*_{xy} con el valor ϵ
6: **para** $k = -w$ **to** w **hacer**
7: **para** $l = -w$ **to** w **hacer**
8: **si** $(x+k, y+k) \in L$ **entonces**
9: $p(x+k, y+l) \leftarrow p(x+k, y+l) + p^{\star}_{x,y}(k+w, l+w)$
10: $N_w(x+k, y+l) \leftarrow N_w(x+k, y+l) + 1$
11: **fin si**
12: **fin para**
13: **fin para**
14: $x = x + \Delta$
15: $y = y + \Delta$
16: **fin mientras**
17: Calcular p usando (5.9)
18: Binarizar p (5.4)
19: **devolver** p

Si comparamos los algoritmos 1 y 2, parecieran ser muy similares, sin embargo, en

el Algoritmo 1 cada valor $p(x, y)$ es guardado una sola vez y para ser calculado se toman en cuenta los valores que están a una distancia menor o igual a w en las direcciones x ó y, como se muestra en la Figura 5.7(a). Es decir, para calcular el valor de $p(x, y)$, marcado en la Figura con el punto rojo, se involucran $(2w + 1) \times (2w + 1)$ valores que son los que se muestran encerrados en la ventana de color rojo. En cambio, en el Algoritmo 2 en el que se promedian los resultados, el número de pixeles involucrados en el cálculo del valor de $p(x, y)$ es mayor, pues para calcular el valor $p(x, y)$ se toman en cuenta valores que se encuentran a una distancia menor o igual a $2w$ en las direcciones x ó y, como se muestra en la Figura 5.7(b). Es decir, para calcular el valor de $p(x, y)$ se involucran $(4w + 1) \times (4w + 1)$ valores, dicho de otra manera, todas las ventanas que encierren al punto (x, y) participarán en el cálculo del valor para dicho pixel, siendo la ventana centrada en el punto azul y enmarcada con el mismo color la primera que participa en la estimación y siendo la ventana centrada en el punto de color magenta la última ventana que participa en el proceso de estimación de $p(x, y)$.



(a) ***Valores involucrados en el cálculo de*** $p(x, y)$ ***usando el Algoritmo 1***

(b) ***Valores involucrados en el cálculo de*** $p(x, y)$ ***usando el Algoritmo 2***

Figura 5.7: Diferencia entre la cantidad de valores tomados en cuenta para calcular cada valor de $p(x, y)$ por el Algoritmo 1 y por el Algoritmo 2

Se puede observar en la Figura 5.7 que usando el Algoritmo 2 resolvemos parcialmente el problema de tener ventanas más grandes, así por ejemplo con $w = 4$, si usamos el Algoritmo 1 involucramos 81 valores en el cálculo del valores de $p(x, y)$ mientras que si usamos el Algoritmo 2 con el mismo valor $w = 4$ se involucran 289 valores lo cual es equiparable a tener ventanas de mayor tamaño. Más aún, si analizamos las veces que participa cada valor en la estimación de $p(x, y)$ en función de la distancia que guarda con las coordenadas (x, y), nos damos cuenta de que los valores que están más alejados de las coordenadas (x, y) participan menos veces en el cálculo que los valores que están más cerca de dichas coordenadas. Dicha situación se observa en la Figura 5.8 en la que se han calculado las veces que participa cada valor en la estimación de $p(x, y)$ si se tiene un $\Delta = 1$ y $w = 4$.

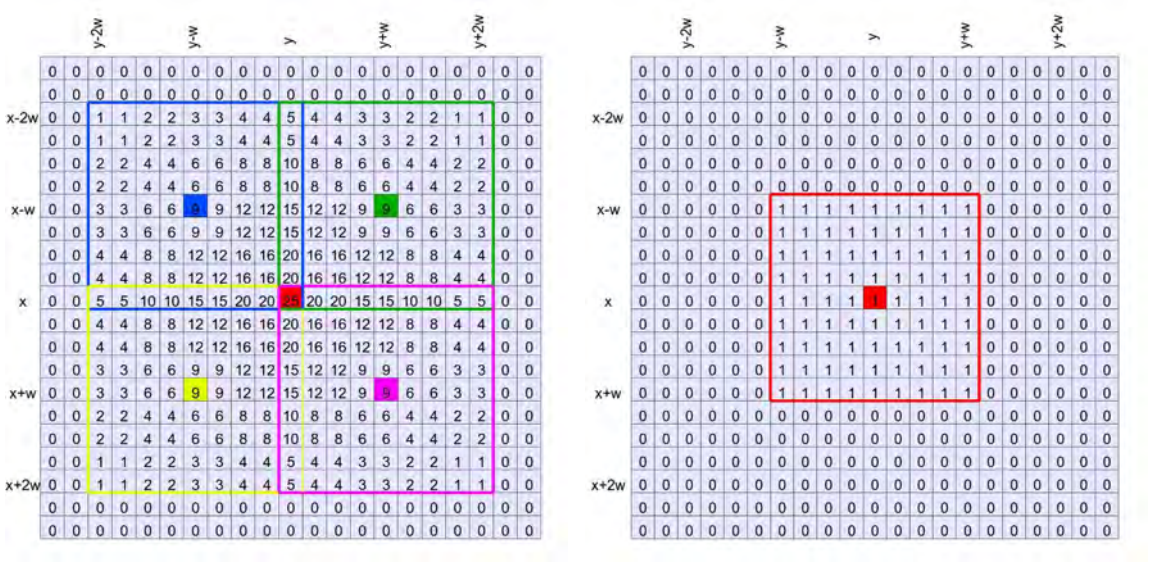
			y-2w			y-w				y				y+w			y+2w			
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
x-2w	0	0	1	2	3	4	5	6	7	8	9	8	7	6	5	4	3	2	1	0	0
	0	0	2	4	6	8	10	12	14	16	18	16	14	12	10	8	6	4	2	0	0
	0	0	3	6	9	12	15	18	21	24	27	24	21	18	15	12	9	6	3	0	0
	0	0	4	8	12	16	20	24	28	32	36	32	28	24	20	16	12	8	4	0	0
x-w	0	0	5	10	15	20	25	30	35	40	45	40	35	30	25	20	15	10	5	0	0
	0	0	6	12	18	24	30	36	42	48	54	48	42	36	30	24	18	12	6	0	0
	0	0	7	14	21	28	35	42	49	56	63	56	49	42	35	28	21	14	7	0	0
	0	0	8	16	24	32	40	48	56	64	72	64	56	48	40	32	24	16	8	0	0
x	0	0	9	18	27	36	45	54	63	72	81	72	63	54	45	36	27	18	9	0	0
	0	0	8	16	24	32	40	48	56	64	72	64	56	48	40	32	24	16	8	0	0
	0	0	7	14	21	28	35	42	49	56	63	56	49	42	35	28	21	14	7	0	0
	0	0	6	12	18	24	30	36	42	48	54	48	42	36	30	24	18	12	6	0	0
x+w	0	0	5	10	15	20	25	30	35	40	45	40	35	30	25	20	15	10	5	0	0
	0	0	4	8	12	16	20	24	28	32	36	32	28	24	20	16	12	8	4	0	0
	0	0	3	6	9	12	15	18	21	24	27	24	21	18	15	12	9	6	3	0	0
	0	0	2	4	6	8	10	12	14	16	18	16	14	12	10	8	6	4	2	0	0
x+2w	0	0	1	2	3	4	5	6	7	8	9	8	7	6	5	4	3	2	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figura 5.8: Número de veces que participa cada valor en el cálculo del valor de $p(x, y)$ con $\Delta = 1$, usando el Algoritmo 2

De la Figura 5.8 podemos destacar que el pixel ubicado en las coordenadas (x, y)

participa el mayor número de veces en la estimación de $p(x, y)$ mientras que los pixeles ubicados a una distancia $2w$ de las coordenadas x, y participan sólo una vez, es decir se le da más peso a los valores cercanos que a los que se encuentran lejos.

Como ya se mencionó, el valor de w y de Δ regulan el número de veces que se estimará el valor $p(x, y)$, así por ejemplo para $w = 4$ y $\Delta = 1$ el valor de $p(x, y)$ se estima 81 veces, como se ilustra en la Figura 5.8, mientras que para los valores $w = 4$ y $\Delta = 2$ el valor de $p(x, y)$ se estima 25 veces, como se ilustra en la Figura 5.9(a) y para $\Delta = (2w + 1)$ (independientemente del valor de w) el valor de $p(x, y)$ se estima únicamente 1 vez, como se ilustra en la Figura 5.9(b).

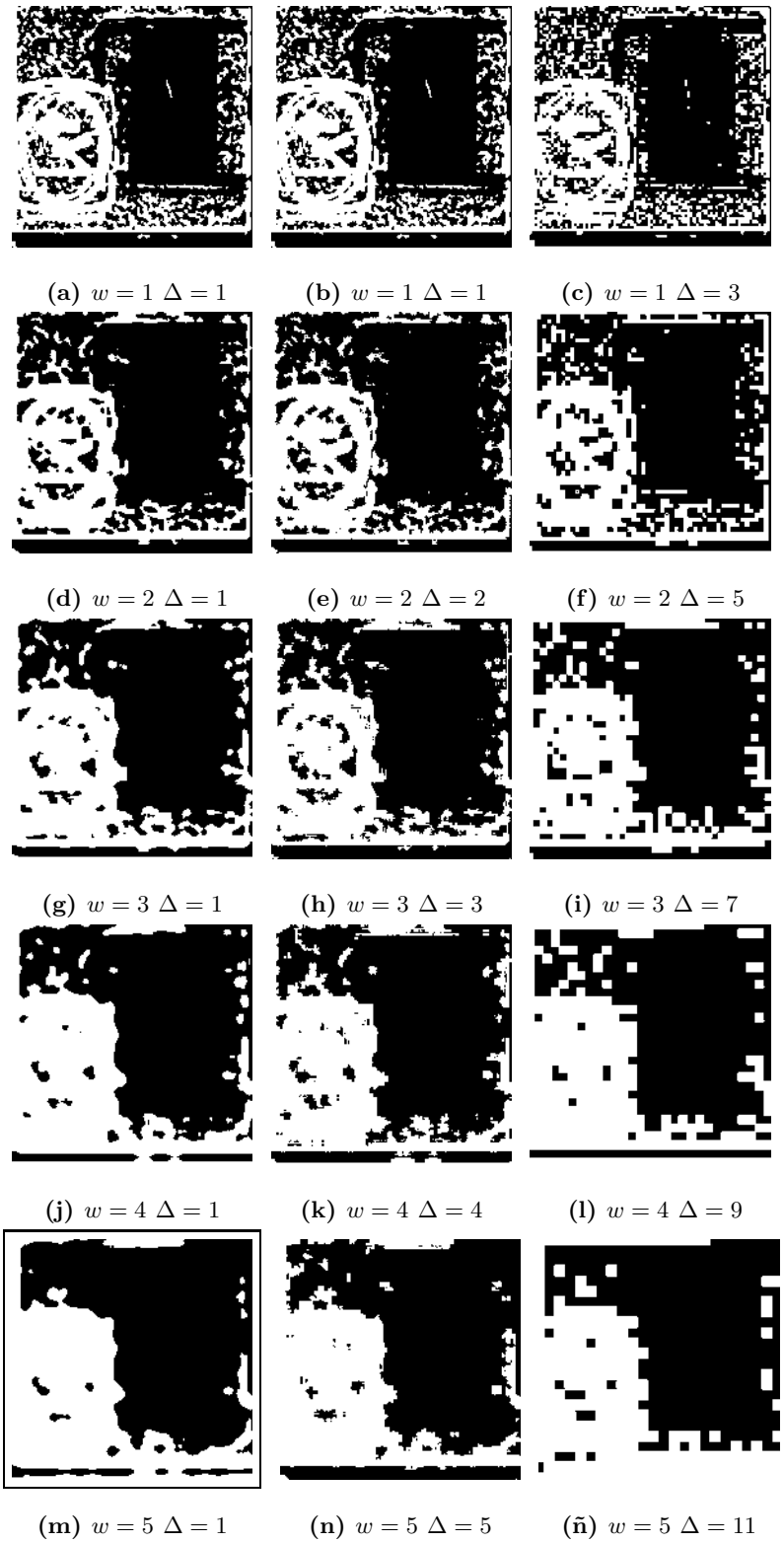


(a) ***Valores involucrados en el cálculo de*** $p(x, y)$ ***con el Algoritmo 2,*** $\Delta = 2$ ***y*** $w = 4$

(b) ***Valores involucrados en el cálculo de*** $p(x, y)$ ***con el Algoritmo 2,*** $\Delta = (2w + 1)$

Figura 5.9: Valores involucrados en el cálculo de $p(x, y)$.

En la Figura 5.10 se muestran los resultados de aplicar el Algoritmo 2 sobre las imágenes de los relojes, se puede observar que a medida que el valor de w aumenta y el valor de Δ disminuye, se logran resultados con mayor coherencia espacial y más aproximados a lo que se espera obtener. Puede observarse que con los valores $\Delta = 1$ y $w = 5$ el resultado se aproxima más a la matriz de pesos estimada manualmente la cual se muestra en la Figura 5.5(b).



(a) $w = 1\ \Delta = 1$ **(b)** $w = 1\ \Delta = 1$ **(c)** $w = 1\ \Delta = 3$

(d) $w = 2\ \Delta = 1$ **(e)** $w = 2\ \Delta = 2$ **(f)** $w = 2\ \Delta = 5$

(g) $w = 3\ \Delta = 1$ **(h)** $w = 3\ \Delta = 3$ **(i)** $w = 3\ \Delta = 7$

(j) $w = 4\ \Delta = 1$ **(k)** $w = 4\ \Delta = 4$ **(l)** $w = 4\ \Delta = 9$

(m) $w = 5\ \Delta = 1$ **(n)** $w = 5\ \Delta = 5$ **(ñ)** $w = 5\ \Delta = 11$

Figura 5.10: Resultados del Algoritmo 2 al variar los valores de w y Δ.

En la Tabla 5.4 se muestra un resumen de los resultados obtenidos con los algoritmos 1 y 2, en dicha tabla se puede apreciar que los resultados arrojados por el Algoritmo 2, superan a los resultados obtenidos con el Algoritmo 1. Se puede ver en la Tabla 5.4 que se obtienen los mejores resultados al aplicar Al Algoritmo 2 con valores de $w = 5$ y $\Delta = 1$.

w	**Alg.1** %	**Alg.2** %		
		$\Delta = (2w + 1)$	$\Delta = w$	$\Delta = 1$
0	**63.86**	**63.86**	**N/A**	**63.86**
1	71.64	71.64	**72.79**	**72.79**
2	76.60	77.40	77.20	**78.75**
3	80.67	80.44	80.15	**82.90**
4	83.56	84.91	83.11	**86.47**
5	85.97	88.42	84.71	**89.07**

Tabla 5.4: Porcentaje de acierto aplicando los algoritmos 1 y 2 a las imágenes de los relojes

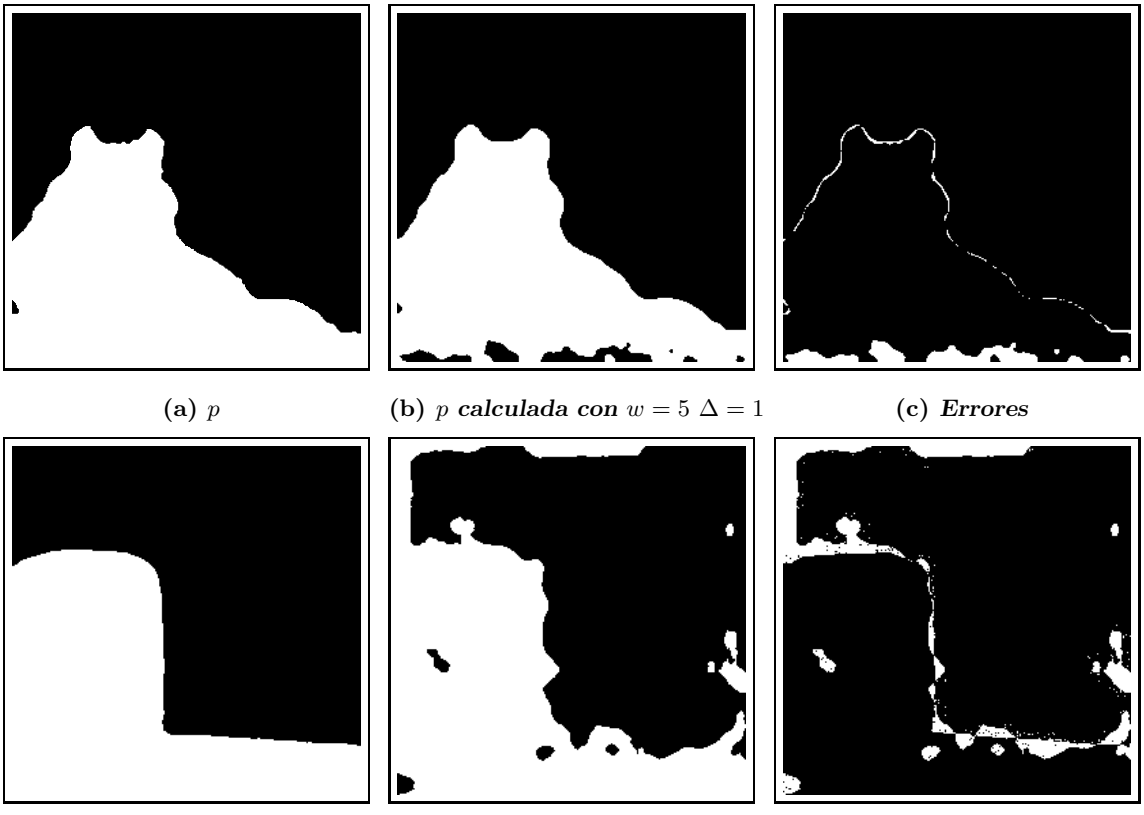
En la Tabla 5.5 se muestra el tiempo consumido por los algoritmos 1 y 2, a medida que varía el valor de w y Δ según el caso. Se puede observar que en prácticamente todos los casos el tiempo consumido por el Algoritmo 2 es menor o igual al tiempo requerido por el Algoritmo 1, pero los resultados mostrados en la Tabla 5.4 revelan que se desempeña mejor el Algoritmo 2, consumiendo el mismo o menor tiempo que el Algoritmo 1. Del análisis de las tablas 5.4 y 5.5 podemos también notar que el promediar los valores en el Algoritmo 2 permite lograr resultados muy similares a los del Algoritmo 1 pero en tiempos mucho menores, si se usa valores $\Delta = w$ o $\Delta = (2w + 1)$ lo que nos permite inferir que variando el valor de Δ podemos obtener resultados muy buenos en tiempos menores.

w	**Tiempo en Seg. Alg.1**	**Tiempo en Seg. Alg.2**		
		$\Delta = (2w + 1)$	$\Delta = w$	$\Delta = 1$
0	0.314	0.293	N/A	0.293
1	5.649	0.728	4.985	4.93
2	169.01	7.143	45.985	126.38
3	1278.4	34.28	209.71	1178.6
4	8902.5	102.2	637.70	8669.4
5	31225	326.4	1544.1	30870

Tabla 5.5: Tiempo consumido por los algoritmos 1 y 2 en las imágenes de los relojes

Podemos enfatizar que para este ejemplo, usando el Algoritmo 2 con $w = 5$ y $\Delta = 11$ se logra un porcentaje de acierto de 88.42 % en 326.467 segundos mientras que el Algoritmo 1 logra como máximo un porcentaje de acierto de 85.97 % con $w = 5$ pero tarda 31225.67 segundos, por lo que podemos decir que será mucho mejor calcular p como el promedio de todas las soluciones encontradas, tal como se indica en el Algoritmo 2.

En la figura 5.11 se muestra el resultado de aplicar el Algoritmo 2 a las imágenes de los lobos y de los relojes. En las Figuras 5.11(a) y 5.11(d), se muestran las matrices p para el ejemplo de los lobos y de los relojes respectivamente. En las Figuras 5.11(b) y 5.11(e), se muestra el mejor resultado obtenido usando el Algoritmo 2 para cada ejemplo. Finalmente en las Figuras 5.11(c) y 5.11(f) se muestra la el error en la matriz de pesos calculada.



(a) p **(b)** p ***calculada con*** $w = 5$ $\Delta = 1$ **(c)** ***Errores***

(d) p **(e)** p ***calculada con*** $w = 6$ $\Delta = 1$**.** **(f)** ***Errores***

Figura 5.11: Resultado de aplicar el Algoritmo 2 sobre las imágenes de los lobos y los relojes.

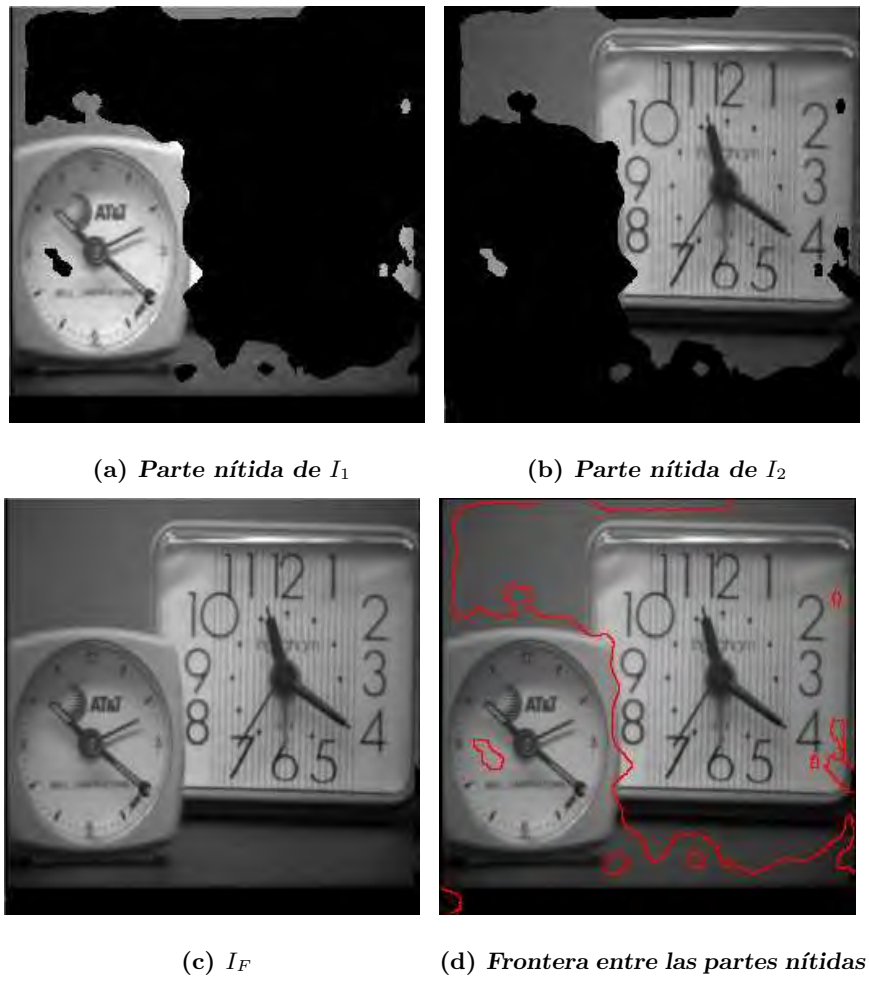
Se puede notar en la Figura 5.11 que la matriz de pesos calculada en ambos casos se acerca mucho a lo esperado, pero aún existen algunas regiones en negro dentro de las regiones

que deben ser blancas y viceversa. En la Tabla 5.6 podemos observar que el porcentaje de acierto es muy bueno pero el tiempo requerido es muy alto por lo que no resulta viable la aplicación del algoritmo con valores de $w \geq 5$ a menos que se use un valor de $\Delta \geq w$. En

Ejemplo	**Parámetros**	**Tiempo en Horas**	**% de acierto**
Lobos	$w = 5$, $\Delta = 1$ y $\epsilon = 0.01$	23.254	97.5986
Relojes	$w = 6$, $\Delta = 1$ y $\epsilon = 0.01$	24.368	92.2653

Tabla 5.6: Tiempo y porcentaje de acierto aplicando el Algoritmo 2, sobre las imágenes de los lobos y de los relojes

la Figura 5.12 se muestra el resultado de aplicar el Algoritmo 2 sobre las imágenes de los relojes.



(a) ***Parte nítida de*** I_1 **(b)** ***Parte nítida de*** I_2

(c) I_F **(d)** ***Frontera entre las partes nítidas***

Figura 5.12: Imagen fusionada calculada con la matriz p resultado del Algoritmo 2 usando como parámetros $w = 6$, $\Delta = 1$ y $\epsilon = 0.01$.

Se puede observar en la Figura 5.12(d), que los errores se dan en zonas con poca textura, lo que dificulta detectar, en cual imagen es más nítida dicha región, por lo que la solución puede ser utilizar ventanas de mayor tamaño, sin embargo también se ha comprobado que a medida que el tamaño de la ventana se incrementa, el tiempo necesario para obtener la solución crece demasiado. Es por esa razón que es necesario establecer técnicas que nos permitan abordar el problema en un contexto que requiera menos recursos de cómputo.

5.3. Submuestreo de imágenes

Las imágenes digitales son una representación abstracta de una escena, es decir, las imágenes digitales son creadas en base a muestras de una función continua. De acuerdo a lo planteado en [Panusopone et al., 1995], el muestreo es el proceso de convertir una función continua $x_c(t)$ en una secuencia discreta de datos llamadas muestras y denotada como $x_d[n]$. El submuestreo y escalamiento espacial de las imágenes permiten abordar el problema en un contexto en el que se requieren menos recursos. Basados en lo que se establece en [Panusopone et al., 1995], podemos definir el submuestreo como el proceso de muestreo de una secuencia discreta para producir una nueva secuencia de proporciones y/o dimensiones diferentes a la secuencia original.

Lo que proponemos es hacer un submuestreo de las imágenes para calcular la matriz de pesos en una escala menor y después escalar la solución para aplicarla con las imágenes originales. Las transformaciones afines son definidas en [Ghali, 2008] como aquellas que preservan el paralelismo y que mapean puntos afines a otros puntos afines. El escalamiento es una transformación afín, lo que implica que las imágenes que sufran dicha transformación mantendrán la forma, sufriendo variación únicamente en el tamaño. Para el submuestreo usaremos la ecuación (5.10) con la que en base a una imagen I_k y un valor de s calculamos la imagen I_k^s, donde si $s = 2$ la imagen I_m se reduce a un tamaño $\frac{N_r}{2} \times \frac{N_c}{2}$.

$$\begin{aligned} I_k^s(x,y) &= I_k(sx,sy) * G(\sigma,x,y) \\ 0 \leq x \leq \tfrac{N_r}{s}, &0 \leq y \leq \tfrac{N_c}{s} \end{aligned} \tag{5.10}$$

Como el proceso de submuestreo implica pérdida de información, antes de hacer el submuestreo se suele aplicar una convolución de las imágenes originales con un kernel Gaussiano,

con el fin de que todos los valores tomen parte en la composición de la nueva imagen. El problema de aplicar la convolución Gaussiana es que el cálculo de la convolución es costoso y más aún si el valor de desviación estándar σ es grande. Otra alternativa es aplicar una convolución con una caja, lo que implica que cada pixel es calculado como la suma o el promedio de los pixeles que se encuentran alrededor de la posición correspondiente en la imagen original. La convolución con una caja ofrece la ventaja de que puede ser calculada de manera muy eficiente usando imágenes incrementales, las cuales son definidas en [Viola y Jones, 2001]. El uso de las imágenes incrementales permite calcular la convolución de una imagen con una caja de manera muy eficiente sin importar el tamaño de la caja. Sin embargo, aunque la convolución con una caja es menos costosa, aplicar un kernel de convolución Gaussiano es más común, debido a que cada pixel de la imagen original es tomado en cuenta en mayor o menor grado, en función de la distancia que guarda con el pixel del centro de la Gaussiana.

Para el escalamiento usaremos la ecuación (5.11) con valores $s \leq 1$ y si $s = .5$ la matriz p^s será ampliada a un tamaño $2N_r \times 2N_c$. Dado que para el escalamiento los valores de s son fraccionarios, se aplicará interpolación bilineal para el cálculo de los valores de $p_s(x, y)$.

$$\begin{gathered} p_s(x,y) = Bilineal(p^s(sx, sy)) \\ 0 \leq x \leq N_r, 0 \leq y \leq N_c \end{gathered} \tag{5.11}$$

En la Figura 5.13 se muestra el proceso para el cálculo de la matriz de pesos p con imágenes que han sido submuestreadas. El primer paso es aplicar un suavizado y submuestreo a las imágenes originales para llevarlas a un tamaño menor en una proporción $\frac{1}{s}$, es decir, llevarlas a un tamaño $\frac{N_c}{s} \times \frac{N_c}{s}$, posteriormente se calcula la matriz de pesos p^s con las imágenes escaladas I_1^s e I_2^s, usando el Algoritmo 2, finalmente se escala la matriz de pesos p^s a una proporción de s veces sus dimensiones, para tener la matriz de pesos del mismo tamaño que las imágenes originales. Se puede notar en el ejemplo mostrado en la Figura 5.13 que el resultado no es el exacto, sin embargo es un resultado muy bueno con un costo en tiempo muy bajo.

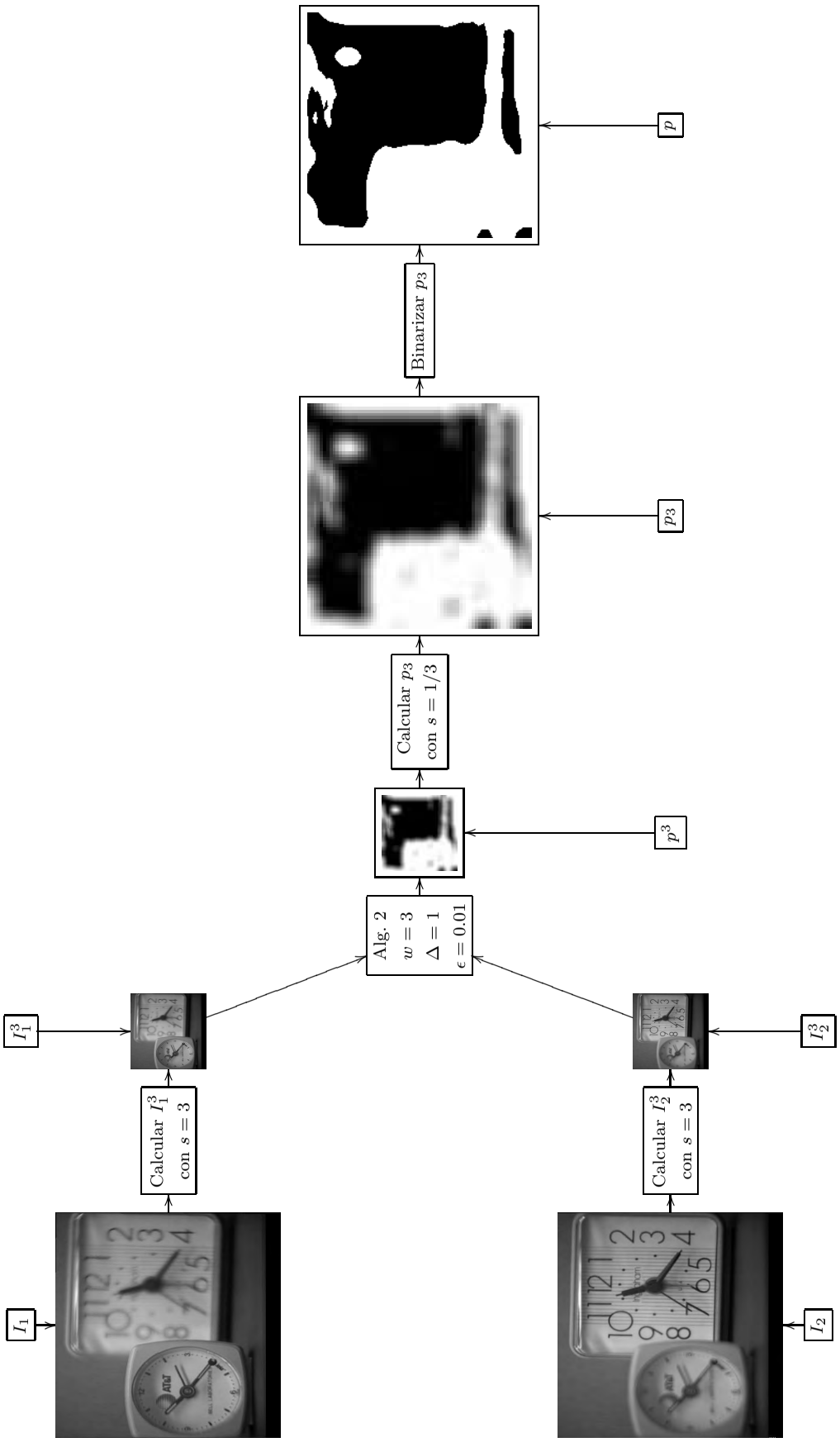


Figura 5.13: Esquema del proceso a seguir para resolver el problema en una escala menor.

En la Tabla 5.7 se muestra una comparación entre usar el Algoritmo 2 con y sin escalamiento.

Alg. 2 (CPV-P)	**Parámetros**	**Tiempo en Seg.**	**% de acierto**
Sin escalamiento	$w = 3$, $\Delta = 1$ y $\epsilon = 0.01$	1178.62	82.90
Con escalamiento	$w = 3$, $\Delta = 1$ y $\epsilon = 0.01$	114.753	82.54

Tabla 5.7: Porcentaje de acierto y tiempo consumido al aplicar el Algoritmo 2, sobre las imágenes de los relojes, con y sin escalamiento

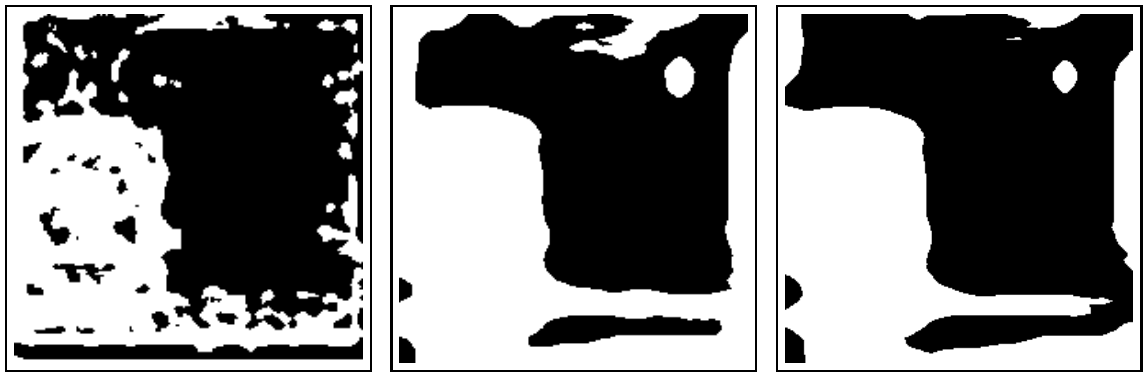
En la Tabla 5.7, se puede ver que el porcentaje de acierto es prácticamente el mismo, sin embargo el tiempo consumido usando el escalamiento es más de 10 veces menor que el tiempo requerido para el cálculo de la función p en el tamaño de las imágenes originales. El resultado obtenido usando escalamiento no es exacto pero nos puede servir de base para calcular la matriz de pesos p. Sin embargo el método Simplex tiene la desventaja de que no permite dar valores iniciales a las variables, pero, dada la naturaleza de nuestro problema, podemos aplicar una heurística. Sabemos que si la imagen I_1 es más nítida que I_2 en la ventana W, el resultado de la optimización serán valores cercanos a 1 y si la nitidez de I_2 es mayor que la de I_1 en la región W el proceso arrojará valores cercanos a 0. Dado que en el método Simplex todas las variables comienzan con valor de 0, si la nitidez de I_2 es mayor que la de I_1, habrá que hacer muy poco, pues la solución estará prácticamente dada, pero en caso de que I_1 sea más nítida que I_2, habrá que hacer muchas iteraciones para llevar las variables a valores cercanos a 1. Sin embargo si tenemos información previa, podemos intercambiar el proceso y en lugar de hacer una maximización, podemos hacer una minimización con lo que en lugar de tener que llevar los valores de las variables a valores cercanos a 1, tendríamos que llevarlas a valores cercanos a 0 y dado que las variables inician con valor de 0, tendremos valores iniciales muy cercana a la solución, lo que se verá reflejado en una disminución en el tiempo necesario para el cálculo de la solución.

En nuestro problema no tenemos información a priori sobre qué imagen es más nítida en la región W, pero tenemos algunas opciones para adquirir dicha información. Una opción es calcular la matriz de pesos en una escala mucho menor y usar dicha información para tomar la decisión de si hacer un proceso de maximización o minimización. Una se-

gunda opción es, dado que el desplazamiento entre una ventana y la siguiente es pequeño, podemos usar información del resultado de la última ventana procesada, para decidir si maximizamos o minimizamos, es decir, si la última ventana calculada arrojó valores cercanos a 0, aplicaremos un proceso de maximización y si el resultado de la última ventana fueron valores cercanos a 1, realizaremos un proceso de minimización. El hacer el cambio entre un proceso de maximización o minimización es muy fácil, pues lo único que tenemos que hacer es cambiar el signo de los coeficientes de las variables. Cabe hacer mención que si se cambió el signo de las variables, se tiene también que cambiar la solución, es decir, una vez que ha terminado el proceso de optimización aplicaremos la siguiente regla $p_{x,y}(k,l) = 1 - p_{x,y}(k,l) \;\; \forall < k,l > \in W$.

El aplicar la heurística planteada, pareciera no tener la menor importancia, sin embargo, permite una reducción en el tiempo consumido por el algoritmo. Por ejemplo para el caso de los relojes el tiempo se reduce más del 50 % al aplicar la heurística. Dada la naturaleza de nuestro problema, para una ventana W, existen distintas soluciones al proceso de optimización, lo que complica aún más el garantizar la coherencia espacial. El aplicar la heurística nos permite mejorar un poco la coherencia espacial, dado que se toma en cuenta la información de la última ventana para optimizar la siguiente ventana.

Se ha probado el desempeño del Algoritmo 2 con valores de $w = 3$, $\Delta = 1$ y $\epsilon = 0.01$, en tres condiciones diferentes y el resultado se muestra en la Figura 5.14.



(a) ***Algoritmo 2 sin escalamiento ni heurísitica*** **(b)** ***Algoritmo 2 con escalamiento.*** **(c)** ***Algoritmo 2 con escalamiento y heurísitica.***

Figura 5.14: Matriz p obtenida al aplicar el Algoritmo CPV-P a las imágenes de los relojes, sin escalamiento, con escalamiento y con escalamiento y heurística.

En la Figura 5.14(a) se muestra el resultado de aplicar el Algoritmo 2 a las imágenes en el tamaño original, en la Figura 5.14(b), se muestra el resultado de aplicar el Algoritmo 2 pero escalando las imágenes a $\frac{1}{3}$ de las dimensiones originales y finalmente en la Figura 5.14(c) se muestra el resultado de aplicar el Algoritmo 2 escalando las imágenes a 1/3 de las dimensiones originales y aplicando la heurística que permite cambiar el proceso de optimización.

En la Tabla 5.8 se muestra un resumen de los resultados de aplicar el Algoritmo 2 sobre las imágenes de los relojes con condiciones diferentes. Se puede observar que el resultado de aplicar el Algoritmo 2 a las imágenes originales o escaladas no afecta en gran medida el porcentaje de acierto de la matriz de pesos, sin embargo, el tiempo se reduce aproximadamente 10 veces, mientras que si aplicamos el escalamiento y la heurística el consumo de tiempo se reduce aún más.

Alg. 2 (CPV-P)	**Tiempo en Seg.**	**% de acierto**
Sin heurística ni escalamiento	1178.6	82.90
Con escalamiento	114.75	82.54
Con escalamiento y heurística	60.825	85.49

Tabla 5.8: Porcentaje de acierto y tiempo consumido al usar el Algoritmo 2, sobre las imágenes de los relojes con $w = 3$, $\Delta = 1$ y $\epsilon = 0.01$ como parámetros.

5.4. Fusión de imágenes usando submuestreo y heurística

Reuniendo los elementos que se han presentado a lo largo de este documento, podemos plantear el Algoritmo 3, usando el submuestreo y una heurística al que llamaremos **CPV-PSH**. En el algoritmo CPV-PSH se usa submuestreo para procesar las imágenes en tamaños menores con el fin de atacar el problema en un contexto en el que hay un menor número de variables. Para dicho algoritmo también se plantea el uso de una heurística que permite identificar, en qué casos es más conveniente maximizar y en cuales es menos costoso aplicar el proceso de minimización. La heurística consiste es obtener el promedio μ de los valores de la última ventana $p_{x,y}$ calculada y si $\mu > 0.5$ significa que lo más probable es que sea más rápido hacer un proceso de minimización en lugar de un proceso de maximización.

Algoritmo 3: **CPV-PSH**(I_1, I_2, H, ϵ, w, Δ, s_{max})

para $s = s_{max}$ **to** 1 **hacer**
  Calcular I_1^s e I_2^s en base a I_1 e I_2 aplicando (5.10).
  Calcular F_1^s y F_2^s con I_1^s, I_2^s y H
  Calcular $\Delta F = F_1^s - F_2^s$
  Hacer $x = 0$, $y = 0$, $p^s(x, y) \leftarrow 0$, $N_w^s(x, y) \leftarrow 0$ $\forall (x, y) \in L$ y $\mu = 0$
  mientras $x < N_r$ y $y < N_c$ **hacer**
    si $\mu > 0.5$ **entonces**
      Calcular $p^*_{x,y}$ que minimiza (5.6) usando ϵ.
    si no
      Calcular $p^*_{x,y}$ que maximiza (5.6) usando ϵ.
    fin si
    para $k = -w, l = -w$ **to** $k = w, l = w$ **hacer**
      si $(x + k, y + k) \in L$ **entonces**
        si $\mu > 0.5$ **entonces**
          $p^s(x + k, y + l) \leftarrow p^s(x + k, y + l) + (1 - p^\star_{x,y}(k + w, l + w))$
        si no
          $p^s(x + k, y + l) \leftarrow p^s(x + k, y + l) + p^\star_{x,y}(k + w, l + w)$
        fin si
        $N_w(x + k, y + l) \leftarrow N_w(x + k, y + l) + 1$
      fin si
    fin para
    $\mu = \frac{\sum_{k=0}^{2w} \sum_{l=0}^{2w} p^\star_{x,y}(k,l)}{(2w+1)^2}$
    $x = x + \Delta$ y $y = y + \Delta$
  fin mientras
  $p^s(x, y) = \frac{p^s(x,y)}{N_w(x,y)}$ $\forall < x, y > \in L$
  Escalar p^s a s veces sus dimensiones con (5.11) y sumarlo a p
fin para
$p = p * \frac{1}{s_{max}}$
Binarizar p usando (5.4) y calcular I_F usando (4.19)
devolver I_F

Dado el Algoritmo 3 lo que resta es establecer los valores adecuados para w, Δ, ϵ y s_{max}. Cabe mencionar que al variar cualquiera de estos parámetros tendremos una variación en la coherencia espacial lograda. En base a los resultados obtenidos en los experimentos se ha llegado a la conclusión de que los valores adecuados para w y ϵ son: $w \geq 4$ y $\epsilon = 0.01$. Fijaremos los valores de w, ϵ y s_max en $w = 4$, $\epsilon = 0.01$ y $s_{max} = 6$ para aplicar el Algoritmo 3 a las imágenes de los relojes, variando el valor de Δ, con el fin de analizar las repercusiones que tiene dicha variación, sobre el resultado y el tiempo consumido por el algoritmo. En la Tabla 5.9 se muestra un resumen de los resultados obtenidos al aplicar el Algoritmo 3, con distintos valores de Δ. De dicha tabla, podemos destacar que con un valor $\Delta = 3$ (el cual es $\frac{1}{3}$ del tamaño de la ventana), se obtiene la mejor solución y además se puede ver que con $\Delta = 2$ y $\Delta = 1$, el tiempo aumenta mucho, mientras que el porcentaje de acierto, permanece prácticamente constante.

Δ	**Tiempo en Seg.**	**% de acierto**
9	60.666	85.34
8	72.793	86.27
7	94.269	89.1
6	124.089	93.38
5	175.042	92.07
4	260.524	91.46
3	458.985	94.75
2	984.126	94.48
1	3970.53	94.55

Tabla 5.9: Resultados del uso del Algoritmo 3, sobre las imágenes de los relojes, con valores de $w = 4$, $\epsilon = 0.01$ y $s_{max} = 6$.

En base a la información de dicha tabla y en base a distintos experimentos que se han realizado, podemos concluir que los valores de los parámetros que mejor resultado ofrecen, son $w = 4, \epsilon = 0.01$ y $\Delta = 3$. Cabe hacer mención de que incrementando el valor de w y/o disminuyendo el valor de Δ, en algunos casos se lograron mejores resultados, sin embargo el incremento en el tiempo es considerablemente grande, sin embargo si se desea mayor precisión en la solución, se puede aplicar un $\Delta = 1$ o ventanas con $w > 4$. En la Figura 5.15, se muestra un ejemplo de la aplicación del Algoritmo 3, sobre las imágenes de los relojes, usando como parámetros $w = 4, \epsilon = 0.01, \Delta = 3$ y $s_{max} = 6$.

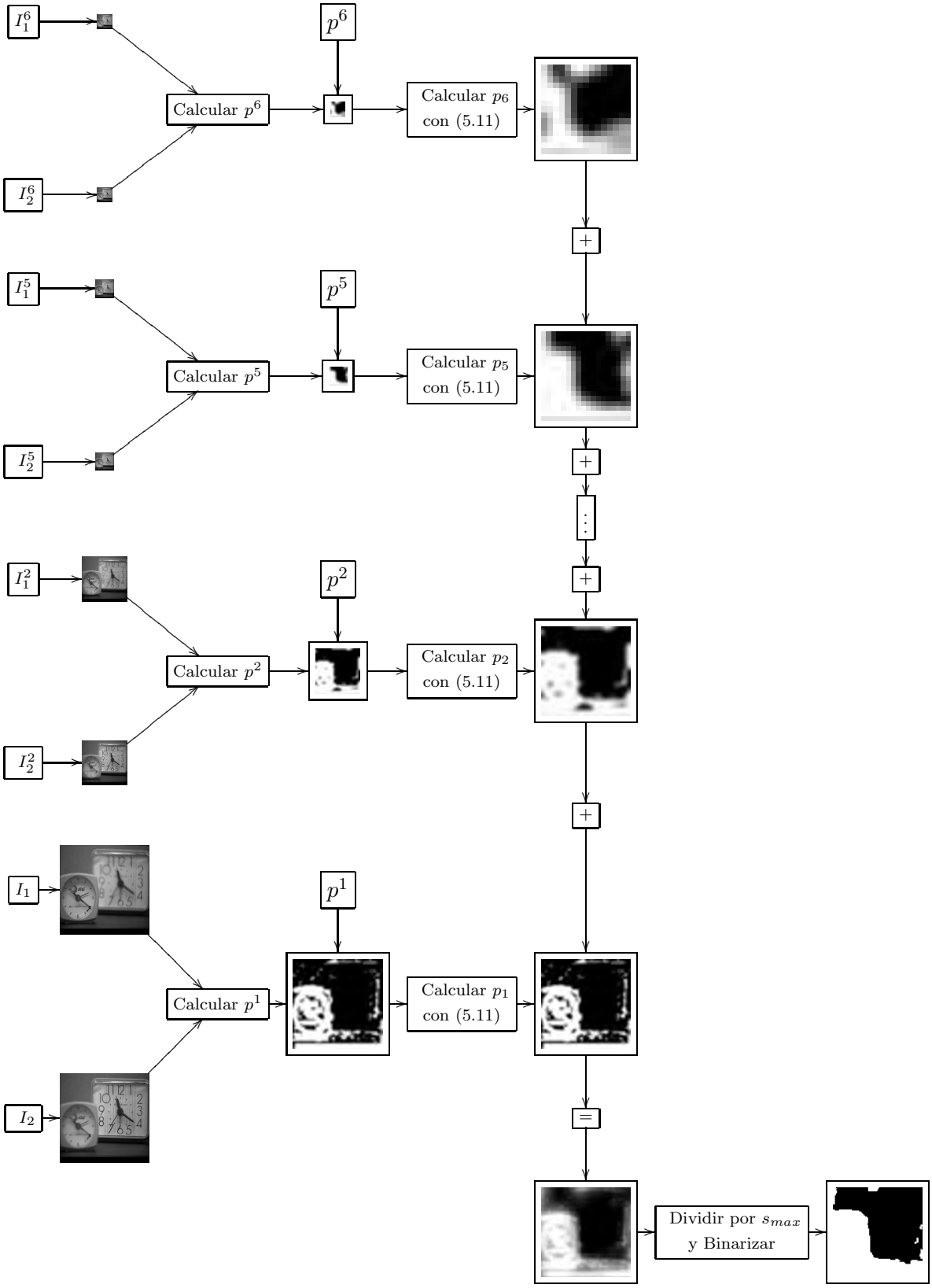


Figura 5.15: Esquema del proceso de fusión usando el **Alg. CPV-PSH**.

Se puede observar en la Figura 5.15 que p^6 no es muy exacta, sin embargo nos da una aproximación de la matriz de pesos a un costo muy bajo. También se puede observar que a medida que s disminuye, las matrices de pesos p^s estimadas, ofrecen cada vez más detalle y que al promediar las s_{max}, matrices de pesos estimadas, encontramos una matriz de pesos muy cercana a la exacta.

5.5. Conclusiones del capítulo

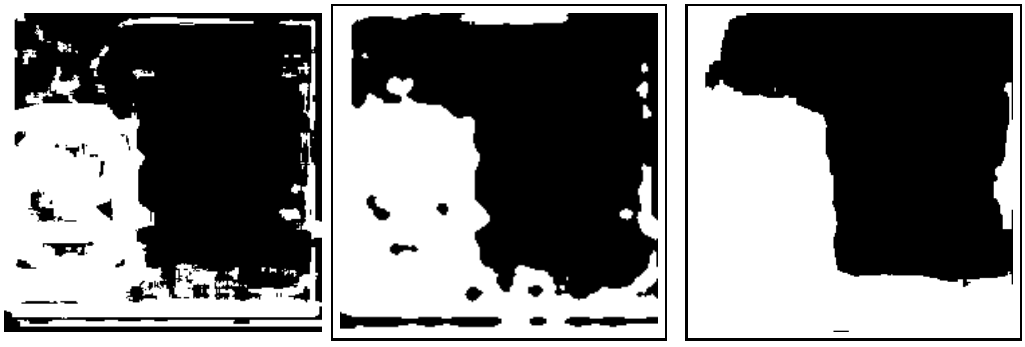
En el desarrollo de este capítulo se ha explicado el proceso llevado a cabo para plantear el Algoritmo 3, en el que se hace uso de herramientas como ventanas deslizantes, el submuestreo y sobremuestreo, para calcular la matriz de pesos p y con ello lograr resolver el problema de fusión de imágenes multi-foco garantizando que la imagen fusionada I_F tenga una nitidez mayor a la nitidez de I_1 e I_2. Esto garantiza además la coherencia espacial y reduciendo el tiempo necesario para el cálculo de la solución.

Si hacemos una comparación del resultado obtenido con cada uno de los algoritmos en el ejemplo de los relojes podemos mostrar los resultados de la Tabla 5.10, en donde se observa que los de ventanas deslizantes **CPV (Alg. 1)** y el de promedio de las soluciones de ventanas deslizantes **CPV-P (Alg. 2)** son superados por el Algoritmo 3 **(CPV-PSH)**, en el porcentaje de acierto de los resultados. Además el tiempo consumido por el algoritmo CPV-PSH es aproximadamente 70 veces menor que el tiempo consumido por los otros dos algoritmos, para este ejemplo.

Algoritmo	**% de acierto**	**tiempo en seg**	**Parámetros**
CPV	85.97	31225.6	$w = 5$ y $\epsilon = 0.01$
CPV-P	89.07	30870.7	$w = 5$ y $\epsilon = 0.01$
CPV-PSH	94.28	446.443	$w = 4$, $\epsilon = 0.01$, $\Delta = 3$ y $s_{max} = 5$

Tabla 5.10: Comparación entre los mejores resultados obtenidos usando cada algoritmo

En la Figura 5.16 se muestran las matrices de pesos calculadas con cada uno de los algoritmos, se puede observar en dicha figura que en el resultado arrojado por el Algoritmo 3 la matriz de pesos p aparece mejor definida, con menos inconsistencias y que se aproxima mucho a la matriz estimada de forma manual.



(a) ***Matriz*** p ***con CPV*** **(b)** ***Matriz*** p ***con CPV-P*** **(c)** ***Matriz*** p ***con CPV-PSH***

Figura 5.16: Mejor resultado de los Algoritmo 1, 2 y 3.

Existen enfoques que sostienen que la eficiencia de los algoritmos de fusión de imágenes multi-foco se debe medir calificando su desempeño sólo en áreas en las que con mayor textura y dejando fuera de la medición aquellas áreas "sin cuidado"que son aquellas con poca textura, por ejemplo paredes o algunos fondos de escena que poseen poca información y que para efectos visuales resulta irrelevante de cual imagen se tomen. Basados en esta idea, en la Figura 5.17 presentamos un mapa de decisión en el que se indica que la región marcada con blanco debe ser tomada de la I_1, porque es es el reloj del enfrente, la parte marcada con negro debe ser tomada de I_2 por ser donde se está el reloj de atrás y la parte marcada con gris no importa de que imagen se tome.

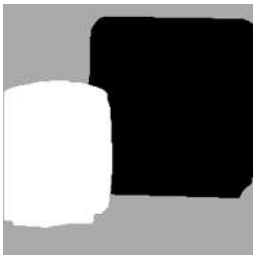


Figura 5.17: Mapa de decisión con regiones "sin cuidado".

Bajo estas condiciones tenemos que con CPV se logra un 94.1 % de acierto, con CPV-P 96.5 % de acierto y con CPV-PSH se logra el 97.3 % de acierto. De los experimentos realizados podemos concluir que los resultados obtenidos con el Algoritmo 3 fueron mejores a los resultados de los otros dos algoritmos planteados en este capítulo, consumiendo menor tiempo y logrando un porcentaje de acierto mayor.

Capítulo 6

Resultados

En el este capítulo se presentan los resultados obtenidos de la aplicación del algoritmo de fusión de imágenes 3 **(CPV-PSH)**, sobre algunos pares de imágenes multi-foco, cabe mencionar que se ha aplicado el método SSD-ARC para alinear las imágenes antes de aplicar el algoritmo de fusión.

6.1. Resultado de aplicar registro de imágenes

Para mostrar el impacto que tiene el registrar las imágenes antes de aplicar el proceso de fusión, se ha aplicado el Algoritmo 3 sobre las imágenes multi-foco de la escultura, las cuales se muestran en las Figuras 6.1(a) y 6.1(a) con lo que se logra obtener la matriz de pesos mostrada en la Figura 6.2(a) y la imagen fusionada mostrada en la Figura 6.11(g).

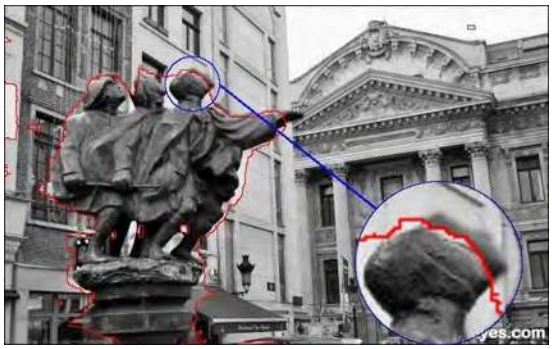
(a) I_1 **(b)** I_2

Figura 6.1: Imágenes de fusión multi-foco de la escultura y el edificio en el fondo

En la Figura 6.2(c) se ha hecho un acercamiento a uno de los artefactos que aparecen más notables en la imagen fusionada, se puede notar que dicha situación se ha generado debido a un desplazamiento espacial entre una imagen multi-foco y la otra.

(a) ***Matriz*** p**.**

(b) I_F**.**



(c) ***Artefacto que aparecen en la*** I_F

Figura 6.2: Problema de registro en imágenes multi-foco

El problema de registro impide que el Algoritmo 3 funcione de manera adecuada. Sin embargo si sustituimos I_2 por la imagen registrada I_M y aplicamos el proceso de fusión se logra el resultado mostrado en la Figura 6.3(b).

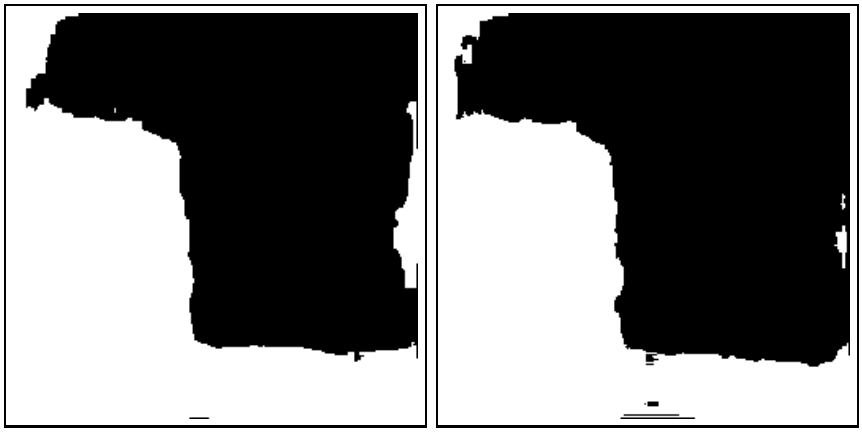
(a) ***Matriz*** p ***obtenida.***

(b) ***Imagen fusionada***

Figura 6.3: Matriz p e I_F obtenidas al aplicar el Algoritmo 3 con imágenes registradas

Nótese en la imagen 6.3(b) que al aplicar el registro de imágenes, se ha logrado evitar que en la imagen fusionada aparezca el artefacto que aparece en la Figura 6.2(b), en la que aparece repetido el límite de la cabeza de uno de los personajes de la escultura. Además se puede notar que el límite de la escultura se logra aproximar de mejor manera.

En la Figura 6.4 se muestra las matrices de pesos calculadas usando el Algoritmo CPV-PSH para el ejemplo de los relojes. En la imagen de la Figura 6.4(a), se muestra el resultado aplicando la fusión sin el registro y en la Figura 6.4(b) se muestra el resultado de la fusión si se aplica primero el registro y después la fusión de las imágenes.



(a) ***Con imágenes sin registrar*** **(b)** ***Con imágenes registradas***

Figura 6.4: Matriz p calculada con el Algoritmo 3, sin aplicar registro y aplicando el método SSD-ARC para registrar las imágenes.

Si comparamos las matrices p mostradas en la Figura 6.4 con la matriz p calculada manualmente, podemos ver que si se aplica el proceso con imágenes sin registrar se alcanza un porcentaje de acierto del 94.28 % mientras que si aplica la fusión a las imágenes registradas el porcentaje de acierto es del 96 %. Si hacemos la comparación sin tomando en cuenta el mapa de decisión mostrado en la Figura 5.17, sin tomar en cuenta las regiones marcadas en gris, se logra un porcentaje de acierto de 97.3 % de acierto con imágenes sin registrar y 99 % si la fusión se hace con imágenes registradas. Lo que hace notar la importancia de registrar las imágenes antes de aplicar el proceso de registro pero que también deja ver que el Algoritmo 3 funciona de manera aceptable con imágenes sin registrar.

6.2. Resultados de aplicar el Algoritmo CPV-PSH

El proceso que se aplica para fusionar las imágenes multi-foco mostradas en éste capítulo es el mostrado en la figura 6.5.

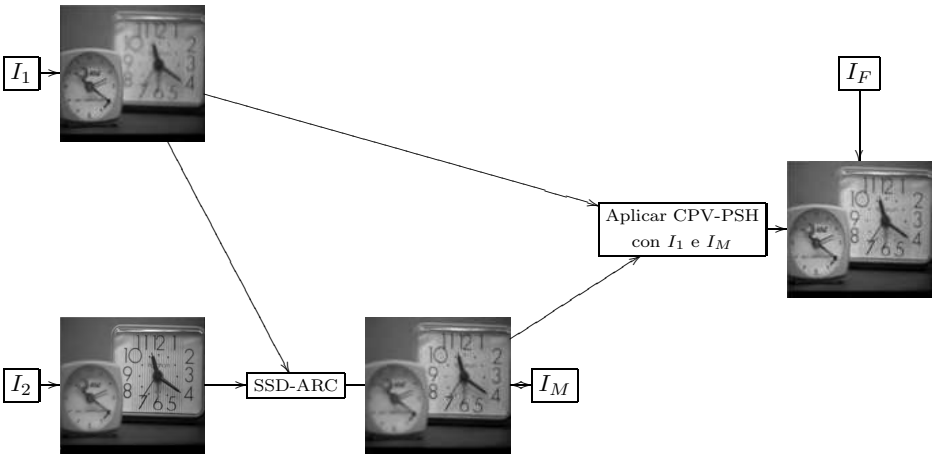
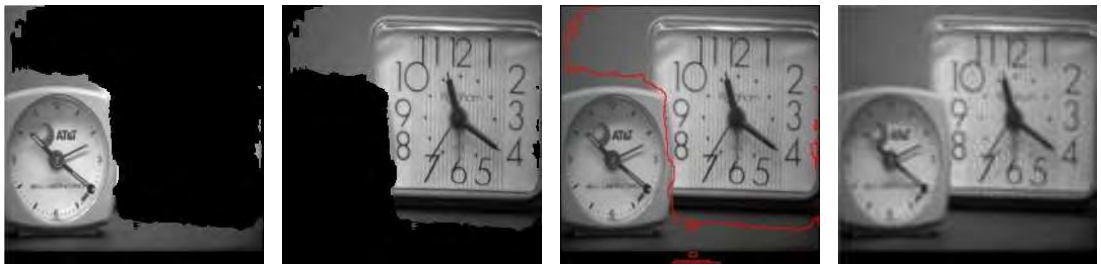


Figura 6.5: Esquema del proceso para fusionar dos imágenes multi-foco.

Como primer ejemplo mostramos la fusión de las imágenes mostradas en las Figuras 6.6(a) y 6.6(b), que son imágenes que ya se habían tratado previamente.

(a) I_1 **(b)** I_2 **(c)** I_M **(d)** ***Matriz*** p

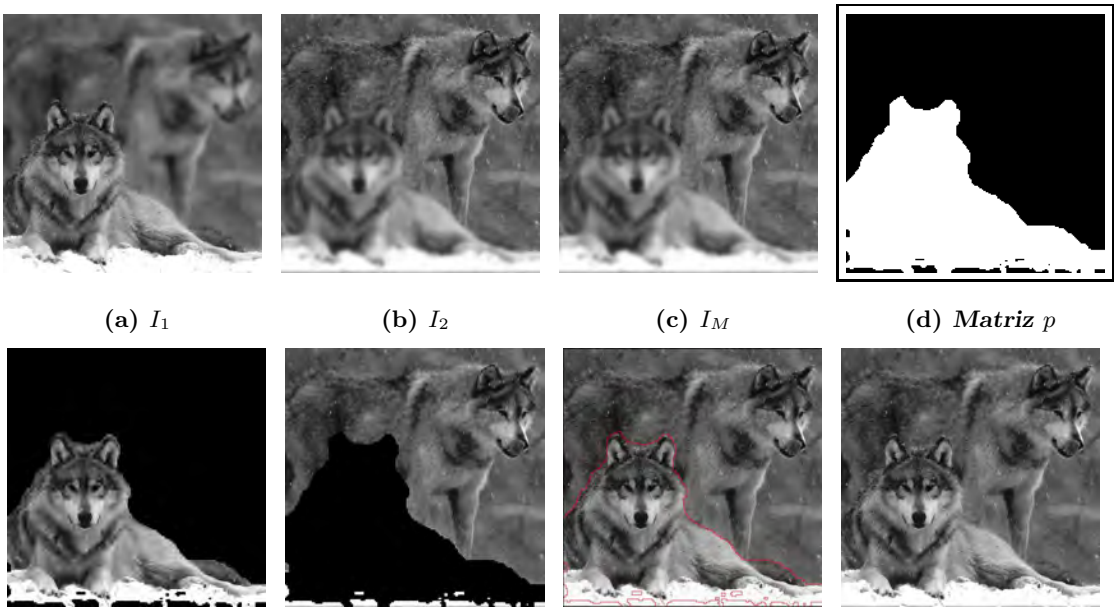


(e) ***Parte nítida de*** I_1**. (f)** ***Parte nítida de*** I_M**.** **(g)** ***Frontera.*** **(h)** ***Resultado.***

Figura 6.6: Resultados de aplicar el proceso de fusión sobre las imágenes de los relojes.

Lo primero que se hace es aplicar el método SSD-ARC como técnica de registro de las imágenes multi-foco, con lo que se logra obtener la imagen registrada a la que llamamos I_M mostrada en la Figura 6.6(c), la cual sustituye a I_2 para el proceso de fusión. Se aplica el Algoritmo 3 a las imágenes I_1 e I_M, con $w = 4, \Delta = 1, \epsilon = 0.01$ y $s_{max} = 5$, con lo que se logra obtener la matriz p mostrada en la Figura 6.6(d) y la segmentación de la parte nítida de I_1 y de I_2 mostrada en las Figuras 6.6(e) y 6.6(f). Al aplicar la regla de fusión dada en la ecuación (4.19) usando la matriz p calculada se logra la fusión de las imágenes mostradas en 6.6(g) y 6.6(h). En la Figura 6.6(g) se muestra la frontera de la segmentación, nótese que la frontera de la segmentación se aproxima mucho a la frontera entre los dos relojes.

En la Figura 6.7 se muestra el resultado para las imágenes multi-foco sintéticas de los lobos que ya se trataron en la Sección 4.2. En éstas aparecen un par de lobos y se creó el par de imágenes multi-foco simulando que en cada fotografía se enfocó en un lobo diferente. Cabe hacer notar que la matriz p calculada es prácticamente la misma que la exacta mostrada en la Figura 4.11, lo que indica que el algoritmo 3 funciona correctamente.



(a) I_1 **(b)** I_2 **(c)** I_M **(d)** ***Matriz*** p

(e) ***Parte nítida de*** I_1**. (f)** ***Parte nítida de*** I_M**.** **(g)** ***Frontera.*** **(h)** ***Resultado.***

Figura 6.7: Resultados de aplicar el proceso de fusión sobre las imágenes de los lobos con $w = 4, \Delta = 3, \epsilon = 0.01$ y $s_{max} = 1$.

Para el ejemplo de los lobos, en [Calderón y Garnica, 2014] se reportan resultados de aplicar un método similar usando la función NMaximize del software Wolfram Mathema-

tica y se reporta un porcentaje de acierto del 98 % con un tiempo de 94 minutos, mientras que nuestra solución tiene el 97.5 % de acierto, pero en un tiempo de 4.5 minutos, que es aproximadamente 21 veces menor al reportado en dicho artículo.

Ahora presentamos otro par de imágenes multi-foco creadas sintéticamente usando la imagen de Lena, que es muy conocida en el procesamiento de imágenes y usando la matriz p_0 con forma de un sol mostrada en la Figura 6.8, esto con el fin de probar el desempeño del algoritmo 3 y verificar que es capaz de segmentar de manera adecuada los picos de la imagen del sol.

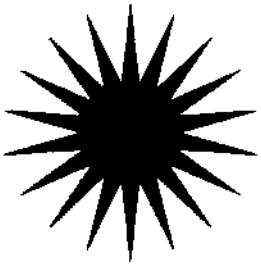
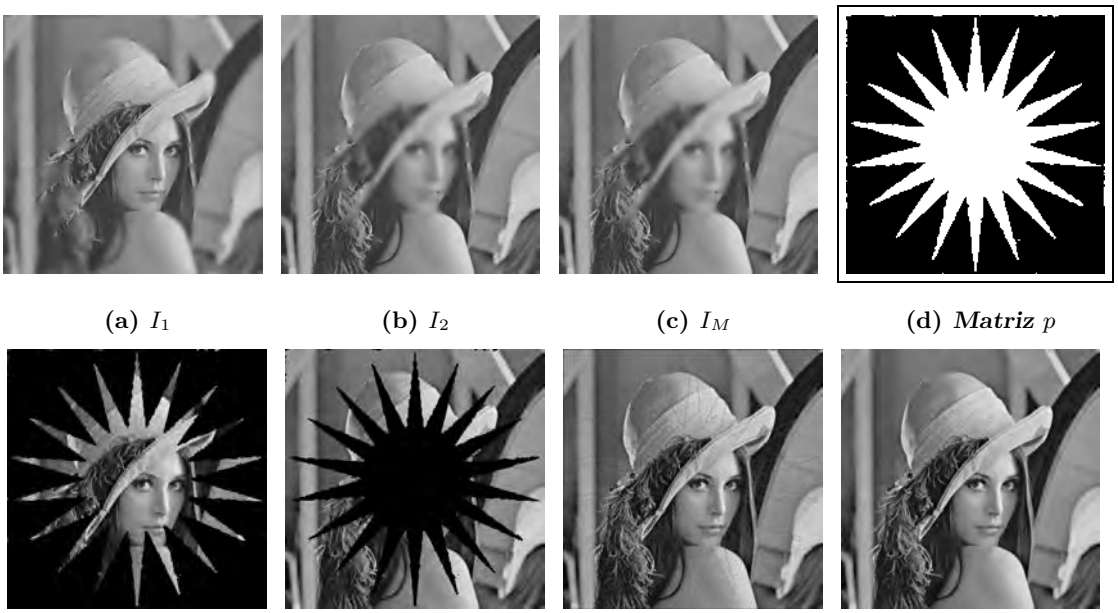


Figura 6.8: Matriz p_0 usada para calcular el par de imágenes multi-foco sintético mostrado en las Figuras 6.9(a) y 6.9(b)



(a) I_1 **(b)** I_2 **(c)** I_M **(d)** ***Matriz*** p

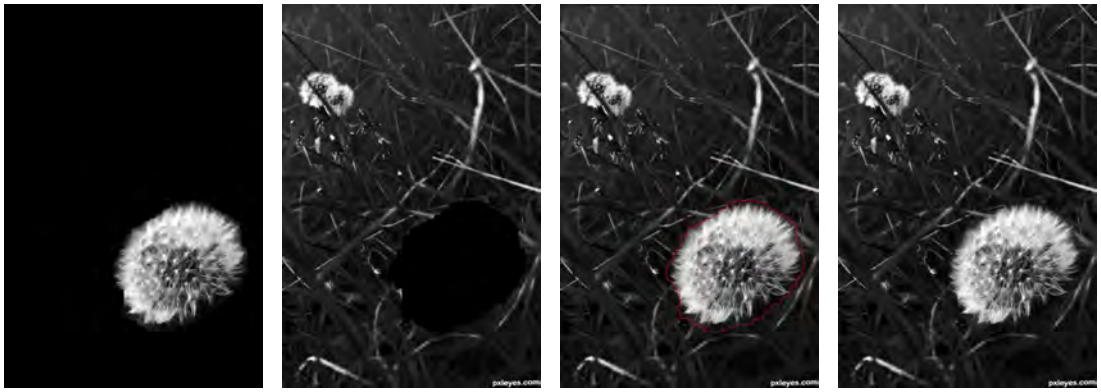
(e) ***Parte nítida de*** I_1**.** **(f)** ***Parte nítida de*** I_M**.** **(g)** ***Frontera.*** **(h)** ***Resultado.***

Figura 6.9: Resultados de aplicar el proceso de fusión sobre las imágenes sintéticas de Lena con $w = 4, \Delta = 3, \epsilon = 0.01$ y $s_{max} = 1$.

Dadas las imágenes mostradas en las Figuras 6.9(a) y 6.9(b) a las que se aplica el proceso de fusión y registro de imágenes, logrando el resultado mostrado en la Figura 6.9(h), nótese que el resultado es muy aproximado al esperado, pues la Figura 6.9 luce totalmente nítida y la matriz de pesos obtenida es prácticamente exacta, pues tiene un porcentaje de acierto de 98.6 %.

Ahora se presenta la imagen de un diente de león mostrada en las Figuras 6.10(a) y 6.10(b) a las que se aplica el proceso de registro para obtener imagen de la Figura 6.10(c). Se aplica el Algoritmo 3 a las imágenes I_1 e I_M, con lo que se logra el resultado mostrado en la Figura 6.10, en la que se puede notar que se logra extraer la flor de I_1 y fusionarla con el fondo de la escena extraído de I_M, con lo que el resultado luce nítido en toda la imagen.

(a) I_1 **(b)** I_2 **(c)** I_M **(d)** ***Matriz*** p



(e) ***Parte nítida de*** I_1**. (f)** ***Parte nítida de*** I_M**.** **(g)** ***Frontera.*** **(h)** ***Resultado.***

Figura 6.10: Resultados de aplicar el proceso de fusión sobre las imágenes del diente de león con $w = 4, \Delta = 3, \epsilon = 0.01$ y $s_{max} = 6$.

Analizando el ejemplo de las imágenes mostradas en las Figuras 6.10(a) y 6.10(b) podemos ver que a pesar del grave problema de registro que tienen las imágenes y la diferencia de tamaño que tiene la flor entre I_1 e I_2, el proceso de fusión se logra de manera adecuada.

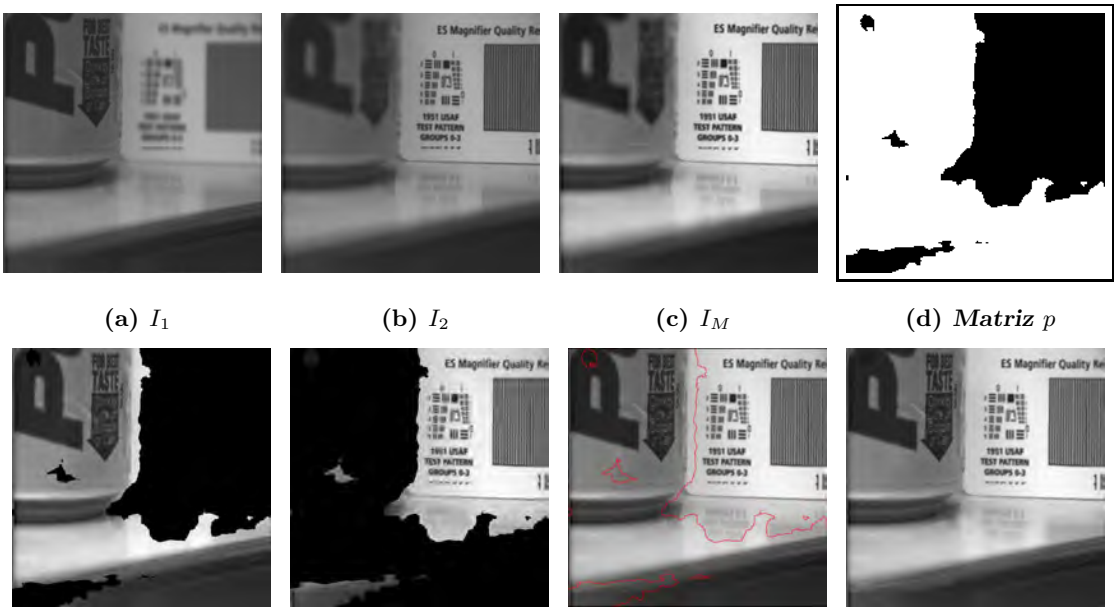
El siguiente ejemplo corresponde a las imágenes de la escultura que se trataron en inicio de éste capítulo, nótese que el resultado mostrado en 6.11(h) no presenta los artefactos que presentaba aplicando el algoritmo de fusión de imágenes sin aplicar antes el registro de imágenes. Además se puede ver que en la matriz de pesos mostrada en la Figura 6.11(d), se indica que la parte que está entre los pies de los personajes de las esculturas, debe ser tomado de la imagen I_M, lo cual es correcto porque pertenece al fondo de la escena.

(a) I_1 **(b)** I_2 **(c)** I_M **(d)** ***Matriz*** p

(e) ***Parte nítida de*** I_1**. (f)** ***Parte nítida de*** I_M**.** **(g)** ***Frontera.*** **(h)** ***Resultado.***

Figura 6.11: Resultados de aplicar el proceso de fusión sobre las imágenes de la escultura con $w = 4, \Delta = 3, \epsilon = 0.01$ y $s_{max} = 3$.

En el siguiente ejemplo se muestra la fusión de las imágenes mostradas en las Figuras 6.12(a) y 6.12(b) en las que aparece una lata de refresco y una caja con un código de barras. En la primera fotografía se ha enfocado la cámara en la lata de refresco y en la segunda fotografía se enfoca en la caja de con el código de barras. Nótese que la segmentación aproxima de muy buena manera la frontera entre los objetos y además se logra detectar que incluso el reflejo de la caja es más nítido en I_M que en I_1 y aunque aparecen errores en la imagen, dichos errores se deben a la falta de textura en ciertas regiones de las imágenes multi-foco las cuales no alcanzaron a ser abarcadas por el tamaño de la ventana.

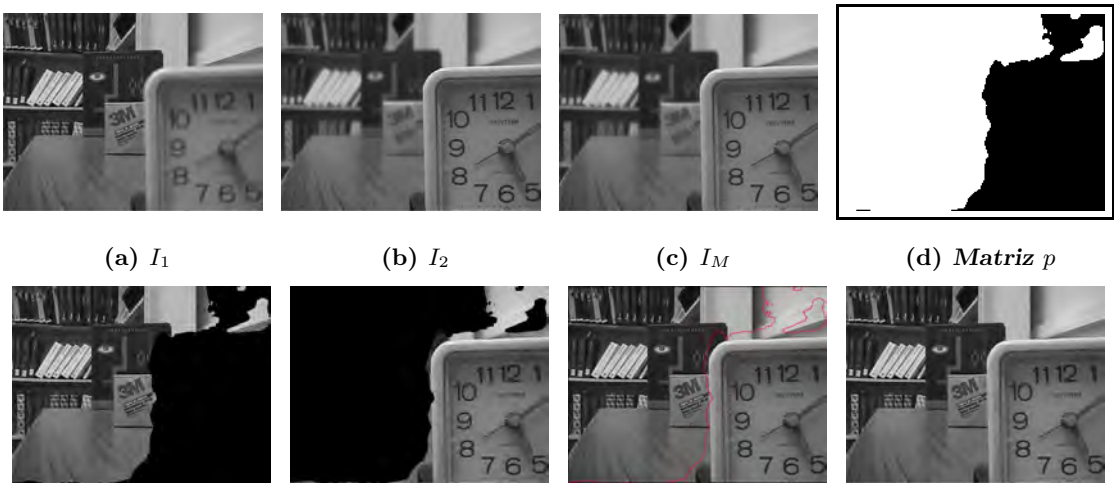


(a) I_1 **(b)** I_2 **(c)** I_M **(d)** ***Matriz*** p

(e) ***Parte nítida de*** I_1**. (f)** ***Parte nítida de*** I_M**.** **(g)** ***Frontera.*** **(h)** ***Resultado.***

Figura 6.12: Resultados de aplicar el proceso de fusión sobre las imágenes de la lata de refresco y la caja con el código de barras, con $w = 4, \Delta = 3, \epsilon = 0.01$ y $s_{max} = 9$.

Ahora presentamos un ejemplo en el que aparece una escena con un reloj y en el fondo de la escena aparecen unos libros. En la fotografía mostrada en la Figura 6.13(a) se



(a) I_1 **(b)** I_2 **(c)** I_M **(d)** ***Matriz*** p

(e) ***Parte nítida de*** I_1**. (f)** ***Parte nítida de*** I_M**.** **(g)** ***Frontera.*** **(h)** ***Resultado.***

Figura 6.13: Resultados de aplicar el proceso de fusión sobre las imágenes del reloj y los libros con $w = 4, \Delta = 3, \epsilon = 0.01$ y $s_{max} = 10$.

ha enfocado en el reloj y en la fotografía mostrada en la Figura 6.13(b) se ha enfocado en el fondo de la escena. Se puede observar en la imagen I_F resultante mostrada en la Figura 6.13 que el resultado luce nítido en toda el área de la imagen.

6.2.1. Fusión de más de dos imágenes multi-foco

Presentamos dos ejemplos obtenidos de [Orozco, 2013], el conjunto I contiene tres imágenes multi-foco y se pretende lograr la fusión de las mismas. El primer ejemplo corresponde a las imágenes de un manzana, un frutero y un objeto en el fondo las cuales se muestran en la Figura 6.14.

(a) I_1 **(b)** I_2 **(c)** I_3

Figura 6.14: Imágenes multi-foco, ejemplo de la manzana y el florero.

Dado que para este caso tenemos un conjunto con tres imágenes multi-foco, el proceso implementado para obtener la imagen fusionada es el mostrado en la figura 6.15, en el que se ilustra que el proceso de fusión se realiza de la siguiente manera:

- Primero se aplica el proceso de registro para alinear I_1 e I_3 tomando como referencia I_2, con lo que se obtienen las imágenes registradas I_{M1} e I_{M2} respectivamente.

- Se aplica el proceso de fusión a las imágenes I_2 e I_{M2} con lo que se obtiene como resultado la imagen I_{F1}.

- La imagen I_{F1} que es el resultado de la fusión de I_2 e I_{F1} se fusiona con I_{M1} y se obtiene como resultado la imagen fusionada final I_F, la cual debe tener las partes más nítidas de cada imagen multi-foco.

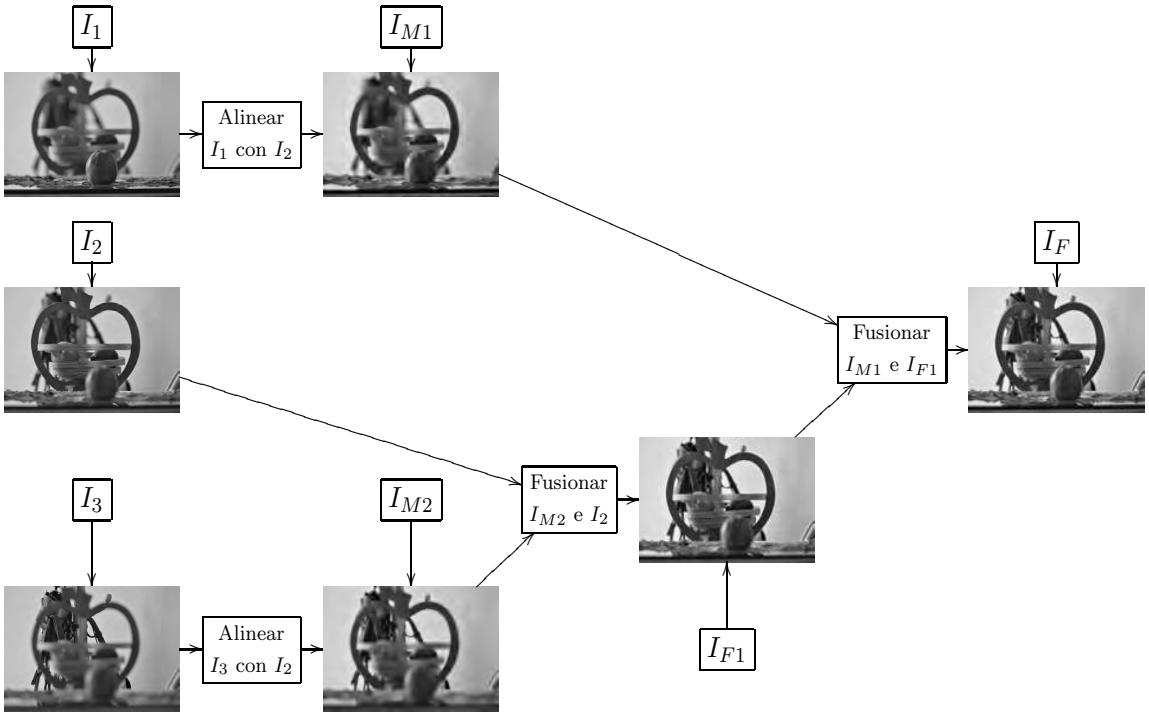
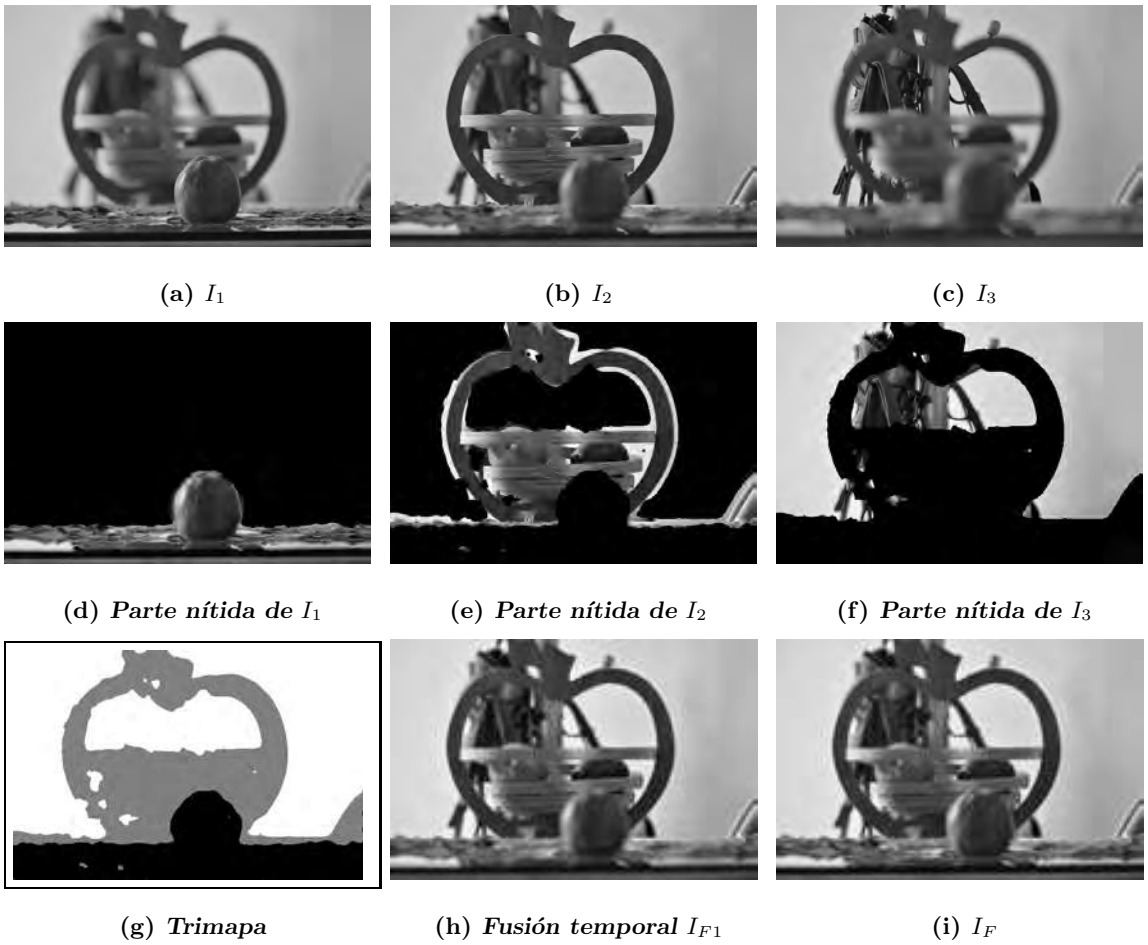


Figura 6.15: Esquema del proceso para fusionar más de dos imágenes multi-foco.

El proceso planteado en el esquema anterior se puede extender para realizar la fusión de tantas imágenes como se tenga en el conjunto I, realizando la fusión de pares de imágenes y después fusionando los resultados de dichos procesos de fusión, hasta llegar a tener un par de imágenes y al fusionarlas se obtenga la imagen fusionada final.

En las Figuras 6.16(a), 6.16(b) y 6.16(c) se muestran las imágenes multi-foco I_1, I_2 e I_3 respectivamente para ejemplo de las imágenes multi-foco de la manzana y el frutero. En la imagen de la Figura 6.16(d) se muestra la parte segmentada de I_1, en la Figura 6.16(e) se muestra la parte segmentada de I_2 y en la Figura 6.16(f) se muestra la parte que se detectó como más nítida den I_3. Se puede notar que en los tres casos se logra detectar de muy buena manera las partes más nítidas de cada imagen. En la Figura 6.16(g) se muestra el mapa de decisión obtenido, el cual se logró al sumar las matrices de pesos calculadas en los dos procesos de fusión realizados. Nótese en dicha imagen que la región en negro corresponde a la parte nítida de I_1, la región de marcada en gris indica la región que es más

nítida en I_2 que en I_1 e I_3 y la región marcada con blanco indica las regiones que son más nítidas en I_3 que en cualquiera de las otras dos imágenes.



(a) I_1 **(b)** I_2 **(c)** I_3

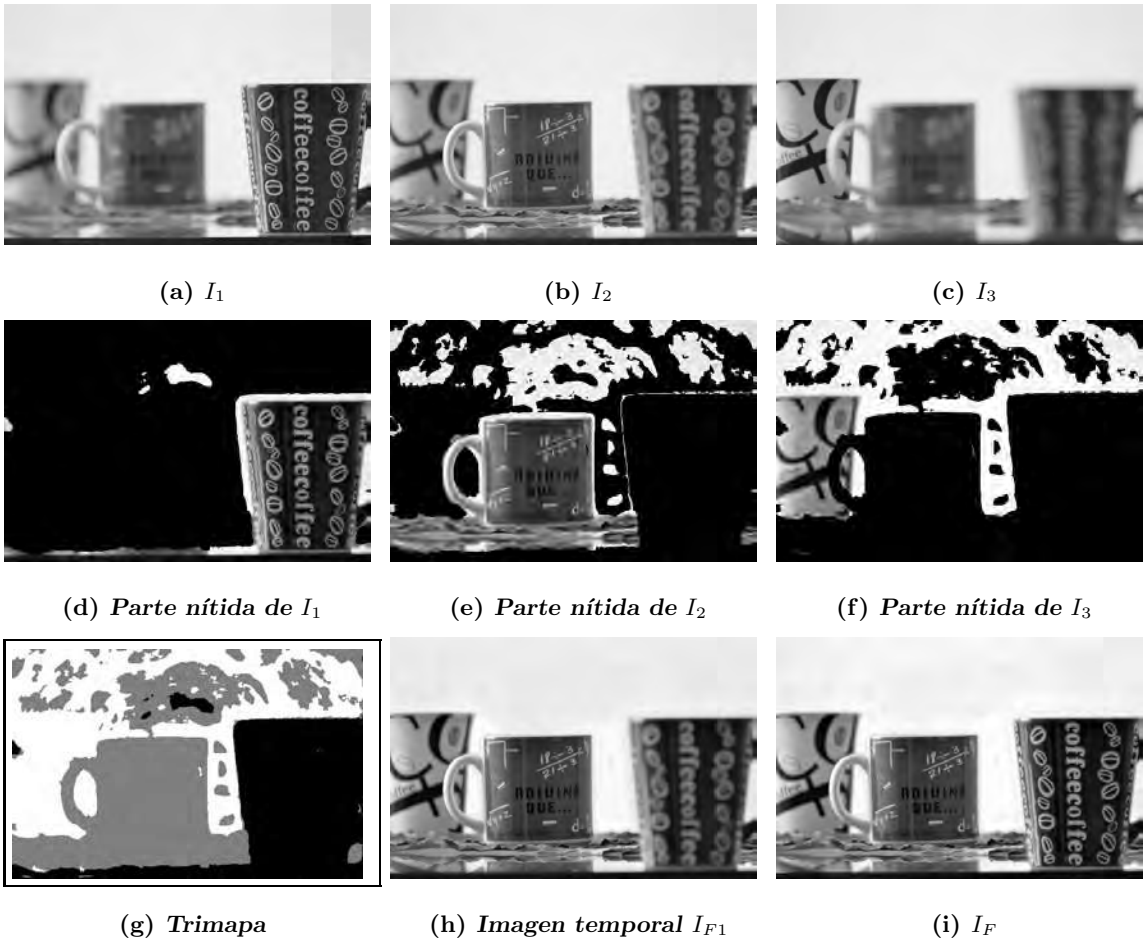
(d) ***Parte nítida de*** I_1 **(e)** ***Parte nítida de*** I_2 **(f)** ***Parte nítida de*** I_3

(g) ***Trimapa*** **(h)** ***Fusión temporal*** I_{F1} **(i)** I_F

Figura 6.16: Ejemplo de fusión de tres imágenes multi-foco, ejemplo de la manzana y el frutero.

El segundo ejemplo de fusión para conjuntos con más de dos imágenes corresponde a las fotografías de tres tazas en donde en cada fotografía se enfoca a una taza diferente. Se debe hacer notar que la segmentación de las imágenes fuente se logra con muy buena aproximación a cada una de las taza. Al unir las partes segmentadas de cada imagen multi-foco y formar con ellas la imagen fusionada I_F, dicha imagen luce completamente nítida y las tres tazas aparecen de forma clara e incluso el cristal sobre el que se encuentran las tazas es segmentado de tal manera que luce también nítido en la imagen final. Cabe mencionar

que aunque la región del fondo de la escena no se logró segmentar de la mejor manera, esto se debe que por la falta de nitidez resulta muy difícil establecer qué imagen posee mayor nitidez en dicha área, una solución para resolver este problema es usar ventanas de mayor tamaño para lograr la coherencia espacial en áreas con poca textura. Sin embargo el aplicar ventanas muy grandes implica el consumo de una gran cantidad de recursos de cómputo.



(a) I_1 **(b)** I_2 **(c)** I_3

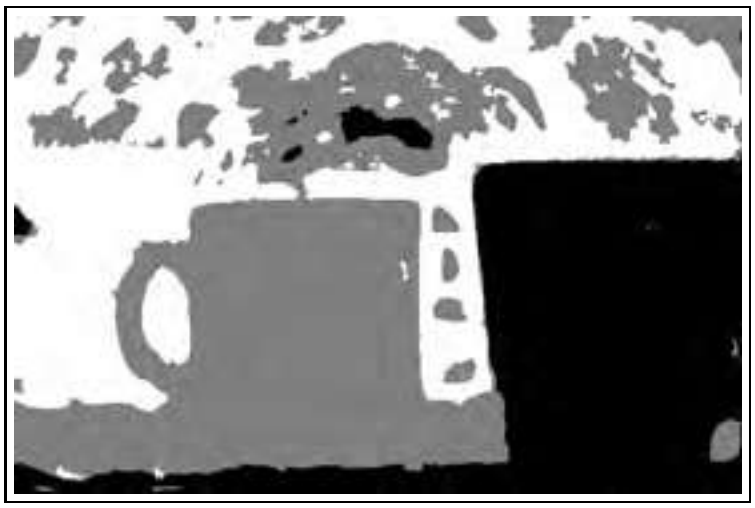
(d) ***Parte nítida de*** I_1 **(e)** ***Parte nítida de*** I_2 **(f)** ***Parte nítida de*** I_3

(g) ***Trimapa*** **(h)** ***Imagen temporal*** I_{F1} **(i)** I_F

Figura 6.17: Ejemplo de fusión de tres imágenes multi-foco, ejemplo de las tazas.

En las Figura 6.18 y 6.19 se muestran los resultados obtenidos con el algoritmo presentado en este documento y los resultados reportados en [Orozco, 2013] en el que se hace uso de filtrado en frecuencia para obtener un mapa de decisión inicial el cual se mejora usando técnicas de segmentación. Se puede notar que con el uso del algoritmo CPV-PSH presentado, se logran resultados mejores que los reportados en [Orozco, 2013]. En el caso del

mapa de decisión para el ejemplo de las tazas, con el algoritmo CPV-PSH se logra delimitar muy bien la región donde se encuentra cada taza mientras que en los resultados de [Orozco, 2013] la taza de enfrente que es más nítida en I_1, no se logra delimitar de forma correcta.



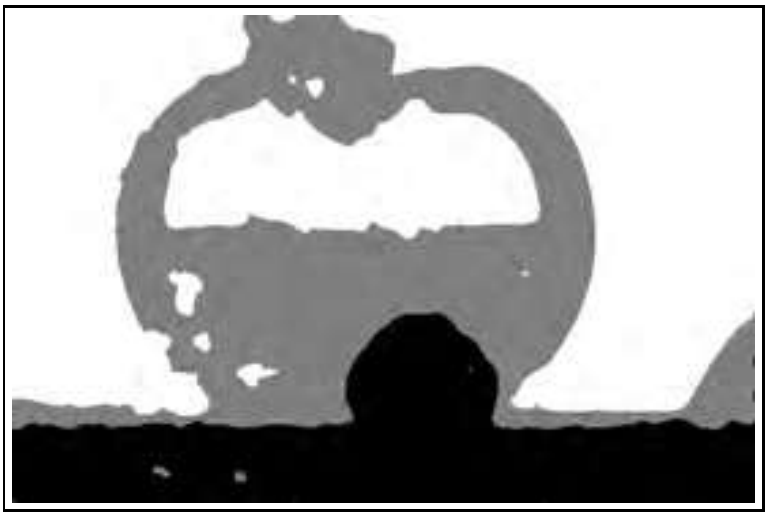
(a) ***Mapa de decisión obtenido con CPV-PSH***

(b) ***Mapa de decisión obtenido usando filtrado en frecuencia y técnicas de segmentación***

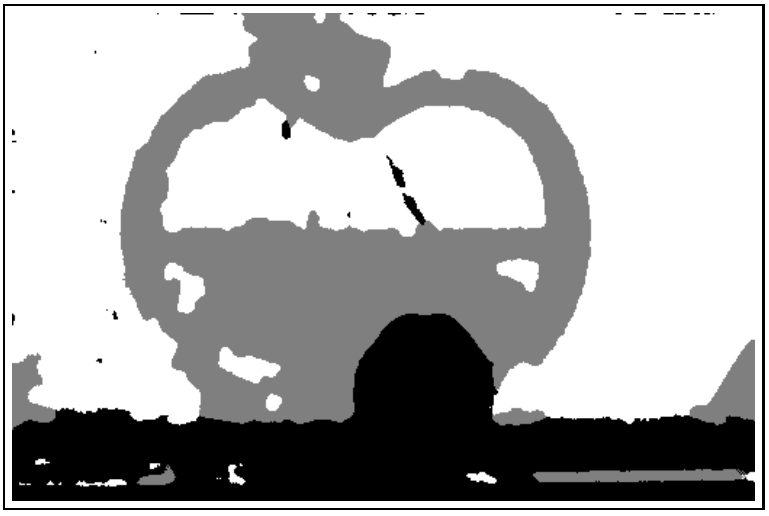
Figura 6.18: Mapas de decisión para el ejemplo de las tazas.

Para el caso de del ejemplo de las imágenes multi-foco de la manzana y el florero se logran resultados muy similares a los reportados por Orozco, sin embargo en el mapa de decisión reportado en dicho trabajo aparecen algunas regiones inconsistentes que no

aparecen aplicando el Algoritmo CPV-PSH. Por desgracia en [Orozco, 2013] no se reportan los valores de tiempo consumidos por la técnica planteada en dicho trabajo, por lo que no podemos comparar los resultados en ese aspecto.



(a) ***Mapa de decisión calculado con CPV-PSH***



(b) ***Mapa de desichión obtenido usando filtrado en frecuencia y técnicas de segmentación***

Figura 6.19: Mapas de decisión para el ejemplo de la manzana y el florero.

6.3. Conclusiones del capítulo

En este capítulo se mostró un ejemplo de que el resultado de la fusión de imágenes multi-foco, es mejor si se aplica un método de registro de imágenes antes de hacer la fusión que si se realiza la fusión sin registrar previamente las imágenes. Sin embargo también se mostró el ejemplo de los relojes en el que se puede resaltar que al aplicar el Algoritmo CPV-PSH, se pueden lograr resultados relativamente buenos aún si las imágenes no están registradas aunque el resultado sigue siendo mejor si se aplica primero el registro de imágenes. Se presentaron varios ejemplos del funcionamiento del Algoritmo 3 **(CPV-PSH)** para la fusión de las imágenes multi-foco. Se ha demostrado al presentar dichos ejemplos que el Algoritmo 3 funciona de muy buena manera para la fusión de imágenes multi-foco. Además se han logrado superar los resultados mostrados en [Calderón y Garnica, 2014] y los reportados en [Orozco, 2013] los cuales son los únicos trabajos que conocemos en los que se reporte una matriz de pesos similar a la calculada en este trabajo.

Capítulo 7

Conclusiones

7.1. Conclusiones

En este trabajo se han analizado distintas técnicas para medir el nivel de enfoque o desenfoque de una imagen.

Se ha planteado una función que permite crear la imagen fusionada como una combinación lineal de las imágenes multi-foco, del análisis de las medidas de calidad de una imagen se ha decidido usar un kernel de convolución que es una versión discretizada del Laplaciano. Se planteó una función lineal a optimizar, la cual nos permite garantizar que el nivel de enfoque en la imagen fusionada sea la máxima posible; se ha decidido agregar restricciones a la función a optimizar, con lo que se ha logrado mejorar la coherencia espacial en la imagen fusionada.

El cambio de enfoque genera que las imágenes multi-foco genera una desalineación espacial en las imágenes multi-foco, además se observó que la transformación que sufren las imágenes al variar el nivel de enfoque no es una transformación rígida, por lo que no se puede resolver con las técnicas comunes de registro paramétrico, lo que nos llevó a usar un método SSD-ARC. SSD-ARC es un método de registro no paramétrico que ha demostrado tener un muy buen desempeño en transformaciones no rígidas. Con el uso de este método de registro se ha logrado alinear las imágenes del conjunto I y con ello mejorar el resultado de la fusión de las imágenes.

El problema de optimización lineal con restricciones se ha planteado y resuelto con el método SIMPLEX, el cual es uno de los métodos que ha demostrado tener un excelente desempeño en problemas de optimización lineal. Sin embargo, por la naturaleza, la cantidad de variables y de restricciones la matriz usada por el método hace imposible para términos prácticos resolver el problema de manera directa para toda la imagen. Dicha situación nos llevó a plantear el problema con el uso de ventanas deslizantes con traslape, lo que nos permite aplicar la técnica por ventanas con un costo mucho menor que si se aplicara para toda la imagen.

Finalmente se observó que el usar técnicas de sub-muestreo ofrece grandes ventajas, pues el sub-muestrear la imagen nos permite, aplicar el método en un contexto con menos variables y se observó que la solución obtenida en las imágenes sub-muestreadas se acerca mucho a la solución del problema en la resolución original.

Combinando técnicas de sub-muestreo, optimización con restricciones y ventanas deslizantes con traslape se han logrado muy buenos resultados, logrando para imágenes sintéticas un acierto del 98.65 %. Aunque en las imágenes reales no hay forma de medir el porcentaje de acierto, se observa que la solución obtenida por nuestro método cumple con los objetivos presentados al inicio del documento de discriminar de manera automática las áreas enfocadas de cada imagen, de segmentar las imágenes garantizando la coherencia espacial y el obtener una imagen fusionada que aparenta haber sido tomada con una lente con profundidad de campo infinita.

Adicionalmente si comparamos nuestros resultados con los resultados mostrados en [Calderón y Garnica, 2014] podemos concluir que el método propuesto en este documento logra un resultado prácticamente igual al presentado en dicho artículo pero con un costo 20 veces menor en tiempo para las imágenes de los lobos, en un tamaño de 512×512. No podemos comparar los nuestros resultados con otros del estado del arte, porque hasta donde conocemos, no existe trabajos que reporten matrices de pesos similares a las calculadas en este trabajo.

7.2. Trabajos futuros

En el área de la fusión de imágenes multi-foco existe aún mucho camino por recorrer, pensando en encontrar algoritmos que permitan realizar la fusión de las imágenes de manera más rápida y con un desempeño mejor, que permita que la imagen fusionada muestre la mayor nitidez posible.

1. Una tarea trascendental para la solución del problema de fusión de imágenes multi-foco es encontrar una medida del nivel de enfoque, robusta e independiente de la imagen en cuestión. La mayoría de los métodos presentados en la literatura, para medir el nivel de enfoque, son muy sensibles al ruido, por lo que un trabajo interesante puede ser el buscar una combinación adecuada de los métodos existentes o un método nuevo que pudiera tener mejor desempeño al momento de medir el nivel de enfoque o desenfoque en una imagen. Se propone investigación con filtros pasa banda que permitan rechazar, por un lado las frecuencias muy altas (ruido) y por otro lado las frecuencias muy bajas (regiones desenfocadas).

2. En el estado del arte es posible encontrar trabajos que plantean que para realizar la fusión de imágenes multi-foco se debe hacer la división de las imágenes fuente en bloques, mientras que otros trabajos plantean que el resultado es mejor si se hace el análisis pixel a pixel. Sin embargo, podemos decir que el utilizar bloques nos ofrece una mejor coherencia espacial, pero el hacer el análisis pixel a pixel permite tener mejor definición en las fronteras de la segmentación. Lo que se plantea como trabajo a futuro es el usar bloques o ventanas de tamaño variante para garantizar la coherencia espacial y al mismo tiempo lograr una mejor definición en las fronteras de la segmentación de las imágenes.

3. Los procesos planteados para realizar la fusión de imágenes multi-foco, que mejores resultados han arrojado, no permiten la aplicación en tiempo real, por lo que un desafío es usar técnicas que permitan hacer el cálculo de la matriz de pesos p en un tiempo menor, sin afectar el desempeño del método que en este trabajo se ha planteado. Una técnica que se ha probado con muy buenos resultados es el uso de

imágenes incrementales, lo que permite realizar el cálculo de la matriz p en un tiempo mucho menor que el método planteado en este trabajo, por lo que se pretende seguir la investigación esta línea para lograr mejorar el método aquí planteado.

Referencias

Bae, S. y Durand, F. (2007). Defocus magnification. *Computer Graphics Forum*, 26(3):571–579.

Brown, L. G. (1992). A survey of image registration techniques. *ACM Comput. Surv.*, 24(4):325–376.

Calderón, F. y Flores, J. (2010). Solution to the registration problem using differential evolution and ssd-arc function. In *Artificial Intelligence (MICAI), 2010 Ninth Mexican International Conference on*, pages 3–10.

Calderón, F., Flores, J., y Romero, L. (2007). Robust parametric image registration. In *Hybrid Evolutionary Algorithms*, volume 75 of *Studies in Computational Intelligence*, pages 337–360. Springer Berlin Heidelberg.

Calderón, F. y Garnica, A. (2014). Multi focus image fusion based on linear combination of images. pages 1–7.

Calderón, F. y Marroquín, J. L. (2003). Un nuevo algoritmo para el cálculo de flujo óptico y su aplicación al registro de imágenes.

Calderón, F. y Romero, L. (2004). Non-parametric registration as a way to obtain an accurate camera calibration. In Monroy, R., Arroyo-Figueroa, G., Sucar, L., y Sossa, H., editors, *MICAI 2004: Advances in Artificial Intelligence*, volume 2972 of *Lecture Notes in Computer Science*, pages 584–591. Springer Berlin Heidelberg.

Calderón, F., Romero, L., y Flores, J. (2006a). Ga–ssd–arc–nlm for parametric image

registration. In *Progress in Pattern Recognition, Image Analysis and Applications*, pages 227–236. Springer.

Calderón, F., Romero, L., y Flores, J. (2006b). *GA-SSD-ARC-NLM for Parametric Image Registration*, volume 4225, pages 227–236. Springer Berlin Heidelberg, Berlin, Heidelberg.

Cao, L., Jin, L., Tao, H., Li, G., Zhuang, Z., y Zhang, Y. (2015). Multi-focus image fusion based on spatial frequency in discrete cosine transform domain. *Signal Processing Letters, IEEE*, 22(2):220–224.

Chun, M. G. y Kong, S. G. (2014). Focusing in thermal imagery using morphological gradient operator. *Pattern Recognition Letters*, 38(0):20 – 25.

De, I., Chanda, B., y Chattopadhyay, B. (2006). Enhancing effective depth-of-field by image fusion using mathematical morphology. *Image and Vision Computing*, 24(12):1278 – 1287.

Elder, J. y Zucker, S. (1998). Local scale control for edge detection and blur estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(7):699–716.

Eskicioglu, A. y Fisher, P. (1995). Image quality measures and their performance. *Communications, IEEE Transactions on*, 43(12):2959–2965.

Fischer, B. y Modersitzki, J. (2008). Ill-posed medicine an introduction to image registration. *Inverse Problems*, 24(3):034008.

Ghali, S. (2008). *Affine Transformations*, pages 35–50. Springer London, London.

González, R. C. y Woods, R. E. (2008). *Digital image processing*. Prentice Hall, Upper Saddle River, N.J.

Goshtasby, A. A. y Nikolov, S. (2007). Image fusion: Advances in the state of the art. *Information Fusion*, 8(2):114 – 118. Special Issue on Image Fusion: Advances in the State of the Art.

Guo, D., Yan, J., y Qu, X. (2015). High quality multi-focus image fusion using self-similarity and depth information. *Optics Communications*, 338(0):138 – 144.

Huang, W. y Jing, Z. (2007). Evaluation of focus measures in multi-focus image fusion. *Pattern Recognition Letters*, 28(4):493 – 500.

Kuthirummal, S., Nagahara, H., Zhou, C., y Nayar, S. (2011). Flexible depth of field photography. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(1):58–71.

Li, S., Kang, X., Hu, J., y Yang, B. (2013). Image matting for fusion of multi-focus images in dynamic scenes. *Information Fusion*, 14(2):147 – 162.

Li, S., Kwok, J. T., y Wang, Y. (2001). Combination of images with diverse focuses using the spatial frequency. *Information Fusion*, 2(3):169 – 176.

Li, S., Kwok, J. T., y Wang, Y. (2002). Multifocus image fusion using artificial neural networks. *Pattern Recognition Letters*, 23(8):985 – 997.

Li, S. y Yang, B. (2008). Multifocus image fusion by combining curvelet and wavelet transform. *Pattern Recognition Letters*, 29(9):1295–1301.

Liu, Y., Liu, S., y Wang, Z. (2015). Multi-focus image fusion with dense {SIFT}. *Information Fusion*, 23(0):139 – 155.

Liu, Y. y Yu, F. (2015). An automatic image fusion algorithm for unregistered multiply multi-focus images. *Optics Communications*, 341(0):101 – 113.

Lucas, B. D., Kanade, T., et al. (1981). An iterative image registration technique with an application to stereo vision. In *IJCAI*, volume 81, pages 674–679.

Ma, Y., Zhan, K., Wang, Z., y service), S. O. (2011). Applications of pulse-coupled neural networks.

Malviya, A. y Bhirud, S. (2009). Wavelet based multi-focus image fusion. In *Methods and Models in Computer Science, 2009. ICM2CS 2009. Proceeding of International Conference on*, pages 1–6.

Nayar, S. y Nakagawa, Y. (1994). Shape from focus. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(8):824–831.

Orozco, R. I. (2013). Fusión de imágenes multifoco por medio de filtrado de regiones de alta y baja frecuencia. Tesis de Maestría, División de Estudios de Postgrado. Facultad de Ingeniería Eléctrica. UMSNH, Morelia Michoacan Mexico.

Panusopone, K., Cheevasuvi, F., y Rao, K. (1995). Adaptive subsampling for image compression. In *Signals, Systems and Computers, 1995. 1995 Conference Record of the Twenty-Ninth Asilomar Conference on*, volume 1, pages 239–243 vol.1.

Pennec, X., Perchant, A., y Ayache, N. (2007). *Non-parametric Diffeomorphic Image Registration with the Demons Algorithm*, volume 4792, pages 319–326. Springer Berlin Heidelberg, Berlin, Heidelberg.

Pertuz, S., Puig, D., y Garcia, M. A. (2013). Analysis of focus measure operators for shape-from-focus. *Pattern Recognition*, 46(5):1415 – 1432.

Potmesil, M. y Chakravarty, I. (1981). A lens and aperture camera model for synthetic image generation. *SIGGRAPH Comput. Graph.*, 15(3):297–305.

Qu, X., Hou, Y., Lam, F., Guo, D., Zhong, J., y Chen, Z. (2014). Magnetic resonance image reconstruction from undersampled measurements using a patch-based nonlocal operator. *Medical Image Analysis*, 18(6):843 – 856. Sparse Methods for Signal Reconstruction and Medical Image Analysis.

Riaz, M., Park, S., Ahmad, M., Rasheed, W., y Park, J. (2008). Generalized laplacian as focus measure. In Bubak, M., van Albada, G., Dongarra, J., y Sloot, P., editors, *Computational Science ICCS 2008*, volume 5101 of *Lecture Notes in Computer Science*, pages 1013–1021. Springer Berlin Heidelberg.

Rivera, M. y Marroquin, J. (2000). The adaptive-rest condition spring system: an edge-preserving regularization technique. In *Image Processing, 2000. Proceedings. 2000 International Conference on*, volume 3, pages 805–808 vol.3.

Rivera, M., Ocegueda, O., y Marroquin, J. (2007). Entropy-controlled quadratic markov measure field models for efficient image segmentation. *Image Processing, IEEE Transactions on*, 16(12):3047–3057.

Romeny, B. M. (1994). *Geometry-driven diffusion in computer vision*, volume 320. Kluwer Academic Publishers Dordrecht.

Sezan, M., Pavlovic, G., Tekalp, A., y Erdem, A. (1991). On modeling the focus blur in image restoration. In *Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991 International Conference on*, pages 2485–2488 vol.4.

Shi, W., Zhu, C., Tian, Y., y Nichol, J. (2005). Wavelet-based image fusion and quality assessment. *International Journal of Applied Earth Observation and Geoinformation*, 6(3-4):241 – 251.

Shirvaikar, M. (2004). An optimal measure for camera focus and exposure. In *System Theory, 2004. Proceedings of the Thirty-Sixth Southeastern Symposium on*, pages 472–475.

Starck, J.-L. y Murtagh, F. (2006). Deconvolution. In *Astronomical Image and Data Analysis*, Astronomy and Astrophysics Library, pages 71–110. Springer Berlin Heidelberg.

Subbarao, M., Choi, T., y Nikzad, A. (1993). Focusing techniques. *OPTICAL ENGINEERING*, 32(11):2824–2836.

Sun, Y., Duthaler, S., y Nelson, B. J. (2004). Autofocusing in computer microscopy: selecting the optimal focus algorithm. *Microscopy research and technique*, 65(3):139–149.

Tian, J. y Chen, L. (2010). Multi-focus image fusion using wavelet-domain statistics. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 1205–1208.

Tomasi, C. y Kanade, T. (1991). *Detection and tracking of point features*. School of Computer Science, Carnegie Mellon Univ. Pittsburgh.

Viola, P. y Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511–I–518 vol.1.

Wallace, G. K. (1991). The jpeg still picture compression standard. *Commun. ACM*, 34(4):30–44.

Wee, C.-Y. y Paramesran, R. (2007). Measure of image sharpness using eigenvalues. *Information Sciences*, 177(12):2533 – 2552.

Yang, Y. (2011). A novel {DWT} based multi-focus image fusion method. *Procedia Engineering*, 24(0):177 – 181. International Conference on Advances in Engineering 2011.

Yang, Y., Huang, S., Gao, J., y Qian, Z. (2014). Multi-focus image fusion using an effective discrete wavelet transform based algorithm. *Measurement Science Review*, 14(2):102 – 108.

Yuan, L., Sun, J., Quan, L., y Shum, H.-Y. (2008). Progressive inter-scale and intra-scale non-blind image deconvolution. *ACM Transactions on Graphics*, 27(3):74.

Zhou, L., Ji, G., Shi, C., Feng, C., y Nian, R. (2006). *A Multi-focus Image Fusion Method Based on Image Information Features and the Artificial Neural Networks*, volume 344, pages 747–752. Springer Berlin Heidelberg, Berlin, Heidelberg.

Zhou, X., Zhou, F., Bai, X., y Xue, B. (2014a). A boundary condition based deconvolution framework for image deblurring. *Journal of Computational and Applied Mathematics*, 261(0):14 – 29.

Zhou, Z., Li, S., y Wang, B. (2014b). Multi-scale weighted gradient-based fusion for multi-focus images. *Information Fusion*, 20(0):60 – 72.

Zhuo, S. y Sim, T. (2011). Defocus map estimation from a single image. *Pattern Recognition*, 44(9):1852 – 1858. Computer Analysis of Images and Patterns.